Preserving Structure in Model-Free Tracking

Lu Zhang and Laurens van der Maaten

Abstract—Model-free trackers can track arbitrary objects based on a single (bounding-box) annotation of the object. Whilst the performance of model-free trackers has recently improved significantly, simultaneously tracking multiple objects with similar appearance remains very hard. In this paper, we propose a new multi-object model-free tracker (using a tracking-by-detection framework) that resolves this problem by incorporating spatial constraints between the objects. The spatial constraints are learned along with the object detectors using an online structured SVM algorithm. The experimental evaluation of our structure-preserving object tracker (SPOT) reveals substantial performance improvements in multi-object tracking. We also show that SPOT can improve the performance of single-object trackers by simultaneously tracking different parts of the object. Moreover, we show that SPOT can be used to adapt generic, model-based object detectors during tracking to tailor them towards a specific instance of that object.

Index Terms—Model-Free Tracking, Multiple-Object Tracking, Online Learning, Structured SVM.

1 Introduction

TRACKING is a fundamental problem in computer vision with applications in a wide range of domains. Whilst significant progress has been made in tracking specific objects (*e.g.*, in tracking faces [1], humans [2], [3], [4], [5], cars [6], and rigid objects [7]), the development of trackers that work well on arbitrary objects remains hard. Because manually annotating sufficient examples of all objects in the world is prohibitively expensive and time-consuming, recently, approaches for *model-free tracking* have received increased interest [8], [9]. In model-free tracking, an object of interest is manually annotated in the first frame of a video sequence, *e.g.*, using a rectangular bounding box drawn around the object. The annotated object needs to be tracked throughout the remainder of the video. Model-free tracking is a challenging task because (1) little prior information is available about the object to be tracked, (2) this information is ambiguous in the sense that the initial bounding box only approximately distinguishes the object of interest from the background, and (3) the object appearance may change drastically over time, in particular, when the object is deformable.

Usually, a tracking system comprises three main components [10]: (1) an *appearance model* that predicts the likelihood that the object is present at a particular location based on the local image appearance, (2) a *location model* that predicts the prior probability that the object is present at a particular location; and (3) a *search strategy* for finding the maximum a posteriori location of the object. In our model-free tracker, the appearance model is implemented by a classifier trained on histogram-of-gradient (HOG) features [11], the location model is based on the relative locations of the objects that are being tracked, and the search strategy is a sliding-window exhaustive search.

• *L. Zhang and L. van der Maaten are with the Department of Intelligent Systems, Delft University of Technology, Mekelweg 4, 2600 GA Delft, The Netherlands.*
E-mail: {lu.zhang, l.j.p.vandermaaten}@tudelft.nl

In many applications, it is necessary to track more than one object. For instance, in surveillance applications, one frequently needs to track particular people [3], faces [12], or cars [13] in a complex environment. A simple approach to tracking multiple objects is to run multiple instances of a single-object tracker. In this paper, we argue that this is suboptimal because such an approach fails to exploit spatial constraints between the objects. For instance, nearby people tend to walk in the same direction on the side walk, cars drive in the same direction on the freeway, and when the camera shakes, all objects will move roughly in the same direction. We show that it is practical to exploit these types of spatial relations between objects in model-free tracking. In particular, we develop a structure-preserving object tracker (SPOT) that *incorporates spatial constraints between objects* using a pictorial-structures framework [14]. We train the appearance models of all the objects and the structural constraints between these objects jointly in an online structured SVM framework [15]. Our experimental evaluations show that the incorporation of structural constraints leads to *substantial performance improvements in multi-object tracking*: SPOT performs very well on Youtube videos with significant camera motion, rapidly moving objects, object appearance changes, and occlusions. We also show that SPOT may be used to *improve single-object trackers* by using part detectors in addition to the object detector, with spatial constraints between the parts. Moreover, we show that SPOT can be used to *tailor detectors for generic objects to specific instances of that object*, which leads to performance improvements in *model-based* tracking.

In summary, our main contributions are: (1) we present a new approach that performs online learning of pictorial-structures models that incorporate spatial constraints between objects, which helps in simultaneous model-free tracking of multiple objects, (2) we show that our approach may improve the performance of single-object model-free trackers by simultaneously tracking a target object and informative parts of that object, and (3) we show that our approach can be used to tailor state-of-the-art generic object detectors to particular objects.

An earlier version of this paper appeared in [16]. Compared to the prior paper, this study contains (1) a substantial number of additional explanations and analysis, (2) various additional experiments to investigate the impact of spatial-structure preservations in tracking, and (3) a new approach for adapting existing generic-object detectors to improve their performance when tracking a specific object.

The outline of the remainder of this paper is as follows. We discuss related work in Section 2. Section 3 introduces our new SPOT tracker that incorporates spatial relations between different objects or difference object parts. We present the results of our experiments in Section 4. Section 5 concludes the paper.

2 Related Work

Our structure-preserving object tracker incorporates ideas from prior work on model-free tracking, deformable template models, and online learning for structured prediction models. Below, we briefly review relevant prior work on these three topics.

2.1 Model-Free Tracking

There exists a plethora of prior work on model-free tracking. Below, we give a concise overview of this work. For extensive reviews on model-free tracking, we refer the reader to [17], [18].

Model-free trackers can be subdivided into (1) *generative* trackers that model only the appearance of the object itself [19], [20], [21] and (2) *discriminative* trackers that model the appearance of both the object and the background [8], [9], [22], [23], [24], [25]. Because generative trackers model the appearance of the target object without considering the appearance of the background, they often fail when the background is cluttered or when multiple moving objects are present. By contrast, discriminative trackers construct a classifier that distinguishes the target object from the background. Recent results suggest that discriminative trackers outperform generative trackers [9] (similar results have been obtained for model-based trackers, *e.g.*, [26], [27]). This result is supported by theoretical results showing that discriminative models always outperform their generative counterparts on a discriminative task such as object tracking [28]. Hence, in this paper, we will focus on learning discriminative object appearance models.

Many prior studies on model-free tracking focus on exploring different feature representations for the target object, including feature representations based on points [29], [30], [31], contours [32], [33], [34], integral histograms [35], subspace learning [21], sparse representations [36], and local binary patterns [9]. In this work, we capitalize on the success of the Dalal-Triggs [11] and Felzenszwalb [6] detectors, and use HOG features instead.

Recent work on model-free tracking also focuses on developing new learning approaches to better distinguish the target object from the background. In particular, previous studies have investigated approaches based on boosting [23], [37], random forests [9], multiple instance learning [8], and structured output learning to predict object transformations [38]. Our tracker is similar to these approaches in that it updates the appearance model of the target object online. Our tracker differs from previous approaches in that it uses a learner that aims to identify *configurations* of objects or object parts; specifically, we use a structured SVM model that is updated online.

Whilst simultaneous tracking of multiple objects using model-based trackers has received significant attention in prior work, *e.g.*, in the context of tracking people [3], [4], [5], [39] and other objects [40], [41], up to the best of our knowledge, model-free tracking of multiple objects has hitherto remained unstudied. The most closely related to our work is a recent study [42] that tries to improve the identification of a single target object by also tracking stable features in the background, thereby improving the location prior for the target object. Our work differs from [42] in that it only tracks target objects and exploits the spatial relations between those; this makes it much easier to track, *e.g.*, multiple objects that have a very similar appearance, such as individual flowers in a large flower field or individual cars on a freeway.

The present paper is related to several prior studies that illustrate the potential of using contextual information to improve the performance of model-free trackers. In particular, [43], [44], [45], [46] automatically identify one or more auxiliary points or regions that are tracked in order to improve the location prior of the tracker. Auxiliary regions are generally defined as regions that are salient, that co-occur with the target object, and whose movement is correlated with that of the target object. They can be tracked, *e.g.*, using color-based mean-shift tracking [44], optical flow [46], or KLT tracking [45]. (In addition to auxiliary regions, [46] also employs "distractor" regions, which are similar in appearance to the target and consistently co-occur with the target, as context.) Our work differs from [44], [46] in that we adapt our model of the spatial relations between objects (or object parts) using an online-learning approach, which allows us to simultaneously track "target" and "auxiliary" objects for much longer periods of time. Our work also differs from prior work in that we do not make a distinction between "target" and "auxiliary" objects. Specifically, we use the same features and tracking-by-detection approach for both types of objects and the interaction between target and auxiliary objects is "undirected": the detection of auxiliary objects may help to identify the location of the target object, but vice versa, the detection of the target objects may also help to identify the location of the auxiliary object. This undirected interaction between target and auxiliary objects may lead to a more robust tracker.

In model-based tracking, various studies have also proposed the use of contextual information to improve the performance of the tracker. For instance, [47] makes assumptions on the environment (*viz.* it assumes the environment is a meeting room) in order to learn and exploit relations between different object categories (*e.g.*, humans and chairs). [48] propose an approach to increase the robustness of model-based trackers by incorporating constraints on the size of the target object(s), on the preponderance of the background, and on the smoothness of track trajectories. [49] exploit prior information on the location of target objects by learning classifier grids, *i.e.* by learning a separate detection classifier for each image location. Our study is different from [47], [48], [49] in that we take a model-free instead of a model-based approach: our tracker relies on just a single annotation of the target objects.

2.2 Deformable Template Models

Deformable template models represent an object by a collection of part models that are spatially related. Each part models the appearance of a portion of the (deformable) object, and the spatial relations between the parts are modeled by some joint distribution over part locations. Popular choices for this joint distribution over part locations include (1) PCA-based models that use a low-rank Gaussian model and (2) *pictorial-structures* models [14], which use a collection of springs between the parts (typically, using a tree structure). PCA-based models are used in, among others, active appearance models [50], active shape models [51], and constrained local models [52] that are commonly used in face or medical image analysis. Models based on pictorial structures are commonly used in tasks such as object detection [6], pose estimation [15], [53], and gesture recognition [54], [55]. In model-free tracking, there generally is insufficient training data to train PCA-based models, which is why we focus on pictorial-structures models in this work.

Pictorial-structures models represent spatial relations between object parts by springs that can be exerted or compressed (at a penalty) when the object under consideration is deformed. In many prior studies, pictorial-structures models are designed manually to incorporate the most important spatial relations between the object parts. For instance, [56], [57] represent the hierarchical structure of the human upper body using eight parts (*viz.* upper/lower arms, head, torso, and hands), and [53] further divide these upper body parts into smaller subparts in order to make the model more flexible. When designing models for the detection or tracking of arbitrary objects, however, such a manual approach to the design of pictorial-structures models is not viable. Therefore, detectors for arbitrary objects typically use generic structure models, such as (1) a star-shaped model that represents the object by a root node that is directly connect with all its parts [6] or (2) a Chow-Liu tree [58] that models only spatial relations between nearby parts [59]. In such detectors, the parts are automatically identified by searching for the most discriminative regions of the object based on a large collection of annotated images.

In this work, we also adopt generic star-shaped and minimum spanning tree models to represent the relations between the multiple objects that are being tracked. (In a second experiment, we also study model-free trackers that incorporate models for object parts that are heuristically determined.). The key difference with prior studies, however, is that we not learn the parameters of the springs in the pictorial-structures model (*i.e.* the length and direction of the springs) based on a large collection of annotated data, but that we learn the spring parameters in an online fashion during tracking [15].

2.3 Online Learning for Structured Prediction

Structured prediction techniques such as conditional random fields [60] and structured support vector machines [61] make predictions over (exponentially) large output spaces that have some inherent structure, such as sequences, trees, or objects configurations, and have been used with much success in a variety of computer vision problems [53], [62], [63]. The detection of multiple objects can naturally be framed as a structured-prediction problem of searching over all possible object configurations.

In our work, a structured predictor is trained in an online manner based on detections in earlier frames. Algorithms for online learning of structured predictors are generally based on perceptron-like learning algorithms [64], [65] or on stochastic (sub)gradient descent algorithms [66], [67]. Both learning algorithms perform cheap parameter updates based on a single training example or a small batch of examples, and work well in practice (in non-convex learning problems, they often construct better solutions than batch learners). We update our multi-object tracker after each frame using a simple (stochastic) subgradient descent algorithm *cf.* [61]; the step size for each parameter update is determined using a passive-aggressive algorithm [68].

3 SPOT TRACKER

The basis of our structure-preserving object tracker (SPOT) is formed by the popular Dalal-Triggs detector [11], which uses HOG features to describe image patches and an SVM to predict object presence. HOG features measure the magnitude and the (unsigned) direction of the image gradient in small cells (we used 8×8 pixel cells). Subsequently, contrast normalization is applied on rectangular, spatially connected blocks of four cells. The contrast normalization is implemented by normalizing the L2-norm of all histograms in a block. The resulting values are clipped at 0.2 and then renormalized according the L2-norm. The advantages of HOG features are that (1) they consider more edge orientations than just horizontal or vertical ones, (2) they pool over relatively small image regions, and (3) they are robust to changes in the illumination of the tracked object. Together, this makes HOG features more sensitive to the spatial location of the object than, *e.g.* Haar features [11], which is particularly important in model-free tracking because the identified location of the object is used to update the classifiers: small localization errors may thus propagate over time, causing the tracker to drift. Moreover, efficient implementations can extract HOG features at high frame rates.

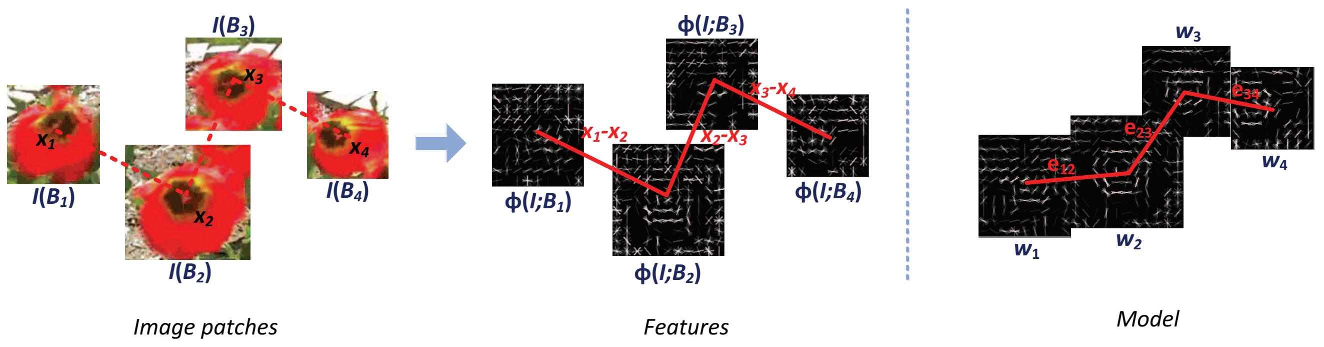

Fig. 1. The left image shows a configuration formed by some patches, where vertices and edges indicate objects and spatial relationships between objects. The middle image shows the features extracted from the left configuration, where vertices and edges indicate HOG features of objects and relative location vectors between objects. The right images shows the structure model we trained, where vertices and edges indicate trained HOG templates of objects and relative location vectors between objects. Our goal is to find the configuration match the trained structure model best in each frame, and update the structure model as well.

3.1 Model

We represent the bounding box that indicates object $i \in V$ (with V representing the set of objects we are tracking) by $B_i = \{\mathbf{x}_i, w_i, h_i\}$ with center location $\mathbf{x}_i = (x_i, y_i)$, width w_i, and height h_i. The HOG features extracted from image $\mathbf{I}$ that correspond to locations inside the object bounding box B_i are concatenated to obtain a feature vector $\phi(\mathbf{I}; B_i)$. Subsequently, we define a graph $G = (V, E)$ over all objects $i \in V$ that we want to track with edges $(i, j) \in E$ between the objects. The edges in the graph model can be viewed as springs that represent spatial constraints between the tracked objects. Next, we define the *score* of a *configuration* $C = \{B_1, \ldots, B_{|V|}\}$ of multiple tracked objects as the sum of two terms: (1) an appearance score that sums the similarities between the observed image features and the classifier weights for all objects and (2) a deformation score that measures how much a configuration compresses or stretches the springs between the tracked objects (the relative angles of the springs are not taken into account in our model). Mathematically, the score of a configuration C is defined as:

$$s(C; \mathbf{I}, \Theta) = \sum_{i \in V} \mathbf{w}_i^{\mathrm{T}} \phi(\mathbf{I}; B_i) - \lambda \sum_{(i,j) \in E} \|(\mathbf{x}_i - \mathbf{x}_j) - \mathbf{e}_{ij}\|^2. \quad (1)$$

Herein, the parameters $\mathbf{w}_i$ represent linear weights on the HOG features for object i, $\mathbf{e}_{ij}$ is the vector that represents the length and direction of the spring between objects i and j, and the set of all parameters is denoted by $\Theta = \{\mathbf{w}_1, \ldots, \mathbf{w}_{|V|}, \mathbf{e}_1, \ldots, \mathbf{e}_{|E|}\}$. We treat the parameter λ as a hyperparameter that determines the trade-off between the appearance and deformation scores. We use Platt scaling [69] to convert the configuration score to a configuration likelihood $p(C|\mathbf{I}; \Theta)$. Our model is illustrated in Figure 1. In preliminary experiments, we also tried to incorporate (temporal) Gaussian priors on the object locations in the score $s(C; \mathbf{I}, \Theta)$, but this did not appear to lead to performance improvements: the location prior specified by the spatial constraints in $s(C; \mathbf{I}, \Theta)$ appears to be sufficient.

3.2 Inference

Given the parameters of the model, finding the most likely object configuration amounts to maximizing Eqn. (1) over all possible configurations C. This maximization is intractable in general because it requires searching over exponentially many configurations, but for tree-structured graphs G, a combination of dynamic programming and min-convolutions can be used to perform the maximization in linear time. Here we only give briefly description of the inference algorithm; for more details, we refer to [6], [70].

The dynamic programming equations for maximizing Eqn. (1) take the form of a message-passing procedure on the tree G. Herein, a message from object i to j is computed by (1) gathering all incoming messages into object i, (2) combining the resulting score with the object detection scores for object i, and (3) transforming the result to incorporate for the stress in the edge between object i and j. The message-passing equations take the form:

$$R_{ij}(\mathbf{x}) = \mathbf{w}_i^{\mathrm{T}} \phi(\mathbf{I}; B_i) + \sum_{\forall k \neq j : (k,i) \in E} \mu_{k \to i}(\mathbf{x}), \quad (2)$$

$$\mu_{i \to j}(\mathbf{x}) = \max_{\mathbf{x}'} \left[R_{ij}(\mathbf{x}') - \lambda \|(\mathbf{x} - \mathbf{x}') - \mathbf{e}_{ij}\|^2 \right], \quad (3)$$

where $B_i = \{\mathbf{x}, w_i, h_i\}$. The message-passing procedure emanates from an arbitrary root object; after a full forward-backward pass on the pictorial-structures tree, the sum of all incoming messages into the root corresponds to the configuration C^* that maximizes $s(C; \mathbf{I}, \Theta)$. The configuration C^* can be recovered by backtracking the computations already performed. See [6] for more details.

The inference algorithm can be implemented very efficiently because the negate of Eqn. (3) is a two-dimensional *min-convolution* problem. The one-dimensional counterpart of Eqn. (3) can be rewritten in the form:

$$\mathcal{D}(p) = \min_q \left[f(q) + (p - q)^2 \right], \quad (4)$$

where $p, q \in \mathcal{G}$ with $\mathcal{G}$ a one-dimensional grid. The value of $\mathcal{D}(p)$ can be computed $\forall p \in \mathcal{G}$ in linear time by looping over the parabolas at all locations p, maintaining a list of coordinates for which the parabolas considered until now have the lowest value [70]. At each step in this loop, the intersection of the current parabola with the last parabola in the list is computed analytically. Based on the location of this intersection, the range of coordinates for which the current parabola has the lowest value can be determined, from which we can decide whether or not the last parabola in the list should be removed or not. The min-convolution algorithm is extended to two dimensions by first running the one-dimensional algorithm over all rows of the image, and then running the one-dimensional algorithm on the output of this first stage across all columns.

3.3 Learning

Like other model-free trackers [8], [9], [23], we use the tracked object configurations in previous frames as positive examples to update our model. After observing an image $\mathbf{I}$, the most likely object configuration C is found by maximizing Eqn. (1). We assume that this object configuration C is a true positive example, and therefore, we would like the score at this ground truth location $s(C; \mathbf{I}, \Theta)$ to be larger than any other configuration $\hat{C}$ by at least a margin $\Delta(C, \hat{C})$. Herein, Δ is a task-loss function that equals zero iff $\hat{C} = C$. The task loss is assumed to be non-negative, $\forall \hat{C} \neq C : \Delta(C, \hat{C}) > 0$, and upper bounded by a constant M, $\exists M \forall C : \max_{\hat{C}} \Delta(C, \hat{C}) < M$. To adapt our model in such a way that it assigns a higher score to the positive locations and lower scores to other locations, we update the model parameters Θ by taking a gradient step in the direction of the structured SVM loss function [61]. Here, the structured SVM loss ℓ is defined as the maximum violation of the task loss by configuration $\hat{C}$:

$$\ell(\Theta; \mathbf{I}, C) = \max_{\hat{C}} \left[s(\hat{C}; \mathbf{I}, \Theta) - s(C; \mathbf{I}, \Theta) + \Delta(C, \hat{C}) \right]. \tag{5}$$

The structured SVM loss function does not contain quadratic terms, but it is the maximum of a set of affine functions. As a result, the structured SVM loss in Eqn. (5) is a convex function in the parameters Θ.

In our SPOT tracker, we define the task-loss $\Delta(C, \hat{C})$ based on the amount of overlap between the correct configuration C and the incorrect configuration $\hat{C}$:

$$\Delta(C, \hat{C}) = \sum_{i \in V} \left(1 - \frac{B_i \cap \hat{B}_i}{B_i \cup \hat{B}_i} \right). \tag{6}$$

Herein, the union and intersection of two bounding boxes B_i and $\hat{B}_i$ are both measured in pixels. If $\hat{C}$ has no overlap with C, the task loss equals $|V|$, and the task loss equals zero iff $\hat{C} = C$. In preliminary experiments, we also explored task losses that only considered the distance between object centers (and not the size of objects), but we did not find these to improve the performance of our tracker.

The gradient of the structured SVM loss in Eqn. (5) with respect to the parameters Θ is given by:

$$\nabla_{\Theta} \ell(\Theta; \mathbf{I}, C) = \nabla_{\Theta} s(C^*; \mathbf{I}, \Theta) - \nabla_{\Theta} s(C; \mathbf{I}, \Theta). \tag{7}$$

in which the "negative" configuration C^* is given by:

$$C^* = \operatorname*{argmax}_{\hat{C}} \left(s(\hat{C}; \mathbf{I}, \Theta) + \Delta(C, \hat{C}) \right). \tag{8}$$

In practice, this gradient is not very appropriate for learning the parameters of our multi-object tracker because due to the influence of the deformation score, the negative configuration C^* tends to comprise bounding boxes B_i that have very low appearance scores $\mathbf{w}_i^{\mathrm{T}} \phi(\mathbf{I}; B_i)$. As a result, these bounding boxes are inappropriate examples to use as a basis for updating $\mathbf{w}_i$. For instance, if we would incorporate the deformation score when identifying a negative example in the *Air Show* video (see Fig. 2), we would often select empty sky regions as negative examples. These regions are very uninformative, which hampers the learning of good appearance models for the airplanes under consideration. To address this problem, we ignore the structural constraints when selecting the negative configuration. Specifically, we use a search direction $\mathbf{p}$ for learning that is defined as:

$$\mathbf{p} = \nabla_{\Theta} \tilde{s}(\tilde{C}^*; \mathbf{I}, \Theta) - \nabla_{\Theta} s(C; \mathbf{I}, \Theta), \tag{9}$$

where the negative configuration is given by:

$$\tilde{C}^* = \operatorname*{argmax}_{\hat{C}} \left(\tilde{s}(\hat{C}; \mathbf{I}, \Theta) + \Delta(C, \hat{C}) \right), \tag{10}$$

with the negative appearance score $\tilde{s}$ being defined as:

$$\tilde{s}(\tilde{C}^*; \mathbf{I}, \Theta) = \sum_{i \in V} \mathbf{w}_i^{\mathrm{T}} \phi(\mathbf{I}; B_i). \tag{11}$$

It is straightforward to show that the inner product between the search direction and the true gradient direction remains positive, $\mathbf{p} \cdot \nabla_{\Theta} \ell(\Theta; \mathbf{I}, C) > 0$, as a result of which the learner will still converge (under suitable conditions on the step size [71]). The configuration $\tilde{C}^*$ can be computed efficiently by (1) adding a term to each object filter response that contains the ratio of overlapping pixels for a bounding box at that location with the detected bounding box to account for the task loss Δ and (2) re-running exactly the same efficient inference procedure as the one that was used to maximize Eqn. (1) over configurations.

We use a passive-aggressive algorithm to perform the parameter update [68]. The passive-aggressive algorithm sets the step size in such a way as to substantially decrease the loss, while ensuring that the parameter update is not too large. In particular, the passive-aggressive algorithm uses the following parameter update:

$$\Theta \leftarrow \Theta - \frac{\ell(\Theta; \mathbf{I}, C)}{\|\mathbf{p}\|^2 + \frac{1}{2K}} \mathbf{p}, \tag{12}$$

where $K \in (0, +\infty)$ is a hyperparameter that controls the "aggressiveness" of the parameter update. The selection of K will be discussed in Section 4.1. In preliminary experiments, we also experimented with a confidence-weighted learning algorithm [72] that finds an individual step size for each parameter. However, the performance of this confidence-weighted learning algorithm was very similar to that of the passive-aggressive algorithm, which is why we use the passive-aggressive algorithm in the remainder of the paper.

When an object is occluded, we do not want to update the appearance model for that object (even if its location was correctly predicted thanks to the spatial constraints in our model). To avoid erroneous updates of our appearance model, we only update a weight vector $\mathbf{w}_i$ corresponding to detection B_i when the exponentiated score for that object exceeds some threshold. In particular, we only update the $\mathbf{w}_i$ and $\mathbf{e}_{ij}$ whenever $\frac{1}{Z}\exp(\mathbf{w}_i^{\mathrm{T}}\phi(\mathbf{I};B_i)) > 0.4$.

The weights $\mathbf{w}_i$ are initialized by training an SVM that discriminates between the manually annotated object from 50 randomly selected negative examples (extracted from the first frame) that have little to no overlap with the ground-truth annotation. The parameters $\mathbf{e}_{ij}$ are initialized based on the ground-truth annotations of the objects: $\mathbf{e}_{ij} \leftarrow \mathbf{x}_i - \mathbf{x}_j$, with $\mathbf{x}_i$ and $\mathbf{x}_j$ the locations of objects i and j in the first frame.

3.4 Graph Structure

A remaining issue is how we determine the structure of the graph G, *i.e.* how we decide on which objects are connected by an edge. Ideally, we would employ a fully connected graph G, but this would make inference intractable (see 3.2). Hence, we explore two approaches to construct a tree on the objects $i \in V$: (1) a star model [6] and (2) a minimum spanning tree model. In the star model, each object is connected by an edge with a dummy object $r \in V$ that is always located at the center of all the objects, *i.e.* there are no direct relations between the objects. This requires a minor adaptation of the score function:

$$s(C;\mathbf{I},\Theta) = \sum_{i \in V/r} \mathbf{w}_i^{\mathrm{T}}\phi(\mathbf{I};B_i) - \lambda \sum_{(i,r)\in E} \|(\mathbf{x}_i - \mathbf{x}_r) - \mathbf{e}_i\|^2. \quad (13)$$

The minimum spanning tree model is constructed based on the object annotations in the first frame; it is obtained by searching the set of all possible completely-connected tree models for the tree that minimizes $\sum_{(i,j)\in E}\|\mathbf{x}_i - \mathbf{x}_j\|^2$, where $\mathbf{x}_i$ and $\mathbf{x}_j$ are the locations of objects i and j in the first frame.

3.5 Computational Complexity

The main computational costs of running our tracker are in the extraction of the HOG features (which are shared between object detectors) and in the computation of the appearance score per object; after these quantities are computed, the maximization of Eqn. (1) takes only a few milliseconds. The computational complexity grows linearly in the number of objects being tracked (*i.e.*, in $|V|$). We found that it is possible to track approximately four objects simultaneously in real-time (in small videos of size 320×240 pixels) on a consumer laptop.

4 Experiments

We performed three sets of experiments to evaluate the performance of our tracker. In the first set of experiments, we evaluate the performance of the SPOT tracker on a range of multi-object tracking problems, comparing it to the performance of various state-of-the-art trackers that do not employ structural constraints between the objects. In the second set of experiments, we study the use of SPOT to improve single-object tracking by tracking parts of an object and constraining the spatial configuration of those parts. The third and last experiment studies the use of SPOT for adapting generic object detectors to detectors for specific objects during tracking.

4.1 Experiment 1: Multiple-Object Tracking

We first evaluate the performance of the SPOT tracker on videos in which multiple objects need to be tracked.

4.1.1 Setup

We used nine videos with multiple objects in this set of experiments. Three of these videos (*Shaking*, *Basketball*, and *Skating*) were already used in earlier studies [73]; the other six were downloaded from YouTube. The videos were selected based on characteristics that are challenging for current model-free trackers, such as the presence of multiple, nearby objects with a very similar appearance. The average length of the videos is 842 frames. The left column of Figure 2 shows the first frame of each of the nine videos along with the corresponding ground-truth annotations of the objects, *i.e.* the left column of Figure 2 shows all labeled training data that is available to train our tracker. The videos are available from http://visionlab.tudelft.nl/spot.

We experiment with three variants of the SPOT tracker: (1) a baseline tracker that does not use structural constraints (*i.e.* a SPOT tracker with $\lambda = 0$; no-SPOT), (2) a SPOT tracker that uses a star tree to model spatial relations between objects (star-SPOT), and (3) a SPOT tracker that uses a minimum spanning tree to model structural constraints (mst-SPOT). We compare the performance of our SPOT trackers with that of two state-of-the-art (single-object) trackers, *viz.* the OAB tracker [23] and the TLD tracker [9], of which we run multiple instances to separately track each object. The OAB and TLD trackers were run using the implementations provided by their developers.

Following [8], we evaluate the performance of the trackers by measuring[1]: (1) *average location error* (ALE): the average distance of the center of the identified bounding box to the center of the ground-truth bounding box and (2) *correct detection rate* (CDR): the percentage of frames for which the overlap between the identified bounding box and the ground truth bounding box is at least 50 percent. For each video, these two measurements are averaged

1. Unlike [9], we do not compute the recall (or F-measure) of the trackers: the recall of a tracker is only informative when in the target object is occluded or has disappeared in a substantial portion of the video. In the videos we considered in our experiments, there are no such occlusions or disappearances of the target object.

over all target objects, and over five separate runs. In all experiments with star-SPOT and mst-SPOT, we fixed $\lambda = 0.001$ and $K = 1$. In preliminary experiments, we found that the results are very robust under changes of λ and K.

4.1.2 Results

The performance of the five trackers (OAB, TLD, no-SPOT, star-SPOT, and mst-SPOT) on all nine videos is presented in Table 1. Figure 2 shows five frames of all videos with the tracks obtained by mst-SPOT. Videos showing the complete tracks are provided in the supplemental material. Videos showing all tracking results are available on http://visionlab.tudelft.nl/spot. We first qualitatively describe the results on four of the videos.

Air Show. The video contains a formation of four visually similar planes that fly very close to each other; the video contains a lot of camera shakes. Whereas our baseline trackers (OAB, TLD, and no-SPOT) confuse the planes several times during the course of the video, star-SPOT and mst-SPOT track the correct plane throughout the entire video.

Car Chase. This video is challenging because (1) the two cars are very small and (2) both cars are occluded for around 40 frames, while various other cars are still visible. Whereas this occlusion confuses the baseline trackers, the two SPOT trackers do not lose track because they can use the location of one car to estimate the location of the other.

Red Flowers. The video shows several similar flowers that are moving and changing appearance due to the wind, and that sometimes (partially) occlude each other; we track four of these flowers. The baseline trackers lose track very quickly due these partial occlusions. By contrast, the two SPOT trackers flawlessly track all flowers during the entire length of the video (2249 frames), because they can use the structural constraints to distinguish the different flowers.

Hunting. The cheetah and gazelle in this video clip are very hard to track, because their appearance changes significantly over time and because their relative location is changing (the cheetah passes the gazelle). Nevertheless, the SPOT trackers can exploit the fact that both animals move in a similar direction, which prevents them from losing track.

Taken together, the results presented in Table 1 and Figure 2 show (1) that our baseline no-SPOT tracker performs on par with state-of-the-art trackers such as TLD; and (2) that the use of spatial constraints between objects leads to substantial performance improvements when tracking multiple objects, in particular, when minimum spanning trees are used (mst-SPOT). The performance improvements are particularly impressive for videos in which objects with a similar appearance are tracked, such as the *Car Chase* and *Red Flowers* videos, because the structural constraints prevent the tracker from switching between objects. Structural constraints are also very helpful in videos with a lot of camera shake (such as the *Air Show* video), because camera shake causes all objects to move in the same direction in the image. The SPOT tracker even outperforms single-object trackers when perceptually different objects are tracked

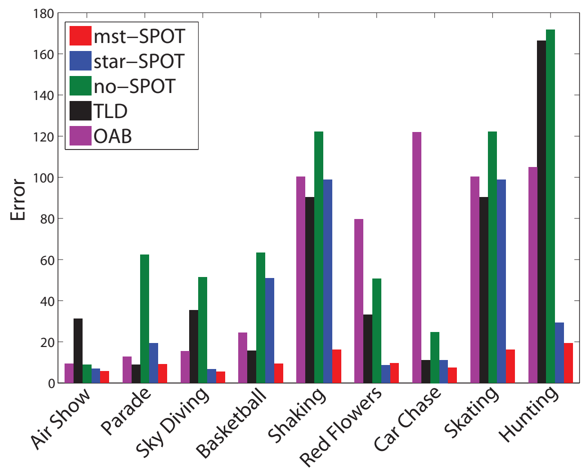

Fig. 3. Average location error of five multiple-object trackers on all nine videos (lower is better). Figure best viewed in color.

that have a relatively weak relation in terms of their location, such as in the *Hunting* video, because it can share information between objects to deal with, *e.g.*, motion blur and camera shake. The mst-SPOT tracker outperforms the star-SPOT tracker in almost all videos, presumably, because a minimum spanning tree introduces direct (rather than indirect) constraints between the locations of the objects.

We also performed experiments with multi-scale versions of our SPOT tracker. The multiple-scale trackers are run at three scales for each frame (viz. relative scales 0.9, 1.0, and 1.1), and select the highest posterior probability among the three scales to determine the location and scale of the object. In all experiments, we assume that the aspect ratio of the rectangular bounding box is fixed. Table 2 presents the performance obtained by three multi-scale, multi-object trackers on the same nine movies. In particular, we use the (multi-scale) CXT tracker [46], a multi-scale version of the TLD tracker [9], and a multi-scale version of the no-SPOT tracker as a baseline[2]. We compare the performance of these baseline trackers with that of our multi-scale mst-SPOT tracker. (Following [8], we only measure tracking errors in terms of the average location error in these experiments. Measuring the correct detection rate is problematic because the ground truth bounding boxes are all in single scale.) The results presented in Table 2 are in line with those presented in Table 1: the mst-SPOT tracker substantially outperforms trackers without structural constraints on most (but not all) of the videos. On videos such as the *Parade* and *Sky Diving*, the CXT and TLD trackers appear to outperform mst-SPOT despite their lack of structural constraints. We surmise this has to do with the size of the objects: the fixed-size of HOG cells (of 8×8 pixels) that SPOT uses in its appearance model may be too large for the small target objects in these two videos. Indeed, the Haar-representations that the CXT

2. In the experiments with multi-scale trackers, we did not consider the OAB tracker because it cannot straightforwardly be adapted from single-scale to multi-scale mode.

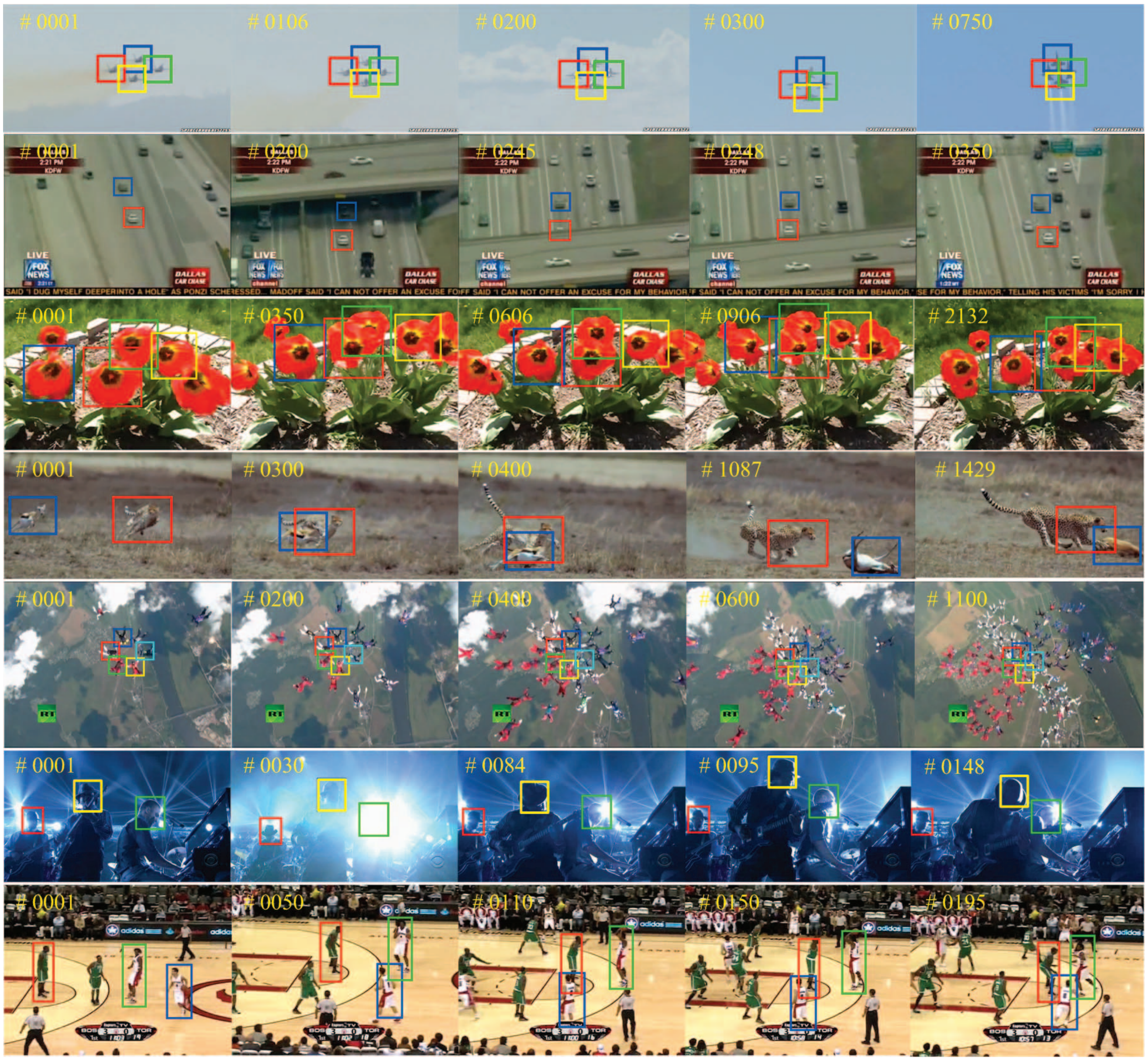

Fig. 2. Tracking results obtained by mst-SPOT on all nine videos used in Experiment 1 (from top to bottom: *Air Show*, *Car Chase*, *Red Flowers*, *Hunting*, *Sky Diving*, *Shaking*, and *Basketball*). The colors of the rectangles indicate the different objects that are tracked. Figure best viewed in color. Videos showing the full tracks are presented in the supplementary material.

and TLD tracker employ may be more suited for tracking such small objects. Comparing the results in Table 1 and 2, it can be seen that the difference in performance between single-scale and multi-scale SPOT trackers is rather small, presumably, because most videos exhibit relatively little scale changes over time.

4.2 Experiment 2: Single Object Tracking

With some minor modifications, our SPOT tracker may also be used to improve the tracking of single objects. As the appearance of an object may frequently change due to partially occlusions, a holistic appearance model may easily loose track. The appearance of (some of the) smaller parts of the object is not altered by partial occlusions; therefore, separately tracking parts may help to locate the object of interest. SPOT can be used to track the parts of a single object (treating them as individual objects in V) with structural constraints between the parts. Inspired by [6], we experiment with a SPOT tracker that has a single "global" object detector and a number of "local" part detectors. We experiment with a star-shaped model in which the global detector forms the root of the star (star-SPOT), and with a model that constructs a minimum spanning tree over the global object and the local part detectors (mst-SPOT). In single-object tracking, the alteration we made earlier to the search direction of the learner has become unneeded:

TABLE 1
Performance of five model-free trackers on multiple-object videos measured in terms of (1) average location error (ALE; lower is better) in pixels between the centers of the predicted and the ground-truth bounding box and (2) correct detection rate (CDR; higher is better). To measure the correct detection rate, a detection is considered correct if the overlap between the identified bounding box and the ground truth bounding box is at least 50%. The results are averaged over five runs and over all target objects in each video. The best performance on each video is boldfaced.

	OAB [23]		**TLD** [9]		**no-SPOT**		**star-SPOT**		**mst-SPOT**	
	ALE	*CDR*	*ALE*	*CDR*	*ALE*	*CDR*	*ALE*	*CDR*	*ALE*	*CDR*
Air Show	9.3	0.86	31.3	0.53	8.8	0.92	6.9	0.92	**5.6**	**1.00**
Car Chase	121.8	0.57	11.2	0.76	24.8	0.78	11.2	0.82	**7.5**	**0.91**
Parade	12.7	**0.82**	**8.8**	0.71	62.3	0.29	19.4	0.35	9.2	0.63
Red Flowers	79.7	0.09	33.3	0.30	50.6	0.38	**8.6**	0.98	9.5	**0.99**
Hunting	104.9	0.25	166.4	0.08	171.7	0.07	29.2	0.72	**19.4**	**0.87**
Sky Diving	15.5	0.76	35.3	0.13	51.4	0.48	6.73	0.98	**5.4**	**1.00**
Shaking	61.9	0.47	14.3	0.47	58.3	0.47	28.7	0.38	**7.7**	**0.97**
Basketball	24.4	0.63	15.6	0.67	63.3	0.67	50.9	0.54	**9.4**	**1.00**
Skating	100.2	0.05	90.3	0.42	122.2	0.35	98.9	0.27	**16.2**	**0.85**
Avg. rank	3.9	3.6	2.9	3.3	4.4	3.2	2.6	3.1	**1.2**	**1.2**

TABLE 2
Performance of four multi-scale model-free trackers on multiple-object videos measured in terms of the average location error (ALE; lower is better) in pixels between centers of the predicted and the ground-truth bounding box. The results are averaged over five runs and over all target objects in each video. The best performance on each video is boldfaced.

	CXT [46]	**TLD** [9]	**no-SPOT**	**mst-SPOT**
	ALE	*ALE*	*ALE*	*ALE*
Air Show	**6.5**	21.3	11.6	7.6
Car Chase	66.1	22.4	69.2	**4.1**
Parade	**8.0**	8.8	19.8	15.7
Red Flowers	59.7	40.2	72.6	**13.7**
Hunting	20.5	133.5	125.9	**17.6**
Sky Diving	6.4	**5.8**	19.5	9.8
Shaking	107.5	14.3	30.9	**9.9**
Basketball	54.1	15.6	34.5	**9.4**
Skating	130.7	90.3	88.4	**27.3**
Avg. rank	2.7	2.4	3.3	**1.6**

because all part detectors involve the same object, it is actually important to incorporate the structural constraints when identifying a "negative" configuration. Therefore, we used SPOT trackers that learn using the true gradient of the structured SVM loss in our single-object tracking experiments, *i.e.* we set $\mathbf{p} = \nabla_\Theta \ell(\Theta; \mathbf{I}, C)$.

4.2.1 Setup

Because a single bounding box is used to annotate the object in the first frame of the video, we need to determine what parts the tracker will use. As a latent SVM [6] is unlikely to work well on a single training example, we use a heuristic approach that assumes that relevant parts correspond to discriminative regions inside the object bounding box. We initialize part i at the location in which the weights of the initial global SVM $\mathbf{w}$ are large and positive, because these correspond to features that are highly indicative of object presence. In particular, we initialize B_i as:

$$B_i = \operatorname*{argmax}_{B_i' \subset B} \sum_{(x_i', y_i') \in B_i'} \big(\max(0, w_{x_i' y_i'})\big)^2, \tag{14}$$

where B denotes the bounding box of the (single global) object. We fix the number of parts $|V|-1$ in advance, setting it to 2; we fix the width and height of the part bounding boxes to 40% of the width and height of the bounding box indicating the deformable object; and we ensure that the selected part cannot have more than 50% overlap with the other parts in the first frame. Unlike [6], we extract the features for the part detectors on the same scale as the features for the global detector. In preliminary experiments, we also tried using finer-scale HOG features to describe the parts, but we did not find this to lead to performance improvements. Using the same features for all detectors has computational advantages because the features only need to be computed once, which is why we opt for it here.

The experiments are performed on a publicly available collection of thirteen videos. The first twelve videos of them are from [8], and the last one is from [9]. The videos contain a wide range of objects that are subject to sudden movements and (out-of-plane) rotations, and have cluttered, dynamic backgrounds. The videos have an average length of 718 frames. Each video contains a single object to be tracked, which is indicated by a bounding box in the first frame of the video. (First-frame annotations for all movies are shown in [8].)

Again, we evaluate the performance of the trackers by measuring the average location error (ALE) and the correct detection rate (CDR) of the tracker, and averaging over five runs. We compare the performance of our tracker with that of three state-of-the-art trackers, viz., the OAB tracker [23], the MILBoost tracker [8], and the TLD tracker [9]. (All trackers were run on a single scale.) We could not run the implementation of the MILBoost tracker ourselves as it is outdated (the MILBoost tracker was not considered in Experiment 1 for this reason), but because we use exactly the same experimental setup as [8], we adopt the results

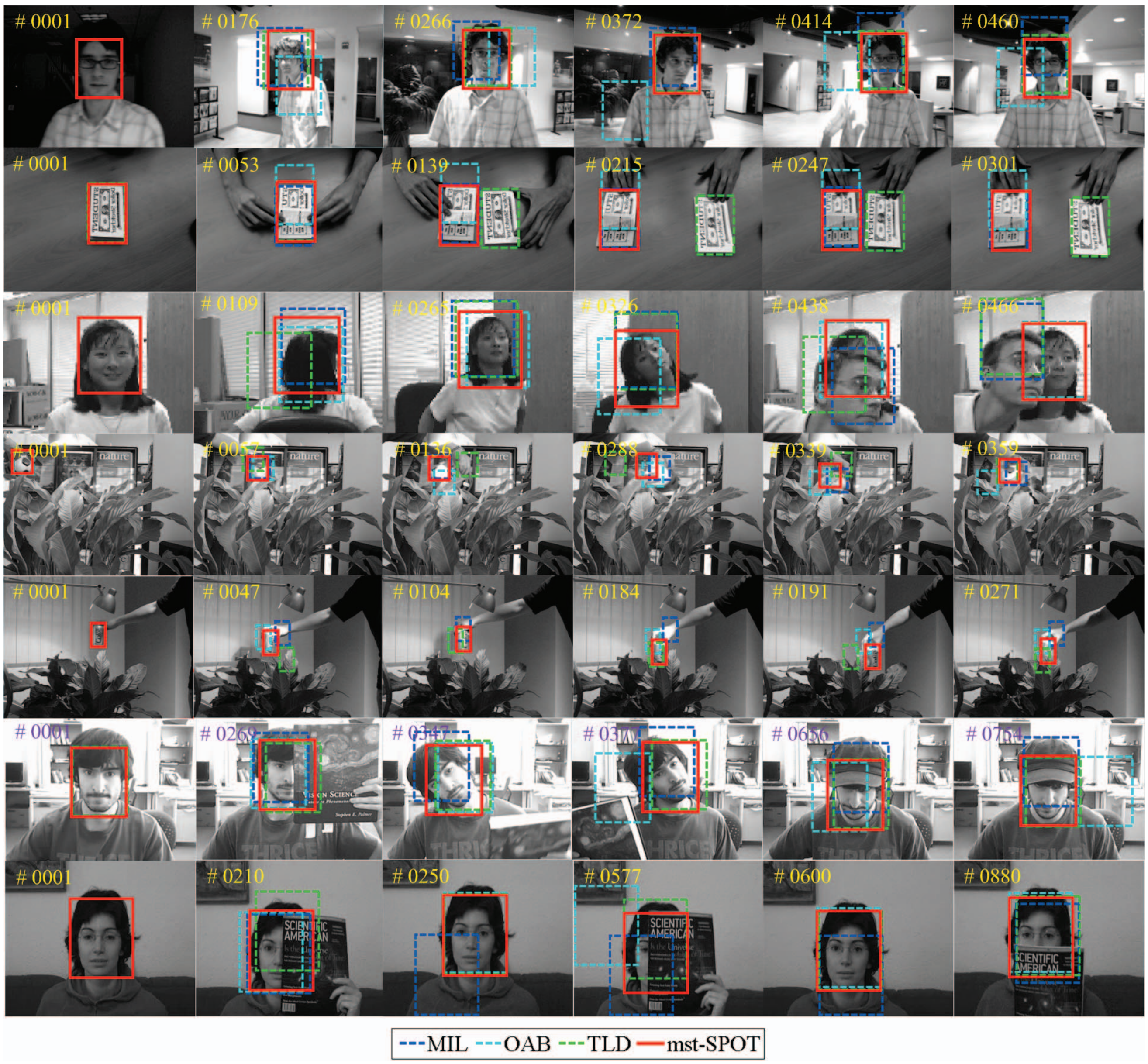

Fig. 4. Tracking results on seven of the thirteen videos (*David Indoor*, *Dollar*, *Girl*, *Tiger 2*, *Coke Can*, *Occl. Face 2*, and *Occl. Face*) obtained by the MIL, OAB, TLD, and mst-SPOT trackers. Figure best viewed in color.

presented in [8] here.

4.2.2 Results

Table 3 and Figure 5 present the performance of all six trackers on all thirteen videos. Figure 4 illustrates the tracks obtained with the MIL, TLD, and mst-SPOT trackers on seven of the thirteen videos. The results reveal the potential benefit of using additional part detectors when tracking a single object: mst-SPOT is the best-performing tracker on eight of the thirteen videos in terms of average location error, and on nine of the thirteen videos in terms of correct detection rate. The performance improvements are particularly impressive in challenging movies such as the *Tiger* videos, in which parts of the object are frequently occluded by leaves. In such situations, the SPOT trackers benefit from the presence of part detectors that can accurately detect the non-occluded part(s) of the object. The results also show that mst-SPOT generally outperforms star-SPOT, which is interesting because it suggests that for object detection in still images, pictorial-structure models with a minimum spanning tree [59] may be better than those with a star tree [6]. Whilst mst-SPOT outperforms the other tracker on most videos, its performance on the *Motocross* and *Tea Box* videos is poor. For the *Motocross* video, we surmise that this is because our HOG descriptors are not appropriate for the small target region being tracked in that video. The results on the *Tea Box* video suggest that single-object SPOT —like most other model-free trackers— may be hampered by out-of-plane rotations of the target object.

TABLE 3
Performance of six model-free trackers on single-object videos measured in terms of (1) average location error (ALE; lower is better) in pixels between the centers of the predicted and the ground-truth bounding box and (2) correct detection rate (CDR; higher is better). To measure the correct detection rate, a detection is counted as correct if the overlap between the identified bounding box and the ground truth bounding box is at least 50%. The results are averaged over five runs. The best performance on each video is boldfaced.

	OAB [23]		**MIL** [8]		**TLD** [9]		**no-SPOT**		**star-SPOT**		**mst-SPOT**	
	ALE	*CDR*	*ALE*	*CDR*	*ALE*	*CDR*	*ALE*	*CDR*	*ALE*	*CDR*	*ALE*	*CDR*
Sylvester	20.1	0.42	10.9	0.73	20.0	0.91	9.6	0.88	9.3	0.90	**7.1**	**0.93**
David	45.0	0.34	22.9	0.61	4.5	**1.00**	4.3	**1.00**	4.5	**1.00**	**3.5**	**1.00**
Cola Can	11.2	0.37	20.9	0.22	16.3	0.52	28.5	0.27	21.4	0.37	**7.1**	**0.75**
Occl. Face 1	17.9	0.92	27.2	0.78	16.8	0.99	5.7	**1.00**	5.5	**1.00**	**4.6**	**1.00**
Occl. Face 2	22.5	0.85	20.2	0.82	22.1	0.77	9.7	0.99	12.1	0.85	**7.4**	**1.00**
Surfer	23.7	0.61	9.2	0.76	**7.9**	**0.84**	9.8	0.46	89.2	0.26	13.4	0.43
Tiger 1	43.1	0.25	15.3	0.58	28.7	0.13	7.8	**0.90**	22.1	0.37	**6.1**	0.89
Tiger 2	21.6	0.44	17.1	0.64	37.5	0.27	25.9	0.42	26.5	0.39	**7.6**	**0.88**
Dollar	24.7	0.79	14.8	0.95	3.9	**1.00**	**3.8**	**1.00**	4.5	**1.00**	5.5	**1.00**
Cliff Bar	33.2	0.67	**11.6**	0.77	12.3	0.36	36.3	0.42	67.6	0.35	12.1	**0.79**
Tea Box	**8.6**	**0.94**	10.2	0.86	39.0	0.18	15.8	0.74	28.6	0.43	41.9	0.40
Girl	13.5	0.97	32.0	0.57	24.7	0.78	14.7	0.97	10.5	**1.00**	**10.4**	**1.00**
Motocross	27.3	0.33	81.9	0.05	**7.5**	**0.57**	58.6	0.07	40.3	0.14	34.8	0.22
Avg. rank	4.1	3.8	3.9	4.2	3.8	3.6	3.3	3.0	3.6	3.2	**2.1**	**1.9**

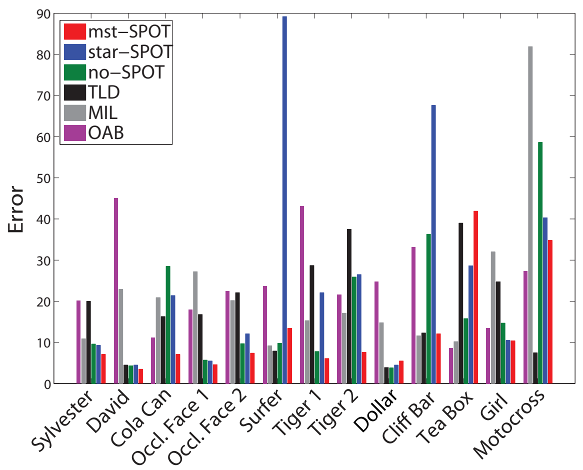

Fig. 5. Average location error of six single-object trackers on all thirteen videos (lower is better). Figure best viewed in color.

4.3 Experiment 3: Tracking and Identification

In the third and last experiment, we explore the merits of the SPOT tracker in the context of model-based trackers that follow the tracking-by-detection paradigm. In particular, we explore the ability of SPOT to improve the performance of object detectors that are trained to recognize generic objects classes such as faces, pedestrians, or cars. After an initial detection is made using the Felzenszwalb object detector [6], we may use single-object SPOT to track the detected object over time. In our experiments, the SPOT tracker is initialized using the parameters of the off-the-shelf detector; after the initial detection, the parameters of the tracker are then adapted to recognize the particular object under consideration.

4.3.1 Setup

We perform experiments on three videos for pedestrian detection from ETH data set [74], called *Sunny day*, *Jelmoli*, and *Bahnhof*. The *Sunny day* movie has a length of 354 frames and contains a total of 29 persons, the *Jelmoli* movie has 450 frames and about 40 persons, and the *Bahnhof* movie has 1000 frames and about 70 persons. To avoid tracking false positive detections, we only select those detections which have high response at the same location longer than 3 frames.

We compare our tracker with that of three state-of-the-art pedestrian detectors that are used in a tracking-by-detection scenario: (1) a Dalal-Triggs detector (HOG) that was obtained by training a linear SVM on HOG features [11], (2) a Felzenszwalb detector (LatSVM) that was obtained by training a latent SVM on HOG features at two scales [6], and (3) the "fastest pedestrian detector in the wild" (FPDW) that was obtained by Dollár [75]. All three detectors were trained on the pedestrian class of the Pascal VOC 2007 challenge. We ran the three detectors using their default parameter values. Following [76], we evaluate the performance of all pedestrian detectors by evaluating the miss rate as a function of the false positive rate.

4.3.2 Results

Figure 6 shows the miss rate as a function of the false positive rate of all four detectors on all three videos (lower curves indicate better performance). The results show that SPOT consistently outperforms the baseline LatSVM detector, which is identical to SPOT without the online-learning component. This shows that tailoring the appearance model of an object detector to a particular object of interest may indeed improve performance. Both LatSVM and SPOT substantially outperform the Dalal-Triggs detector (HOG), which highlights the merits of using part detectors in addition to a global object detector. SPOT is also very competitive compared to FPDW, in particular, in scenarios in which low false positive rates are required.

Figure 7 shows some of the tracking results of mst-SPOT for on the *Sunny day*, *Jelmoli*, and *Bahnhof* videos. The results highlight the strong performance of mst-SPOT for pedestrian detection. In particular, mst-SPOT appears to perform well when multiple pedestrians partially occlude each other thanks to the use of part detectors.

5 Conclusion and Future Work

In this paper, we have developed a new model-free tracker that tracks multiple objects by combining multiple single-object trackers via constraints on the spatial structure of the objects. Our experimental results show that the resulting SPOT tracker substantially outperforms traditional trackers in settings in which multiple objects need to be tracked. We have also showed that the SPOT tracker can improve the tracking of single objects by including additional detectors for object parts. Moreover, the SPOT tracker can be used to adapt a generic detector to a specific object while tracking. The computational costs of our tracker only grow linearly in the number of objects (or object parts) that is being tracked, which facilitates real-time tracking. Of course, the ideas presented in this paper may readily be implemented in other model-free trackers that are based on tracking-by-detection, such as the TLD tracker [9]. It is likely that including structural constraints in those trackers will lead to improved tracking performance, too.

In future work, we aim to explore the use of different structural constraints between the tracked objects; for instance, for tracking certain deformable objects it may be better to use a structural model based on PCA (as is done in, *e.g.*, constrained local models [27]). We also plan to develop approaches that identify the relevant parts of a deformable object in a more principled way during tracking via online learning algorithms for latent SVMs.

Acknowledgments

This work was supported by EU-FP7 Social Signal Processing (SSPNet) and by the China Scholarship Council. The authors thank two anonymous reviewers for helpful comments on earlier versions of this paper. In addition, we thank David Tax, Marco Loog, and Martijn van de Giessen for many helpful discussions, and Zdenek Kalal and Helmut Grabner for their implementations of the TLD and OAB trackers.

References

[1] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2001, pp. 511–518.

[2] M. Isard and J. Maccormick, "Bramble: A Bayesian multi-blob tracker," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2001, pp. 34–41.

[3] M. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool, "Online multiperson tracking-by-detection from a single, uncalibrated camera," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 33, no. 9, pp. 1820–1833, 2011.

[4] S. Pellegrini, A. Ess, K. Schindler, and L. van Gool, "You'll never walk alone: Modeling social behavior for multi-target tracking," in *Proc. IEEE Intl Conf. Computer Vision*, 2009, pp. 261–268.

[5] Z. Wu, A. Thangali, S. Sclaroff, and M. Betke, "Coupling detection and data association for multiple object tracking," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2012, pp. 1948–1955.

[6] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part based models," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1627–1645, 2010.

[7] V. Lepetit and P. Fua, "Keypoint recognition using randomized trees," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 28, no. 9, pp. 1456–1479, 2006.

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

[9] Z. Kalal, J. Matas, and K. Mikolajczyk, "P-n learning: Bootstrapping binary classifiers by structural constraints," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2010, pp. 49–56.

[10] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Computing Surveys*, vol. 38, no. 4, 2006.

[11] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2005, pp. 886–893.

[12] Z. Kalal, K. Mikolajczyk, and J. Matas, "Face-TLD: Tracking-Learning-Detection Applied to Faces," *Proc. IEEE Conf. International Conference on Image Processing*, 2010.

[13] C. Caraffi, T. Vojir, J. Trefny, J. Sochman, and J. Matas, "A System for Real-time Detection and Tracking of Vehicles from a Single Car-mounted Camera," in *ITS Conference*, Sep. 2012, pp. 975–982.

[14] M. Fischler and R. Elschlager, "The representation and matching of pictorial structures," *IEEE Transactions on Computers*, vol. 22, no. 1, pp. 67–92, 1973.

[15] S. Branson, P. Perona, and S. Belongie, "Strong supervison from weak annotation: Interactive training of deformable part models," in *Proc. IEEE Int. Conf. Computer Vision*, 2011, pp. 1832–1839.

[16] L. Zhang and L. van der Maaten, "Structure preserving object tracking," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2013.

[17] H. Yang, L. Shao, F. Zheng, L. Wang, and Z. Song, "Recent advances and trends in visual tracking: A review," *Neurocomputing*, vol. 74, no. 18, pp. 3823–3831, 2011.

[18] Y. Wu, J. Lim, and M.-H. Yang, "Online object tracking: A benchmark," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2013.

[19] A. Balan and M. Black, "An adaptive appearance model approach for model-based articulated object tracking," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 1, 2006, pp. 758–765.

[20] M. J. Black and A. D. Jepson, "Eigentracking: Robust matching and tracking of articulated objects using a view-based representation," *Intl J. Computer Vision*, vol. 26, no. 1, pp. 63–84, 1998.

[21] D. Ross, J. Lim, R.-S. Lin, and M.-H. Yuang, "Incremental learning for robust visual tracking," *Intl J. Computer Vision*, vol. 77, no. 1, pp. 125–141, 2008.

[22] S. Avidan, "Ensemble tracking," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 29, no. 2, pp. 261 –271, 2007.

[23] H. Grabner, M. Grabner, and H. Bischof, "Real-time tracking via on-line boosting," in *British Machine Vision Conference*, 2006, pp. 47–56.

[24] R. Lin, D. Ross, J. Lim, and M. H. Yang, "Adaptive discriminative generative model and its applications," in *Proc. Neural Information Processing Systems*, 2004, pp. 801–808.

[25] X. Liu and T. Yu, "Gradient feature selection for online boosting," in *Proc. IEEE Intl Conf. Computer Vision*, 2007, pp. 1–8.

[26] S. Avidan, "Support vector tracking," *IEEE Trans. Pattern Analysis and Machine Intelligence*, pp. 1064 –1072, 2004.

[27] D. Cristinacce and T. F. Cootes, "Automatic feature localisation with constrained local models," *Pattern Recognition*, vol. 41, no. 10, pp. 3054–3067, 2008.

[28] T. Minka, "Discriminative models, not discriminative training," Microsoft Research, Tech. Rep. MSR-TR-2005-144, 2005.

[29] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proc. International joint conference on Artificial intelligence*, vol. 81, 1981, pp. 674 –679.

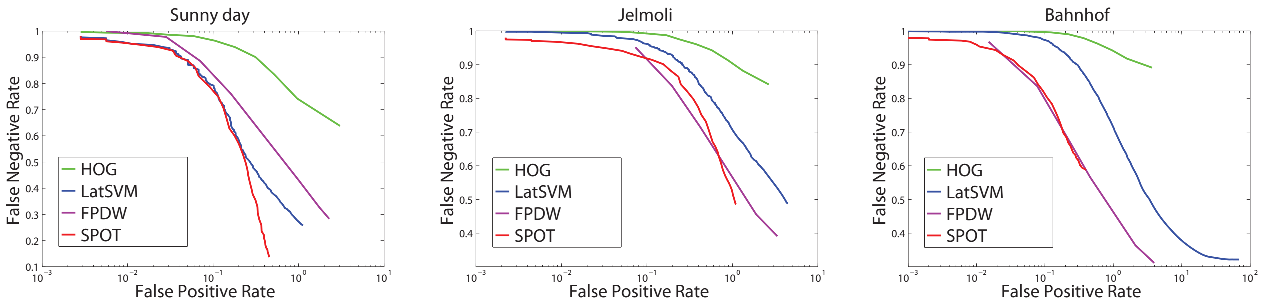

Fig. 6. Miss rate and false positive rate of HOG, LatSVM, FPDW, and mst-SPOT tracker in pedestrian detection on three movies from the ETH data set (lower curves indicate better performance). Figure best viewed in color.

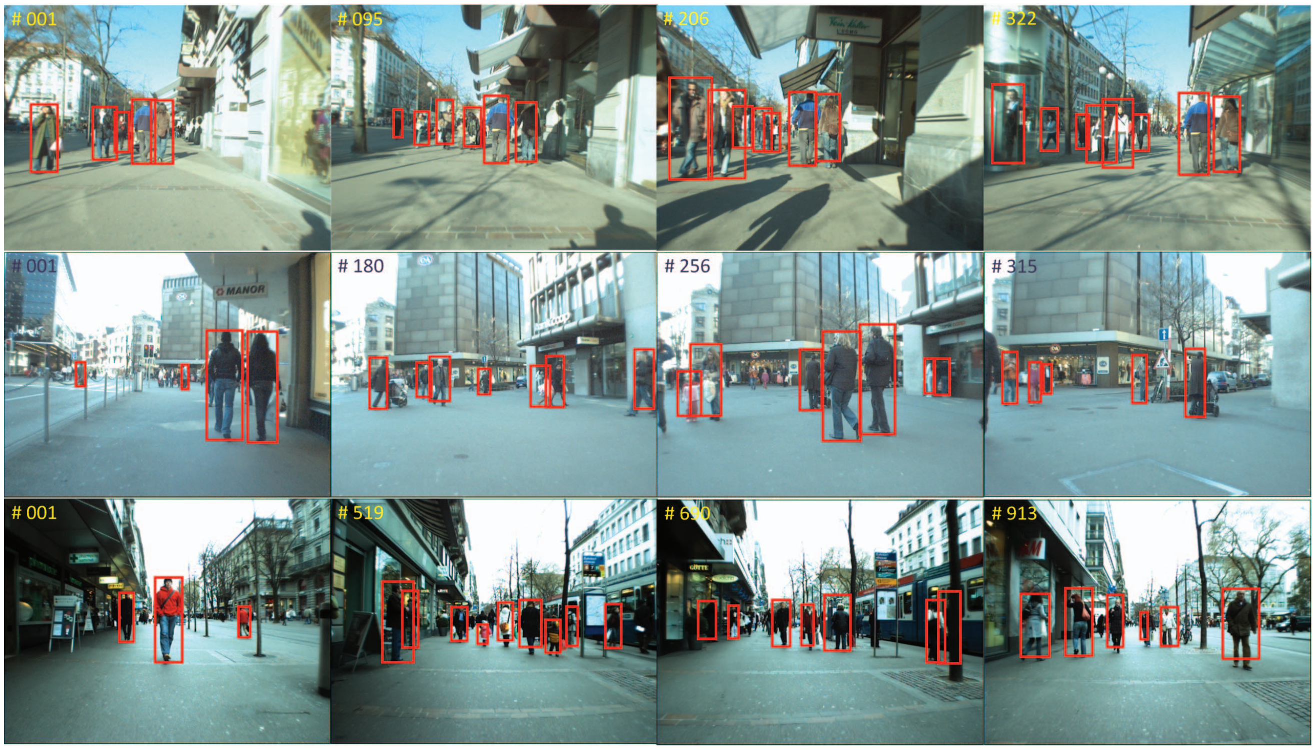

Fig. 7. Pedestrian detections obtained using mst-SPOT on the *Sunny day* (top), *Jelmoli* (middle), and *Bahnhof* (bottom) videos from the ETH data set. Figure best viewed in color.

[30] P. Sand and S. Teller, "Particle video: Long-range motion estimation using point trajectories," *Intl J. Computer Vision*, vol. 80, no. 1, pp. 72–91, 2008.

[31] J. Shi and C. Tomasi, "Good features to track," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 1994.

[32] C. Bibby and I. Reid, "Real-time tracking of multiple occluding objects using level sets," in *proc. European Conference on Computer Vision*, 2010.

[33] M. Isard and A. Blake, "Condensation - conditional density propagation for visual tracking," *Intl J. Computer Vision*, vol. 29, no. 1, pp. 5–28, 1998.

[34] S. Birchfield, "Elliptical head tracking using intensity gradients and color histograms," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 1998.

[35] A. Adam, E. Rivlin, and I. Shimshoni, "Robust fragments-based tracking using the integral histogram," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2006, pp. 798–805.

[36] X. Mei and H. Ling, "Robust visual tracking and vehicle classification via sparse representation," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 33, pp. 2259–2272, 2011.

[37] H. Grabner, C. Leistner, and H. Bischof, "Semi-supervised on-line boosting for robust tracking," in *proc. European Conference on Computer Vision*, 2008.

[38] S. Hare, A. Saffari, and P. Torr, "Struck: Structured output tracking with kernels." in *Proc. IEEE Intl Conf. Computer Vision*, 2011, pp. 263–270.

[39] T. Yu and Y. Wu, "Decentralized multiple target tracking using netted collaborative autonomous trackers," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2005.

[40] K. Okuma, A. Taleghani, N. D. Freitas, J. J. Little, and D. G. Lowe, "A boosted particle filter: Multitarget detection and tracking," in *proc. European Conference on Computer Vision*, 2004, pp. 28–39.

[41] T. Yu and Y. Wu, "Collaborative tracking of multiple targets," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 1, 2004, pp. 834–841.

[42] G. Duan, H. Ai, S. Cao, and S. Lao, "Group tracking: exploring mutual relations for multiple object tracking," in *proc. European Conference on Computer Vision*, 2012, pp. 129–143.

[43] L. Cerman, J. Matas, and V. Hlavac, "Sputnik tracker: Looking for a companion improves robustness of the tracker," in *Proceedings of the Scandinavian Conference on Image Analysis*, 2009.

[44] M. Yang, Y. Wu, and G. Hua, "Context-aware visual tracking," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 31, no. 7, pp. 1195–1209, 2009.

[45] H. Grabner, J. Matas, L. Van Gool, and P. Cattin, “Tracking the invisible: Learning where the object might be,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2010, pp. 1285–1292.
[46] T. B. Dinh, N. Vo, and G. Medioni, “Context tracker: Exploring supporters and distracters in unconstrained environments,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2011, pp. 1177–1184.
[47] Y. Li and R. Nevatia, “Key object driven multi-category object recognition, localization and tracking using spatio-temporal context.” in *proc. European Conference on Computer Vision*, 2008, pp. 409–422.
[48] S. Stalder, H. Grabner, and L. Van Gool, “Cascaded confidence filtering for improved tracking-by-detection,” in *proc. European Conference on Computer Vision*, 2010, pp. 369–382.
[49] P. M. Roth, S. Sternig, H. Grabner, and H. Bischof, “Classifier grids for robust adaptive object detection,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2009, pp. 2727–2734.
[50] G. J. Edwards, C. J. Taylor, and T. F. Cootes, “Interpreting face images using active appearance models,” in *Proc. IEEE Conf. Automatic Face and Gesture Recognition*, 1998, pp. 300–305.
[51] A. Blake and M. Isard, *Active shape models*. Springer, 1998.
[52] D. Cristinacce and T. Cootes, “Facial feature detection and tracking with automatic template selection,” in *7th International Conference on Automatic Face and Gesture Recognition*, 2006, pp. 429–434.
[53] Y. Yang and D. Ramanan, “Articulated pose estimation using flexible mixtures of parts,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2011, pp. 1385–1392.
[54] Y. Yang, S. Baker, A. Kannan, and D. Ramanan, “Recognizing proxemics in personal photos,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2012, pp. 3522–3529.
[55] C. Desai and D. Ramanan, “Detecting actions, poses, and objects with relational phraselets,” in *European Conference on Computer Vision*, 2012, pp. 158–172.
[56] M. Andriluka and S. R. an Bernt Schiele, “Pictorial structures revisited: People detection and articulated pose estimation,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2009, pp. 1014–1021.
[57] V. Ferrari, M. Marin-Jimenez, and A. Zisserman, “Pose search: Retrieving people using their pose,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2009.
[58] C. Chow and C. Liu, “Approximating discrete probability distributions with dependence trees,” *IEEE Transactions on Information Theory*, vol. 14, no. 3, pp. 462–467, 1968.
[59] X. Zhu and D. Ramanan, “Face detection, pose estimation, and landmark localization in the wild,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2012, pp. 2879–2886.
[60] J. Lafferty, A. McCallum, and F. Pereira, “Conditional random fields: Probabilistic models for segmenting and labeling sequence data,” in *Proceedings of the* 18^{th} *International Conference on Machine Learning*, 2001, pp. 282–289.
[61] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun, “Large margin methods for structured and interdependent output variables,” *Journal of Machine Learning Research*, vol. 6, pp. 1453–1484, 2005.
[62] M. Blaschko and C. Lampert, “Learning to localize objects with structured output regression,” *proc. European Conference on Computer Vision*, pp. 2–15, 2008.
[63] P. Ott and M. Everingham, “Shared parts for deformable part-based models,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2011, pp. 1513–1520.
[64] M. Collins, “Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms,” in *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*. Association for Computational Linguistics, 2002, pp. 1–8.
[65] L. van der Maaten, M. Welling, and L. K. Saul, “Hidden-unit conditional random fields,” *JMLR W&CP*, vol. 15, pp. 479–488, 2011.
[66] L. Bottou, “Online algorithms and stochastic approximations,” in *Online Learning and Neural Networks*, 1998.
[67] S. Shalev-Shwartz, Y. Singer, and N. Srebro, “Pegasos: Primal estimated sub-gradient solver for svm,” in *Proceedings of the 24th international conference on Machine learning*, ser. Intl Conf. on Machine Learning. ACM, 2007, pp. 807–814.
[68] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer, “Online passive-aggressive algorithms,” *Journal of Machine Learning Research*, vol. 7, no. Mar, pp. 551–585, 2006.
[69] J. C. Platt, “Probabilistic outputs for support vector machines and comparison to regularized likelihood methods,” *Advances in Large Margin Classifiers, MIT Press*, 1999.
[70] P. Felzenszwalb and D. Huttenlocher, “Distance transforms of sampled functions.” Cornell University CIS, Tech. Rep., 2004.
[71] G. Zoutendijk, *Methods of Feasible Directions*. Amsterdam, The Netherlands: Elsevier Publishing Company, 1960.
[72] M. Dredze and K. Crammer, “Confidence-weighted linear classification,” in *Intl Conf. on Machine Learning*. ACM, 2008, pp. 264–271.
[73] J. Kwon and K. Lee, “Visual tracking decomposition,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2010, pp. 1269–1276.
[74] A. Ess, B. Leibe, K. Schindler, , and L. van Gool, “A mobile vision system for robust multi-person tracking,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2008.
[75] P. Dollár, S. Belongie, and P. Perona, “The fastest pedestrian detector in the west,” in *BMVC*, 2010.
[76] P. Dollár, C. Wojek, B. Schiele, and P. Perona, “Pedestrian detection: An evaluation of the state of the art,” vol. 99, 2011.

Lu Zhang received her M.Sc. degree in Signal Processing from University of Chinese Academy of Sciences, China, in 2010. Her M.Sc. thesis was about hardware-based multi-target tracking with a celestial background. She is currently a Ph.D. student in the Pattern Recognition & Bioinformatics Group of Delft University of Technology. Her research interests include model-free tracking as well as object, gesture, and action recognition.

Laurens van der Maaten is an Assistant Professor in computer vision and machine learning at Delft University of Technology, The Netherlands. Previously, he worked as a post-doctoral researcher at University of California, San Diego, as a Ph.D. student at Tilburg University, and as a visiting Ph.D. student at University of Toronto. His research interests include deformable template models, dimensionality reduction, classifier regularization, and tracking.