2013 IEEE Conference on Computer Vision and Pattern Recognition

Structure Preserving Object Tracking

Lu Zhang Laurens van der Maaten
Computer Vision Lab, Delft University of Technology
Mekelweg 4, 2628 CD Delft, The Netherlands
{lu.zhang, l.j.p.vandermaaten}@tudelft.nl

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 (based on tracking-by-detection) 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 significant 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.

1. Introduction

Object tracking is a fundamental problem in computer vision with applications in a wide range of domains. Whereas significant progress has been made in tracking specific objects (*e.g.*, faces [22], humans [11], and rigid objects [15]), tracking generic objects remains hard. Since 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 [2, 12]. In model-free tracking, the object of interest is manually annotated in the first frame of a video sequence (using a rectangular bounding box). The annotated object needs to be tracked throughout the remainder of the video. Model-free tracking is a challenging task because (1) little 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.

Most tracking systems comprise three main components: (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 [7], the location model is based on the relative locations of objects, and the search strategy is a sliding-window exhaustive search.

In many applications, it is necessary to track more than one object. A simple approach to tracking multiple objects is to run multiple instances of a single-object model-free tracker. In this paper, we argue that this is suboptimal because such an approach fails to exploit spatial constraints between the objects. For instance, flowers move in the same direction because of the wind, cars drive in the same direction on the freeway, and when the camera shakes, all objects will move in the same direction. We show that it is practical to exploit such spatial constraints between objects in model-free trackers by developing a structure-preserving object tracker (SPOT) that *incorporates spatial constraints between objects* via a pictorial-structures framework [8]. We train the individual object classifiers and the structural constraints jointly using an online structured SVM [3]. Our experiments show that the incorporation of structural constraints leads to *substantial performance improvements in multi-object tracking*: SPOT performs very well on Youtube videos with camera motion, rapidly moving objects, object appearance changes, and occlusions. In addition, we show that SPOT may also be used to *significantly improve single-object trackers* by using part detectors in addition to the object detector, with spatial constraints between the parts.

In summary, our main contributions are: (1) we show that incorporating spatial constraints between objects in model-free trackers to improves their performance on multiple-object tracking and (2) we show that using a part-based model improves the performance of single-object model-free trackers. We discuss related work in section 2. Section 3 introduces our new tracker, section 4 presents our experimental results, and section 5 concludes the paper.

1063-6919/13 $26.00 © 2013 IEEE
DOI 10.1109/CVPR.2013.240

2. Related Work

Model-free trackers can be subdivided into (1) trackers that model only the appearance of the object itself [20] and (2) trackers that model the appearance of both the object and the background [2, 9, 12]. Recent results suggest that the latter type of trackers, which train a classifier to discriminate between the object and the background appearance, outperform the former type of trackers. This result is supported by theoretical results showing that discriminative models always outperform their generative counterparts on a discriminative tasks [17]. Hence, we will focus on learning discriminative object appearance models in this work.

Much of the recent work in model-free tracking focuses on exploring different feature representations for the object that is being tracked: among others, previous studies have used integral histograms [1], subspace learning [20], sparse representations [16], and local binary patterns [12]. In this work, we capitalize on the success of the Dalal-Triggs detector [7] 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 background. In particular, previous studies have investigated approaches based on boosting [9], random forests [12], multiple instance learning [2], and structured output learning to predict object transformations [10]. 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, an online structured SVM [3].

Simultaneous tracking of multiple objects has been studied a lot as well, in particular, in the context of tracking people (*e.g.*, [4, 18, 23, 25]). These trackers use a model of what a human looks like, which makes tracking much easier. By contrast, we aim to develop a model-free tracker that can track generic objects based on a single annotation without any prior knowledge. Up to the best of our knowledge, there is no previous work that attempts to perform such model-free tracking of multiple objects.

3. Structure-Preserving Object Tracker

The basis of our tracker is formed by the popular Dalal-Triggs detector [7], 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 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 [7], which is very important 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.

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; both w_i and h_i are fixed. The HOG features extracted from image $\mathbf{I}$ that correspond to locations inside the object bounding box 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 $(i, j) \in E$. 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* $C = \{B_1, \ldots, B_{|V|}\}$ as the sum of two terms: (1) an appearance score that measures the similarity 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 as follows:

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

Herein, the parameters $\mathbf{w}_i$ represent linear weights on the HOG features for object i, $\mathbf{e}_{ij}$ is a vector that represents the length and direction of the spring between object i and j, and the set of all parameters is denoted by Θ: $\Theta = \{\mathbf{w}_1, \ldots, \mathbf{w}_{|V|}, \mathbf{e}_1, \ldots, \mathbf{e}_{|E|}\}$. We treat the parameters λ_{ij} as a hyperparameter because we want to learn the spring coefficients $\mathbf{e}_{ij}$, *i.e.* we set $\forall i, j : \lambda_{ij} = \lambda$. The hyperparameter λ determines the trade-off between the appearance and deformation scores. We use Platt scaling [19] to convert the configuration score to a configuration likelihood $p(C|\mathbf{I}; \Theta)$.

Inference. Given the parameters of the model, finding the most likely object configuration amounts to maximizing Eqn. 1 over C. This maximization is intractable in general because it requires searching over exponentially many configurations, but for tree-structured graphs G, dynamic programming can be used to perform the maximization in a time that is linear in the number of parts; see [8] for details.

Learning. Like other model-free trackers [2, 9, 12], we use the previous images and tracked object configurations as positive examples to train our model. After observing an image $\mathbf{I}$ and inferring the object configuration C (by maximizing Eqn. 1), we perform a parameter update that aims to minimize the structured SVM loss ℓ [21]:

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

where $\Delta(C, \hat{C})$ is defined as follows:

$$\Delta(C, \hat{C}) = \sum_{i \in V} \left(1 - \frac{B_i \cap \hat{B}_i}{B_i \cup \hat{B}_i} \right). \qquad (2)$$

The union and intersection of the two bounding boxes is measured in terms of pixels. The loss can be rewritten as:

$$\ell(\Theta; \mathbf{I}, C) = \max_{\hat{C}} \Bigg(\mathrm{vec}(\Theta)^{\mathrm{T}} \left(\hat{\Psi} - \Psi \right) - \sum_{(i,j) \in E} \lambda \left(\|\hat{\mathbf{q}}_{ij}\|^2 - \|\mathbf{q}_{ij}\|^2 \right) + \Delta(C, \hat{C}) \Bigg),$$

where $\Psi = \left[\phi_1, \ldots, \phi_{|V|}, 2\lambda \mathbf{q}_{i_1 j_1}, \ldots, 2\lambda \mathbf{q}_{i_{|E|} j_{|E|}}\right]^{\mathrm{T}}$, $\mathrm{vec}(\cdot)$ concatenates all parameters in a column vector, and $\mathbf{q}_{ij} = \mathbf{x}_i - \mathbf{x}_j$. Because it is the maximum of a set of affine functions of Θ, the loss function is convex. The gradient of the structured SVM loss with respect to $\theta \in \Theta$ is given by:

$$\begin{aligned} \nabla_\theta \ell(\Theta; \mathbf{I}, C) &= \nabla_\theta s(C^*; \mathbf{I}, \Theta) - \nabla_\theta s(C; \mathbf{I}, \Theta) \\ &= \Psi^* - \Psi, \end{aligned} \qquad (3)$$

in which the configuration C^* is given by:

$$C^* = \operatorname*{argmax}_{\hat{C}} \left(s(\hat{C}; \mathbf{I}, \Theta) + \Delta(C, \hat{C}) \right). \qquad (4)$$

The configuration 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 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 parameters updates [5]. The updates take the form:

$$\theta \leftarrow \theta - \frac{\ell(\Theta; \mathbf{I}, C)}{\|\nabla_\theta \ell(\Theta; \mathbf{I}, C)\|^2 + \frac{1}{2K}} \nabla_\theta \ell(\Theta; \mathbf{I}, C), \qquad (5)$$

in which $K \in (0, +\infty)$ is a hyperparameter that controls the "aggressiveness" of the parameter update.

Initialization. The weights $\mathbf{w}_i$ are initialized by randomly selecting 50 negative examples from the first frame that have little to no overlap with the initial annotation $B_i^{(init)}$, and training SVMs to discriminate these negative examples from the positive example given by the initial annotation. The parameters $\mathbf{e}_{ij}$ are initialized based on the initial object annotations as well: $\mathbf{e}_{ij} \leftarrow \mathbf{x}_i^{(init)} - \mathbf{x}_j^{(init)}$.

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. Hence, we explore two approaches to construct a tree on the objects $i \in V$: (1) a star model [8] and (2) a minimum spanning tree model [24]. 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:

$$s(C; \mathbf{I}, \Theta) = \sum_{i \in V/r} \mathbf{w}_i^{\mathrm{T}} \phi(\mathbf{I}; B_i) - \sum_{(i,r) \in E} \lambda_i \|(\mathbf{x}_i - \mathbf{x}_r) - \mathbf{e}_i\|^2. \qquad (6)$$

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 $\sum_{ij \in E} \|\mathbf{x}_i - \mathbf{x}_j\|^2$.

Computational Complexity. The main computational costs of running our tracker are in the extraction of HOG features (which are shared between object detectors) and in the computation of the appearance score per object. After these appearance scores 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 $|V|$). Our Matlab implementation tracks multiple objects simultaneously in real-time.

4. Experiments

We performed two 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. An implementation of our tracker is available from `http://visionlab.tudelft.nl/spot`.

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.

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 [13]; 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, near objects with a very similar appearance. The average length of the videos is 842 frames. The left column of Figure 1 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 1 shows all labeled training data that is available to train our tracker.

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 model (star-SPOT), and (3) a SPOT tracker that uses a minimum spanning tree (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 [9] and the TLD tracker [12], which we use to separately track each object. The OAB and TLD trackers were run using the implementations provided by their developers.

We evaluate the performance of the trackers by measuring (1) *average distance error* (Err.): the average distance of the center of the identified bounding box to the center of the ground-truth bounding box and (2) *precision* (Prec.): the average 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 over all target objects, over all frames, and over five separate runs of the tracker. In all experiments with star-SPOT and mst-SPOT, we fixed $\lambda = 0.001$ and $K = 1$. In preliminary experiments, we found that are results are very robust under changes of λ and K.

Results. The performance of the five trackers on all nine videos is presented in Table 1. The results in the table show: (1) that our baseline no-SPOT tracker performs on par with state-of-the-art trackers such as TLD and OAB, 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 (mst-SPOT) are used. The performance improvements are particularly impressive for videos in which objects with a similar appearance are tracked, such as the *Car Chase*, *Parade*, 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 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. The mst-SPOT tracker outperforms star-SPOT in most videos, presumably, because a minimum spanning tree imposes direct (rather than indirect) constraints on the object locations.

Figure 1 shows five frames of all videos with the tracks obtained by mst-SPOT (colors correspond to objects). Videos showing the full tracks are provided online. We 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 shake. Whereas the 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 right 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 because of the partial occlusions. By contrast, the two SPOT trackers flawlessly track all flowers during the entire length of the video (2249 frames) by using 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.

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. The problem of using a global appearance model is that it is susceptible to partial occlusions of the object. By contrast, the appearance of (most of the) parts of the object remains unaltered by such occlusions. SPOT may be used to track the parts of a single object (treating them as individual objects in V) with structural constrains between the parts. This makes the tracker more robust to partial occlusions. Inspired by [8], 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 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).

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 model will use. As a latent SVM [8] is unlikely to work well on a single training example, we use a heuristic 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

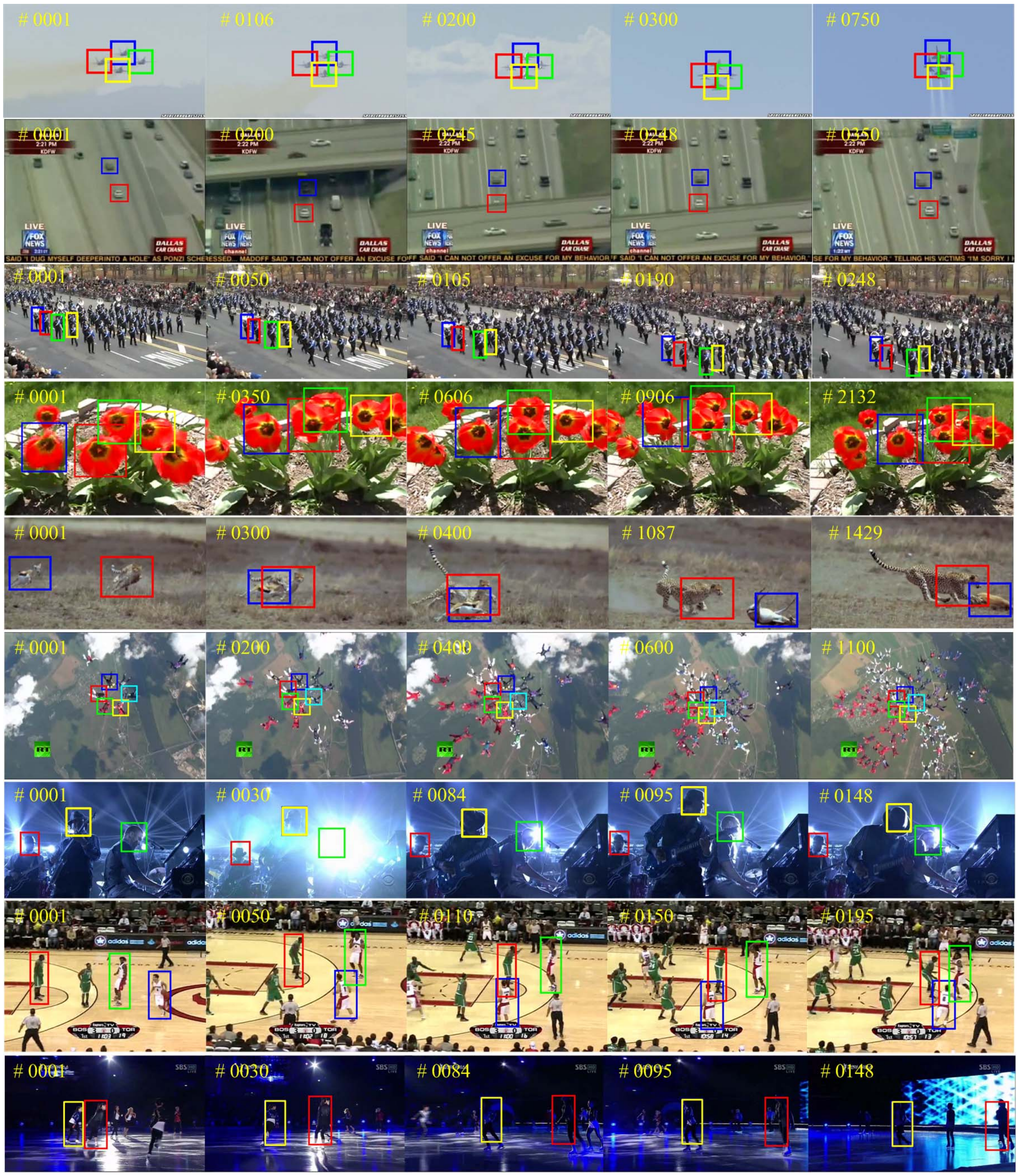

Figure 1. Tracking results obtained by mst-SPOT on all nine videos used in Experiment 1 (from top to bottom: *Air Show*, *Car Chase*, *Parade*, *Red Flowers*, *Hunting*, *Sky Diving*, *Shaking*, *Basketball*, and *Skating*). The colors of the rectangles indicate the different objects that are tracked.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

Table 1. Performance of five model-free trackers on multiple-object videos measured in terms of (1) average distance in pixels between centers of the predicted and the ground-truth bounding box (Err.; lower is better) and (2) precision (Prec.; higher is better). To measure the precision, 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 [9]		**TLD** [12]		**no-SPOT**		**star-SPOT**		**mst-SPOT**	
	Err.	*Prec.*	*Err.*	*Prec.*	*Err.*	*Prec.*	*Err.*	*Prec.*	*Err.*	*Prec.*
Air Show	9.3	0.86	31.3	0.53	8.8	**0.92**	**6.9**	**0.92**	10.7	0.68
Car Chase	121.8	0.57	11.2	0.76	24.8	0.78	11.2	0.82	**9.2**	**0.89**
Parade	12.7	**0.82**	8.8	0.71	62.3	0.29	19.4	0.35	**8.6**	0.69
Red Flowers	79.7	0.09	33.3	0.30	50.6	0.38	8.6	0.98	**8.2**	**0.99**
Hunting	104.9	0.25	166.4	0.08	171.7	0.07	29.2	0.72	**17.9**	**0.87**
Sky Diving	15.5	0.76	35.3	0.13	51.4	0.48	**6.73**	**0.98**	13.6	0.95
Shaking	61.9	0.47	14.3	0.47	58.3	0.47	28.7	0.38	**9.8**	**0.97**
Basketball	24.4	0.63	15.6	0.67	63.3	0.67	50.9	0.54	**12.7**	**0.85**
Skating	100.2	0.05	90.3	0.42	122.2	0.35	98.9	0.27	**14.9**	**0.85**
Avg. rank	3.8	3.4	2.9	3.3	4.3	3.1	2.4	2.9	**1.4**	**1.7**

initial global SVM $\mathbf{w}$ are large and positive:

$$B_i = \underset{B_i' \subset B}{\operatorname{argmax}} \sum_{(x',y') \in B_i'} \left(\max(0, w_{x'y'})\right)^2, \qquad (7)$$

where B_i and B denote the bounding box of the part and of the global object, respectively. 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 B_i to 40% of the width and height of the bounding box B, and we ensure that the selected part cannot have more than 50% overlap with the other parts. Unlike [8], 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 represent the parts, but this did not lead to performance improvements. In addition, using the same features for all detectors has computational advantages because the features only need to be computed once.

The experiments are performed on a publicly available collection of twelve videos [2]. 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 556 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 [2].)

Again, we evaluate the performance of the trackers by measuring the average distance error and the precision 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 [9], the MILBoost tracker [2], and the TLD tracker [12]. All trackers were run on a single scale; results with multi-scale trackers are presented in the supplemental material. We could not run the implementation of the MILBoost tracker 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 in [2], we adopt the results presented there.

Results. Table 2 presents the performance of all six trackers on all twelve videos. Figure 2 shows the tracks obtained with the MIL, OAB, TLD, and mst-SPOT trackers on seven of the twelve 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 twelve videos in terms of average distance between centers, and on nine of the twelve videos in terms of precision. 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 suggests that for object detection in still images, pictorial-structure models with a minimum spanning tree [24] may be better than those with a star tree [8].

5. Conclusion and Future Work

In this paper, we have developed a new model-free tracker that simultaneously 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. Moreover, we have showed that the SPOT tracker can also improve the tracking of single objects by including additional detectors for object parts in the tracker. 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 im-

Table 2. Performance of six model-free trackers on single-object videos measured in terms of (1) average distance in pixels (Err) between the centers of the predicted and the ground-truth bounding box (lower is better) and (2) precision (higher is better). To measure the precision, 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 [9]		**MIL** [2]		**TLD** [12]		**no-SPOT**		**star-SPOT**		**mst-SPOT**	
	Err.	*Prec.*	*Err.*	*Prec.*	*Err.*	*Prec.*	*Err.*	*Prec.*	*Err.*	*Prec.*	*Err.*	*Prec.*
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	189.2	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**
Avg. rank	4.3	3.9	3.7	4.1	4.0	3.8	3.2	2.8	3.6	3.1	**2.0**	**1.8**

plemented in other model-free trackers that are based on tracking-by-detection, such as the TLD tracker. It is likely that including structural constraints in such trackers will improve their performance in tracking of multiple objects, 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 [6]) or on GPLVMs [14]. We also plan to investigate whether it is possible to identify the relevant parts of a deformable object in a more principled way during (model-free) tracking by developing online learning algorithms for latent SVMs, and we intend to investigate whether online structured SVMs can be used to adapt deformable template models during tracking.

Acknowledgements. This work was supported by EU-FP7 Social Signal Processing (SSPNet) and by the China Scholarship Council. The authors thank David Tax, Marco Loog, and Martijn van de Giessen for helpful discussions.

References

[1] A. Adam, E. Rivlin, and I. Shimshoni. Robust fragments-based tracking using the integral histogram. In *CVPR*, pages 798–805, 2006.

[2] B. Babenko, M.-H. Yang, and S. Belongie. Robust object tracking with online multiple instance learning. *PAMI*, 33(8):1619–1632, 2011.

[3] S. Branson, P. Perona, and S. Belongie. Strong supervison from weak annotation: Interactive training of deformable part models. In *ICCV*, pages 1832–1839, 2011.

[4] M. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool. Online multiperson tracking-by-detection from a single, uncalibrated camera. *PAMI*, 33(9):1820–1833, 2011.

[5] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7(Mar):551–585, 2006.

[6] D. Cristinacce and T. F. Cootes. Automatic feature localisation with constrained local models. *Pattern Recognition*, 41(10):3054–3067, 2008.

[7] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, pages 886–893, 2005.

[8] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *PAMI*, 32(9):1627–1645, 2010.

[9] H. Grabner, M. Grabner, and H. Bischof. Real-time tracking via on-line boosting. In *BMVC*, pages 47–56, 2006.

[10] S. Hare, A. Saffari, and P. Torr. Struck: Structured output tracking with kernels. In *ICCV*, pages 263–270, 2011.

[11] M. Isard and J. Maccormick. Bramble: A Bayesian multi-blob tracker. In *CVPR*, pages 34–41, 2001.

[12] Z. Kalal, J. Matas, and K. Mikolajczyk. P-n learning: Bootstrapping binary classifiers by structural constraints. In *CVPR*, pages 49–56, 2010.

[13] J. Kwon and K. Lee. Visual tracking decomposition. In *CVPR*, pages 1269–1276, 2010.

[14] N. Lawrence. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *Journal of Machine Learning Research*, 6:1783–1816, 2005.

[15] V. Lepetit and P. Fua. Keypoint recognition using randomized trees. *PAMI*, 28(9):1456–1479, 2006.

[16] X. Mei and H. Ling. Robust visual tracking and vehicle classification via sparse representation. *PAMI*, 33:2259–2272, 2011.

[17] T. Minka. Discriminative models, not discriminative training. Technical Report MSR-TR-2005-144, Microsoft Research, 2005.

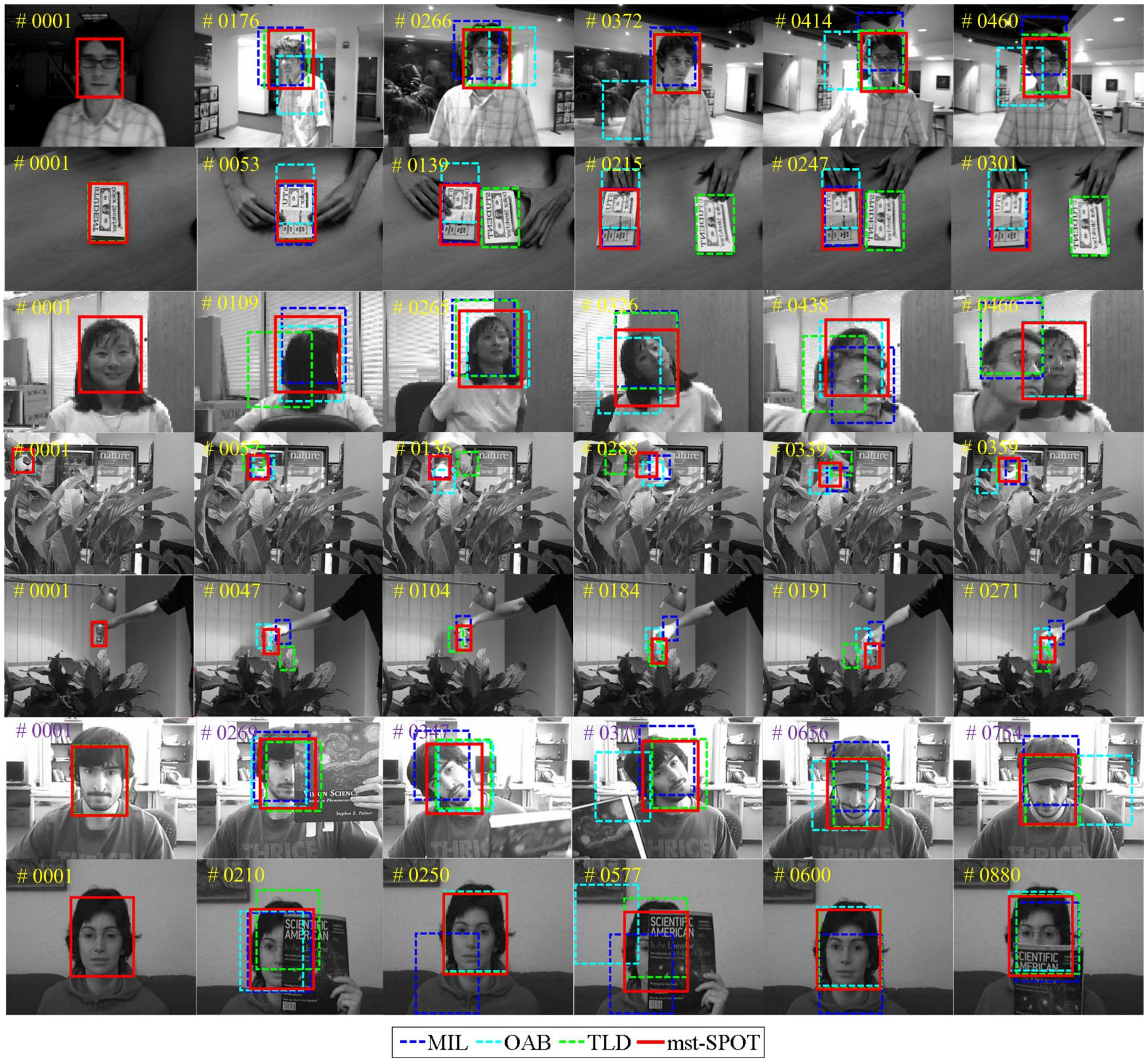

Figure 2. Tracking results on seven of the twelve videos (*David Indoor*, *Dollar*, *Girl*, *Tiger 2*, *Coke Can*, *Occl. Face 2*, and *Occl. Face*) obtained by the MIL, OAB, TLD, and mst-SPOT trackers. Figure best viewed in color.

[18] S. Pellegrini, K. Ess, A.and Schindler, and L. van Gool. You'll never walk alone: Modeling social behavior for multi-target tracking. In *ICCV*, pages 261–268, 2009.

[19] J. C. Platt. Probabilistic outputs for support vector machines and comparison to regularized likelihood methods. *Advances in Large Margin Classifiers, MIT Press*, 1999.

[20] D. Ross, J. Lim, R.-S. Lin, and M.-H. Yuang. Incremental learning for robust visual tracking. *IJCV*, 77(1):125–141, 2008.

[21] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, 2005.

[22] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *CVPR*, pages 511–518, 2001.

[23] Z. Wu, A. Thangali, S. Sclaroff, and M. Betke. Coupling detection and data association for multiple object tracking. In *CVPR*, pages 1948–1955, 2012.

[24] Z. Xiangxin and D. Ramanan. Face detection, pose estimation, and landmark localization in the wild. In *CVPR*, pages 2879–2886, 2012.

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