Occlusion and Motion Reasoning for Long-Term Tracking

Yang Hua, Karteek Alahari, and Cordelia Schmid

Inria*

Abstract. Object tracking is a reoccurring problem in computer vision. Tracking-by-detection approaches, in particular Struck [20], have shown to be competitive in recent evaluations. However, such approaches fail in the presence of long-term occlusions as well as severe viewpoint changes of the object. In this paper we propose a principled way to combine occlusion and motion reasoning with a tracking-by-detection approach. Occlusion and motion reasoning is based on state-of-the-art long-term trajectories which are labeled as object or background tracks with an energy-based formulation. The overlap between labeled tracks and detected regions allows to identify occlusions. The motion changes of the object between consecutive frames can be estimated robustly from the geometric relation between object trajectories. If this geometric change is significant, an additional detector is trained. Experimental results show that our tracker obtains state-of-the-art results and handles occlusion and viewpoints changes better than competing tracking methods.

1 Introduction

Although tracking objects is a well-established problem in computer vision [11,22,23,46,48,49], it still remains a challenging task. Many of the previous works have approached this problem from the perspective of either a motion tracker or an object detector. In recent years, tracking-by-detection, i.e., approaches that treat the tracking problem as a task of detecting the object over time [2,3,18,20,24,27,40,46,47], has become an increasingly popular method for object tracking. In fact, the latest evaluation papers [36,42,48] have shown such an approach, namely Struck [20], to be the best-performer on a diverse set of examples. One of the critical steps in these methods is to update and improve the object model over time. Consider the example in Figure 1. It shows a scenario where the object of interest—a car—is occluded when it goes under a bridge. If the model is updated in every frame, this results in the well-known issue of drift [32]. In other words, a part of the bridge might be tracked instead of the car in the latter frames in the sequence. In this paper we present an algorithm which can handle such occlusions as well as significant viewpoint changes in a principled way, based on state-of-the-art quasi-dense long-term trajectories [35,45].

* LEAR team, Inria Grenoble Rhône-Alpes, Laboratoire Jean Kuntzmann, CNRS, Univ. Grenoble Alpes, France.

D. Fleet et al. (Eds.): ECCV 2014, Part VI, LNCS 8694, pp. 172–187, 2014.

These trajectories rely on dense optical flow [8] in combination with a forward-backward matching criterion. Sundaram et al. [45] showed that they significantly outperform the Kanade-Lucas-Tomasi (KLT) tracker [30], often used in tracking approaches [14,24]. Our main contribution is to use these long-term trajectories in combination with graph-cut based track labelling to identify the state of the object, e.g., partial or full occlusion, as well as change in viewpoint, and to choose and adapt positive object samples accordingly to improve the model.

The goal of this paper is to track the object, given a bounding box initialization in the first frame. Our approach begins by learning an initial appearance model from the annotation in the first frame, similar to [20,24,46], and uses it to propose candidate object locations in the latter frames. We incorporate motion cues to refine the candidate region search space and avoid incorrect object proposals (Section 2). In order to determine whether a candidate location in a frame contains the object, we compute long-term motion tracks in the video [45], and use them to predict the state of the object, i.e., the transformation it has undergone with respect to the previous frames (Section 3). More specifically, we estimate states such as, partial or full occlusion, change in appearance of the object. This is achieved with an energy-based formulation, where the task is to assign each track an object or background label. When a significant part of the tracks within a candidate region belong to the background, the object is identified to be occluded. We will show that other types of change in state, such as a significant change in viewpoint, can also be estimated with our formulation. With this additional cue in hand, we build a temporally-evolving object model which deals with these state changes by updating the initially learned detector accordingly (Section 3.2). In essence, our tracker proposes a new way to interleave the motion-based and tracking-by-detection paradigms. Its performance is evaluated on sequences from a recent benchmark dataset [48] and videos used in [3,20,24,46] (Section 5).

1.1 Related Work

The two key components of a tracking algorithm are the object representation (appearance model) and the inference method used to localize it in each frame. Examples of object representations include colour histograms [11] as well as appearance representations learned with generative [26,29,33,41] and discriminative [2,10,20,46] models. Our work builds on these discriminative models by using the estimated state of the object to update the model. The inference methods range from Kalman filtering techniques, to those that use multiple cues [4,5,13,34,37,38,43,44] and fuse their results with methods like particle filtering [22], error analysis [44], and Markov chain Monte Carlo schemes [37]. These inference methods have shown promising results, but tend to suffer from drift. Furthermore, it is unclear if they can recover from the object leaving or re-entering the field of view.

Given the significant advances in algorithms for object detection [12,15], the tracking-by-detection approach [2,27,40,47] has gained popularity. A variant of these approaches, known as adaptive tracking-by-detection, updates the object

Fig. 1. Three sample frames from the Carchase sequence in the TLD dataset [24]. Each image also shows the result of our method (in green) and Struck [20] (in red). As the car starts to move to a severely occluded state, updating the appearance model with the prediction result leads to model drift, as shown by the Struck result. Our proposed method estimates such states, and does not update the model in such cases. (Best viewed in pdf.)

model over time [3,18,20,24,46]. It typically consists of two phases: (i) tracking, where the detector is used to predict the object location in each frame; and (ii) learning, where the estimated object locations (in the form of bounding boxes) are used to generate training examples to improve the learned object model. Observing that many of the adaptive methods lack a principled way of generating the training samples, a joint structured output formulation (Struck) was proposed in [20]. This work learns a function to predict the object location in a frame based on its location in the previous frame and a predefined search space. The prediction function is a dot product of a weight vector and a joint kernel map. This weight vector is learned and updated with an online structured output framework. Our approach is based on a similar philosophy, in that, we learn and update a prediction function. However, we use state-of-the-art long-term motion tracks [45] to determine the state of the object and produce an effective set of training exemplars for the update step. In the example shown in Figure 1, we predict that the object is severely occluded in the middle frame and thus do not update our detector. Note that Struck (result shown in red in Figure 1) drifts onto a part of the bridge in this example.

Our method is also closely related to two other recent approaches [24,46]. The TLD algorithm [24] aims to combine the benefits of tracker- and detector-based approaches. It decomposes the tracking task into specialized components for tracking, learning and detection, which are run simultaneously. The result of this algorithm is a combination of the predictions from the frame-to-frame tracking based on median optical flow and a detection component. The two components mutually update each other. Specifically, the results of the tracker provide training data to update the detector, and the detector re-initializes the tracker when it fails, for example, when the object leaves the field-of-view. While this is an interesting approach, it is somewhat restrictive as the object model is a set of fixed-size template patches, i.e., TLD cannot handle severe changes in object size, such as the scenario shown in Figure 2. Furthermore, the motion information is limited to frame-to-frame constraints.

Fig. 2. Three frames from the CarScale sequence in [48] are shown to highlight the severe change in object size over time in some of the videos

Supancic and Ramanan [46] presented a tracking-by-detection method, where the detector is updated using a pre-fixed number of frames, i.e., the top-k frames chosen according to an SVM objective function, irrespective of the state of the object. This does not handle long-term occlusions. In contrast, our algorithm updates the model with only the frames that show a significant presence of the object, as it relies on the long-term motion cues to choose the training exemplars, unlike [46] which uses only the detector.

We experimentally compare with these related works [20,24,46], and show the benefits of our approach in Section 5.

2 Overview

In line with the tracking-by-detection approach, our tracker comprises three stages. First, a detector is learned from a given training set of positive and negative exemplars. Second, we track with this learned detector. Third, we update the object model with a carefully selected subset of frames. We now present an overview of these stages and then provide more details in Sections 3 and 4.

Initial Detector and Tracking. An initial object detector is required to set off our tracker. It is learned with a training set, where the positive example is the ground truth annotation in the first frame of the sequence, and the negative samples are harvested from the frame, similar to [46]. The initial model is then learned with HOG features [12,17] extracted from the bounding boxes and a linear SVM.

The detector is used to predict candidate locations of the object in other frames. In each frame, we find the most likely location of the object by evaluating the detector in a region estimated from motion cues (optical flow computed from the previous frame), and then choosing the bounding box with the best detection score, as shown in Figure 3. The motion-refined search is not only computationally efficient, but also avoids incorrect detections due to background clutter. Note that the bounding box obtained from this step is not labelled as the object yet.

We compute and analyze the motion cues to make the object label assignment in each frame. To this end, we extract long-term point tracks which extend over many frames in the video [45], see Figure 3-*Left*. At this stage, we discard tracks less than 5 frames long, which are typically less reliable. We then propose an energy-based formulation to label the remaining tracks as object or background.

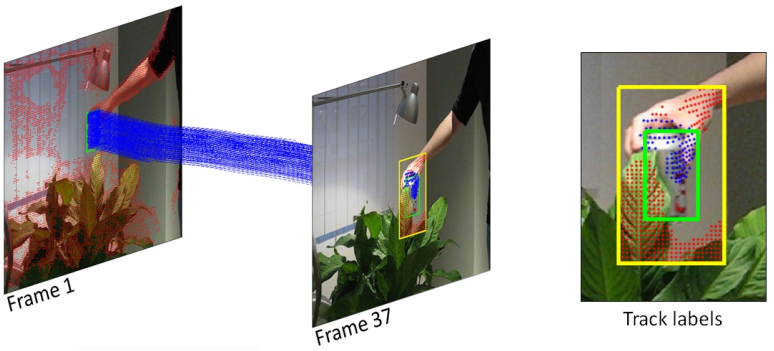


Fig. 3. Left: Long-term tracks beginning in frame 1 of the Coke sequence [48]. The yellow box shows the search region used to compute the bounding box most likely to contain the object (green box). We use the tracks to estimate the object state. Right: Close-up of the track labels in frame 37. Here, less than 60% of the tracks within the predicted bounding box are assigned to the object (blue), and the remaining are labelled as background (red). Thus, the object is predicted to be in an occluded state. (Best viewed in pdf.)

This is related to the labelling framework used in [28] for motion-clustering tracks. The tracks within the bounding box in the first frame, i.e., the ground truth annotation, are initialized with the object label, and those that lie outside are given the background label. With these initial assignments and pairwise energy terms (which measure track similarity), we optimize the energy function and label all the new tracks, i.e., tracks that begin in the second or latter frames, see Figure 3-*Right*.

Occlusion. If a significant part (40% or more) of the tracks within the bounding box take the background label (as in Figure 3), we consider the object to be in an occluded state. In this case, the object model is not updated with the detection result. We continue to track the object with the non-updated detector as long as there are object tracks and a detection response. This step avoids model drift [32,46]. For example, in the sequence in Figure 1, the model is not updated with the frame shown in the middle to avoid the tracker drifting onto a part of the bridge, which occludes the car. To handle cases where the object re-appears after a full occlusion (e.g., the frame on the right in Figure 1), the detector is evaluated over the entire image in subsequent frames.

Temporally-Evolving Detector. When the object is not occluded in a frame, the long-term tracks are used to measure geometric transformations that the object may have undergone, such as change in scale, rotation (see Figures 2 and 4). In this work, we approximate these transformations with a similarity matrix [21],[1] and estimate it with track-point correspondences between consecutive frames. The bounding box is then refined with the estimated transformation

[1] Other transformations, such as homography, can also be used here.

and is assigned the object label. This is illustrated on an example in Figure 4. Based on the severity of the transformation, we either: (i) update the existing detector; or (ii) learn a new detector. In summary, our detector model evolves temporally to account for changes in object appearance by updating, i.e., learning, with new positive instances.

3 Motion Cues in the Tracker

Motion cues serve two purposes in our algorithm: (1) to determine the search region for evaluating the detector; and (2) to estimate the state of the object. We use dense optical flow computed between two frames for the first task, and cues extracted from long-term tracks for the second.

Given the bounding box labelled as the object in a frame, we compute optical flow [8] for all the pixels within the box, and obtain the median flow. With this flow estimate, the bounding box is translated onto the following frame, and the area surrounding it (an enlarged box) is considered as the search region for the detector. In other words, we restrict the search space for the object detector when finding the most likely location of the object in a new frame. An example is illustrated in Figure 3. This useful cue is inspired by many of the traditional motion-based trackers, but is limited to providing only local motion information. We argue that these local cues are insufficient to reliably estimate (e.g., when the optical flow measurements are poor) whether the new bounding box contains the object or not. Our work integrates richer cues computed from long-term tracks into the framework to make a robust estimation of the state of the object. We achieve this with an energy-based formulation involving the long-term tracks.

Each track is represented with a random variable X_i and takes a label $x_i \in \{0, 1\}$, where 0 denotes the background and 1 is the object. Let n denote the number of tracks, and $\mathbf{X} = \{X_1, X_2, \ldots, X_n\}$ be the set of random variables. A labelling $\mathbf{x}$ refers to any possible assignment of labels to the random variables, and takes values from the set $\{0, 1\}^n$. The cost of a label assignment $E(\mathbf{X} = \mathbf{x})$, or $E(\mathbf{x})$ in short, is defined as:

$$E(\mathbf{x}) = \sum_{i=1}^{n} \phi_i(x_i) + \lambda \sum_{(i,j)\in\mathcal{E}} \phi_{ij}(x_i, x_j), \tag{1}$$

where $\phi_i(x_i)$ is the unary term to measure how likely it is for the track i to take label x_i. The function $\phi_{ij}(x_i, x_j)$ is a smoothness term to encourage similar tracks to take the same label. The set of pairs of interacting tracks is denoted by $\mathcal{E}$, and λ is a parameter to regulate the relative strength of the unary and the pairwise terms. The energy function (1) is minimized exactly to obtain the globally optimal labels for the tracks.

The pairwise smoothness term takes the form of a generalized Potts model [6] and is given by:

$$\phi_{ij}(x_i, x_j) = \begin{cases} \exp(-\lambda_d d(i,j)) & \text{if } x_i \neq x_j, \\ 0 & \text{otherwise,} \end{cases} \tag{2}$$

where $d(i,j)$ measures the dissimilarity between the two tracks i and j and λ_d is a parameter set to 0.1. This term is defined between pairs of neighbouring tracks, and it assigns a low cost for two dissimilar tracks to take different labels. We use the popular dissimilarity measure [7] computed as the maximum distance between time-corresponding spatial coordinates $\mathbf{p}_t^i, \mathbf{p}_t^j$ and velocity values $\mathbf{v}_t^i, \mathbf{v}_t^j$ as: $d(i,j) = \max_t \ ||\mathbf{p}_t^i - \mathbf{p}_t^j||_2^2 \frac{||\mathbf{v}_t^i - \mathbf{v}_t^j||_2^2}{5\sigma_t^2}$. This maximum is computed over points in time where the two tracks overlap. The first term, $||\mathbf{p}_t^i - \mathbf{p}_t^j||_2^2$, measures the spatial Euclidean distance, and the second term is the vector difference of the velocities estimated over 5 frames, i.e., $\mathbf{v}_t^i = \mathbf{p}_i^{t+5} - \mathbf{p}_i^t$.

For the unary terms, all the tracks that begin within the ground truth annotation in the first frame are assigned a very high cost to take the background label. This prevents them from changing their label in the latter frames, and is essentially a hard assignment of the object label. Inversely, tracks that lie outside the annotation in the first frame are given a very high cost to take the foreground label. The hard label assignment within the ground truth annotation can be refined, for example, by assigning a subset of the tracks with the object label using Grabcut-like segmentation techniques [9]. We found this refinement step to be non-essential in our case, since the track labels are used in combination with other cues, and not directly to determine the object location. The unary term for any new tracks starting in the second frame and beyond is defined as:

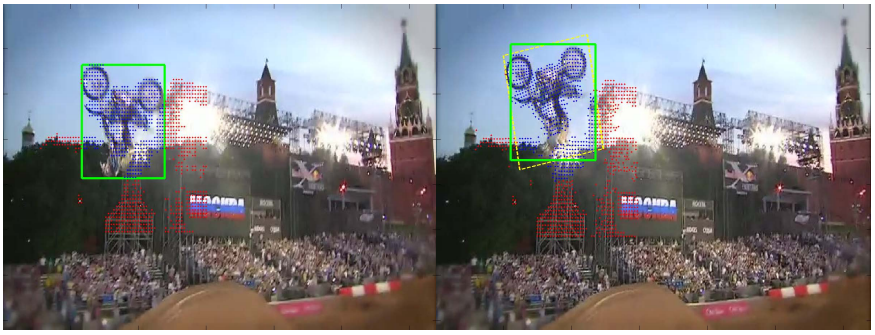
$$\phi_i(x_i = 1) = \begin{cases} 1 - \frac{1}{1+\exp(-(\alpha_t d_t + \beta_t))} & \text{if track } i \in \text{box}_t, \\ 0.5 & \text{otherwise,} \end{cases} \tag{3}$$

and $\phi_i(x_i = 0) = 1 - \phi_i(x_i = 1)$. Here, d_t is the SVM detection score for box_t, the bounding box estimate in frame t. The scalars α_t and β_t map this score into a probabilistic output, and are computed using Platt's method [39]. The intuition behind this unary term is that new tracks within a strong detection are likely to belong to the object. For tracks that lie outside the detection box, we allow the pairwise similarity terms to decide on their labels by giving an equal unary cost for assigning object or background labels.

In order to minimize the energy function (1) we apply the mincut/maxflow algorithm [19,25] on a corresponding graph, where each track is represented as a node. All the tracks within the search region in frame t (shown as a yellow box in Figure 3) are added as nodes. Additionally, tracks labelled in the previous frames which continue to exist in the frame are added. The unary and pairwise costs, computed as described earlier, are added as weights on the edges in the graph. We then perform *st*-mincut on the graph to get the optimal labels for all the nodes. Building the graph and performing inference on it in every frame allows us to update the labels of existing tracks based on new evidence from neighbouring tracks. An illustration of track labels is shown in Figure 3.

3.1 Predicting the State

With the track labels in hand, we determine whether the object has been occluded or a change in viewpoint has occurred. If more than 40% of the tracks



(a) Frame 1 (b) Frame 4

Fig. 4. Sample frames from the MotorRolling sequence [48], where the object undergoes a significant transformation (rotates counter clockwise). (a) Frame 1 showing the bounding box in the first frame. (b) The result of our tracker is shown is green. We show the bounding box transformed with the estimated similarity in yellow. (Best viewed in pdf.)

within box_t belong to the background, it is marked as a partial occlusion. We identify a full occlusion of the object if more than 80% of the tracks are assigned the background label. In other cases where a majority of the tracks continue to belong to the object, we verify if there have been any other transformations, see Figures 2 and 4 for two such examples. We model these transformations with a similarity matrix. It is estimated with a RANSAC approach [21], using points on the tracks (inside the box_{t-1}) in frames $t-1$ and t as correspondences. Since it is feasible to obtain more reliable point correspondences between consecutive frames, we compute frame-to-frame similarity matrices, and then accumulate them over a set of frames. For example, the transformation S_3^1, from frame 1 to 3, is computed as the product of the transformations S_3^2 and S_2^1. When a similarity matrix shows a significant change in scale or rotation, fixed empirically as 15% and 10° respectively in all the experiments, we mark the state as change in viewpoint.

To sum up, the candidate region box_t is labelled as occluded when a full occlusion state if predicted. When a change in viewpoint is estimated, box_t is transformed with the similarity matrix S_t^1 to obtain box_t^S, which is then assigned the object label. In other cases, i.e., neither occlusion nor change in viewpoint, box_t takes the object label.

3.2 Re-training the Model

Re-training (or updating) the model is crucial to the success of a tracking algorithm to handle situations where the object may change in some form over time. We use the predicted state of the object to precisely define the update step as follows.

Case 1: No change in state. The model update is straightforward, if the object is neither occluded, nor has undergone any of the other transformations. The new bounding box, box_t, is treated as a positive exemplar, and is used to update the SVM classifier.

Case 2: Occlusion. When the object is in a (partial or fully) occluded state, the classifier is not updated.

Case 3: Change in viewpoint. The detection result box_t in this case is transformed with the estimated similarity matrix to box_t^S. We then fit an image-axes-aligned box that encloses box_t^S, as illustrated in the example in Figure 4. This transformation changes either the scale or the aspect ratio of the bounding box containing the object. Recall that our initial detector is trained from a single positive example at one scale, and adding other samples with different scales (or aspect ratios) will deteriorate it. We choose to train a new detector with the new bounding box in frame t, and maintain a set of detectors which capture various scales and aspect ratios of the object, as it changes over time. This idea of maintaining multiple detectors for an object category is similar in spirit to exemplar SVMs [31].

A summary of our method is given in Algorithm 1. Note that in the case of a full occlusion, the best detection is obtained by running the detector over the entire image. The state is estimated based on the strength of the detection, and is set either to occlusion or no change in state, in which case a new track is started.

4 Implementation Details

Detector. We chose a linear SVM and HOG features to learn the object detector in this work, following a recent approach [46] which showed its efficacy on the tracking problem. The regularization parameter in the SVM is fixed to 0.1 for all our experiments. The SVM objective function is minimized with LIBLINEAR [16]. The initial detector is learned with one positive sample in the first frame and many negative examples harvested from bounding boxes (sampled from the entire image) that do not overlap with the true positive by more than 10%. We also perform 5 iterations of hard negative mining, similar to [46]. The learned detector is run at its original scale in the motion-predicted search region. Recall that we handle severe changes in object state (change in scale, rotation) by building a set of detectors (Section 2). For all the experiments, we fixed the maximum size of this set to 4, and replaced the worst performing detector (i.e., the detector with the lowest score when evaluated on the new exemplar), whenever necessary. We found this approach to work better in practice compared to one where a single multiscale detector is used. To update the detector efficiently with new samples, we use the standard warm-start strategy [16,46].

State Prediction. The parameter λ in the energy function (1), which controls the strength of the unary and pairwise terms is set to 1 in all our experiments. Pairwise terms are added between pairs of tracks that are less than a distance of 5 pixels in a frame. We minimize (1) with the graph cut algorithm [19,25].

Algorithm 1. Our approach for tracking an object and estimating its state

```
Data: Image frames 1...n, Object location box_1 in frame 1
Result: Object location box_t and state_t in frames t = 2...n
Learn initial detector in frame 1 (Section 2)
Compute long-term tracks (Section 2)
for t = 2...n do
    box_t ← Best detection in frame t
    Compute track labels (Section 3)
    state_t ← Estimate object state in frame t (Section 3.1)
    switch state_t do
        case Full occlusion
            box_t ← ∅
            No detector update
        end
        case Partial occlusion
            No detector update
        end
        case Change in viewpoint
            S_t^1 ← Estimate the transformation
            box_t^S ← Transform(box_t, S_t^1) (Section 3.2)
            box_t ← box_t^S
            Learn new detector model (Section 3.2)
        end
        case Other
            Update detector model (Section 3.2)
        end
    endsw
end
```

The thresholds for determining a partial or full occlusion are empirically fixed to 40% and 80% respectively in all our experiments.

5 Experimental Analysis

We now present a selection of results from experiments on benchmark datasets. Code, additional results and videos are available on the project website [1].

5.1 Datasets

To compare with the most relevant tracking-by-detection approaches, we use the test videos and ground truth annotations from [20,24,46]. We show a sample set of frames from these videos in Figure 5. In particular, we evaluate on the following sequences from the TLD dataset [24]: Carchase, Pedestrian2, Pedestrian3, which contain challenging scenarios with pose, scale and illumination changes, full or partial occlusion, and all the videos used in [20,48,46], many of which contain motion blur, fast motion, rotation, background clutter. We will highlight some of

the most interesting cases from these datasets in the paper, and present further analysis on the project website. We note that the sequences in the tracking benchmark dataset [48], do not annotate occlusion states. For example, frames in the Coke sequence where the Coke can is completely occluded by a leaf are still annotated with a bounding box. This inconsistency in evaluation when occlusion happens was also noted by [42]. As a result, our method is evaluated unfairly, in cases where we estimate an occlusion and do not output a bounding box.

5.2 Evaluation Measures

Some of the previous works in tracking have used mean displacement error in pixels to evaluate the accuracy quantitatively. As argued in [46], this measure is not scale-invariant and is not precise for cases when a method loses track of the object. We follow [24,46] and treat an estimated object location as correct if its sufficiently overlaps with the ground truth annotation. Then we compute precision, recall and the F_1 score. In the results shown in Table 1, we use 50% as the overlap threshold.

5.3 Results

In this section we compare our approach with the state-of-the-art methods, namely TLD (2012) [24], SPLTT (2013) [46], and the winner of 2013 benchmark evaluations [48]—Struck [20]. We used the original implementation provided by the respective authors. For TLD, we set the size of the initial object bounding box as 15, since it did not run with the default value of 24 for some of the sequences.

When evaluated on all the 50 sequences from [48], our approach results in 0.657 mean F_1 score (with 50% overlap threshold), whereas Struck [20], SPLTT [46] and TLD [24] achieve 0.565, 0.661 and 0.513 respectively. We illustrate a selection of these sequences in Table 1 and Figure 5.

Our method shows a significant improvement on some of the sequences (rows 1-3 in Table 1). For the Football1 sequence (row 1, Table 1), our F_1 score is 1.000 compared to 0.554 (SPLTT). In Figure 5(a), we see that Struck (columns 2, 3: red box) tends to drift because the model is not selectively updated. SPLTT also performs poorly (column 2: yellow box, column 3: loses track) as it only relies on frame-to-frame optical flow between candidate detections computed in each frame. If either the optical flow or the detection is weak, SPLTT loses track. TLD (column 3: loses track) also uses frame-to-frame optical flow tracking and is prone to drift. In contrast, our method uses long-term tracks and updates the model selectively, which results in better performance.

For the Trellis sequence, our method shows nearly 10% improvement over Struck (0.919 vs 0.821, see row 2, Table 1). Sample frames are shown in Figure 5(b). Here, TLD is confused by the illumination changes, drifts (blue box in column 1) onto a part of the object (the face), and eventually loses track (columns 2 and 3: no blue box). This is potentially due to the weaker object model. The partial occlusions (column 2) and change in viewpoint (column 3)

Table 1. Comparison of our approach with the state-of-the-art methods using the F_1 measure (higher is better). Our approach *plain* and *occ.+vpoint* refers to variants without and with using the object state respectively.

No.	Sequence	Struck	TLD	SPLTT	Our approach	
					plain	*occ.+vpoint*
1	Football1	0.378	0.351	0.554	1.000	1.000
2	Trellis	0.821	0.455	0.701	0.838	0.919
3	Walking	0.585	0.379	0.541	0.476	0.922
4	Car4	0.404	0.003	0.314	0.401	0.398
5	Jumping	0.859	0.843	0.997	0.994	1.000
6	Suv	0.587	0.913	0.904	0.531	0.907
7	Woman	0.936	0.829	0.891	0.935	0.920
8	Coke	0.948	0.694	0.804	0.801	0.880
9	David	0.240	0.773	0.546	0.635	0.679
10	Deer	1.000	0.817	0.986	1.000	1.000
11	MotorRolling	0.146	0.110	0.128	0.134	0.512
12	MountainBike	0.908	0.355	0.908	1.000	1.000
13	Pedestrian2	0.175	0.500	0.950	0.107	0.979
14	Pedestrian3	0.353	0.886	0.989	0.424	1.000
15	Carchase	0.036	0.340	0.290	0.098	0.312

lead to incorrect model updates, and thus poorer results for SPLTT (yellow box) and Struck (red box). Our method (green box) estimates the state of the object (occlusion or change in viewpoint) and performs a correct update step. For the Walking sequence, we achieve an F_1 score of 0.922 compared to 0.585 from Struck (row 3, Table 1), since our tracks adapt to changes in object size (with the help of long-term tracks).

The performance of our method is comparable on some sequences (rows 4-6 in Table 1). For example, an F_1 score of 0.398 compared to 0.404 (Struck) for the Car4 sequence (see row 4, Table 1). In a few cases, our method performs worse than the trackers we compare with. For example, on the Freeman4 sequence, our method fails to track the object (0.004 F_1 score). Struck, TLD and SPLTT perform better than this (0.177, 0.134 and 0.145 respectively), but are still significantly inferior to their average performance on the entire benchmark dataset. As shown in Figure 5(c), none of the methods show a noteworthy performance, and drift or miss the object often. We observed that the minimum size of our detector was not ideal to find the object in this sequence, which is only 15×16 pixels large. All the trackers also perform poorly on the Soccer sequence—0.166 is the best performance, which is comparable to our score, 0.143. In Figure 5(d) we see that the player's face in this sequence is tracked initially, but due to severe motion blur, fails in the latter frames.

In Figure 5(e) we show sample frames from the Woman sequence, where our method identifies that the object is occluded (column 2). Due to the lack of occlusion labelling in the ground truth annotation, our method is penalized for

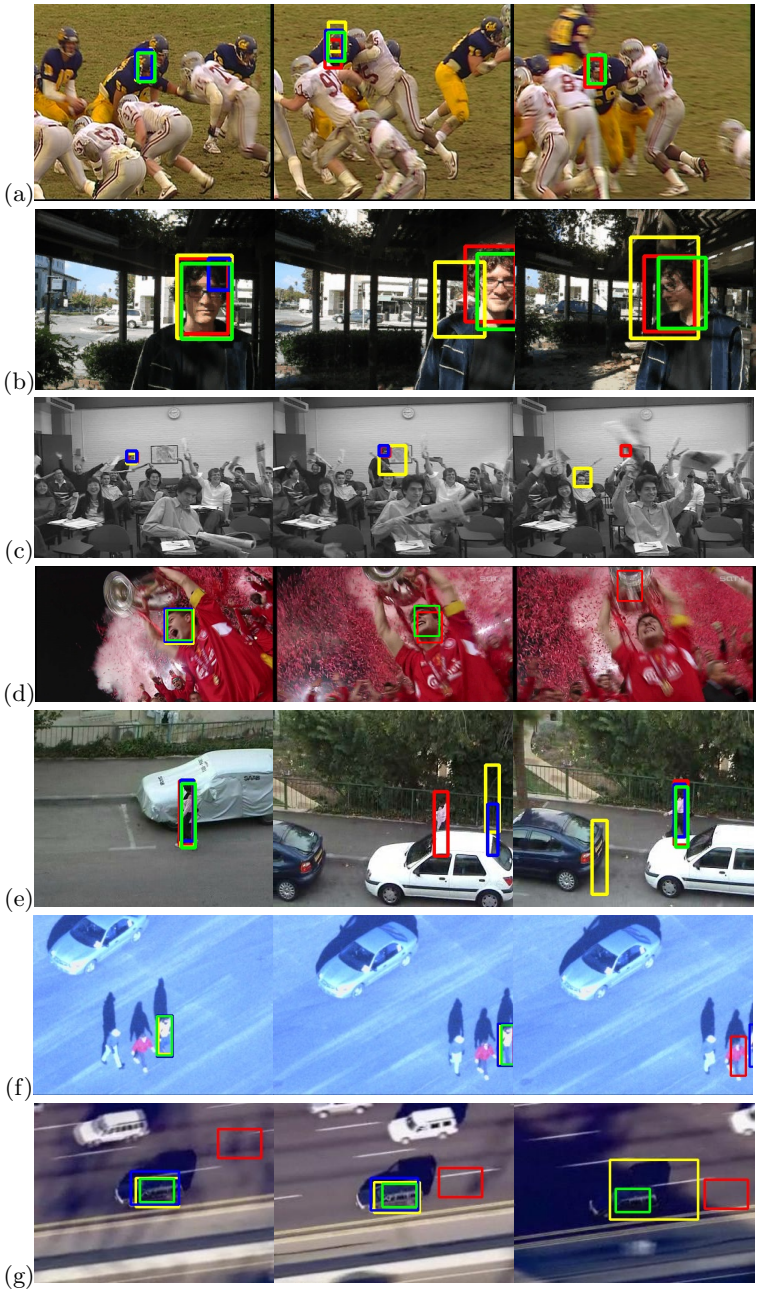


Fig. 5. Tracking results on (a) Football1, (b) Trellis, (c) Freeman4, (d) Soccer, (e) Woman sequences from the benchmark dataset [48], and (f) Pedestrian2, (g) Carchase sequences from the TLD dataset [24]. Green: Our result, Red: Struck, Yellow: SPLTT, Blue: TLD. See text for details. (Best viewed in pdf.)

frames where we estimate occlusion, and hence our result is slightly worse (0.920 vs 0.936 (Struck), shown in row 7, Table 1). The Coke sequence (row 8, Table 1) is another such case, where our method (0.87) performs significantly better than TLD (0.69) and SPLTT (0.80), but is inferior to Struck (0.95). Results on the Pedestrian2 and Carchase long-term sequences, in Figures 5(f) and 5(g), show that Struck cannot handle cases where the object re-enters the field of view after occlusion, unlike our method.

Discussion. Table 1 also shows a component-level evaluation of our method. Estimating the state of the object improves the performance in most cases (e.g., row 3). In some cases we observe a slight decrease in performance over the plain vanilla method (e.g., row 7) due to lack of occlusion labelling in the ground truth annotation (see text for Figure 5(e) above).

Note that long-term tracks are used as an additional information in our work. If there are insufficient point tracks within the bounding box (< 10), we do not estimate the state, and continue in a tracking-by-detection mode. For estimating the object state, we observed two cases. (1) Object and camera motion: In this case, tracks from [45] do not suffer from significant drift as they tend to be relatively short in length. For example, on the Deer sequence (71 frames), the average length of the track is 10.1, and less than 10% of tracks drift. This does not affect our state estimation. (2) Object or camera motion only: Here, tracks can drift, and then result in incorrect occlusion estimates (e.g., Crossing sequence: 120 frames; average track length 77, 50% drift). In the worst case, our tracker predicts full occlusion and misses the object for a few frames, but recovers when the detector is run over the entire image to overcome this occlusion state. In essence, failures in long-term tracks have a limited impact on our system overall. However, a limiting case of our approach is when an object undergoes occlusion, and re-appears in a viewpoint which has not been seen before the occlusion (i.e., no template is learned).

Computation time of our method depends on the image size and the number of tracks in the sequence. For sequences in Table 1, it takes 6.7s/frame on average, with our unoptimized Matlab code (which does not include time to precompute optical flow – 3.4s/frame on GPU).

Acknowledgements. This work was supported in part by the MSR-Inria joint project, the European integrated project AXES and the ERC advanced grant ALLEGRO.

References

1. `http://lear.inrialpes.fr/research/tracking`
2. Avidan, S.: Ensemble tracking. PAMI 29(2), 261–271 (2007)
3. Babenko, B., Yang, M.H., Belongie, S.: Robust object tracking with online multiple instance learning. PAMI 33(8), 1619–1632 (2011)
4. Badrinarayanan, V., Pérez, P., Le Clerc, F., Oisel, L.: Probabilistic color and adaptive multi-feature tracking with dynamically switched priority between cues. In: ICCV (2007)

5. Birchfield, S.: Elliptical head tracking using intensity gradients and color histograms. In: CVPR (1998)
6. Boykov, Y., Jolly., M.P.: Interactive graph cuts for optimal boundary & region segmentation of objects in n-d images. In: ICCV (2001)
7. Brox, T., Malik, J.: Object segmentation by long term analysis of point trajectories. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part V. LNCS, vol. 6315, pp. 282–295. Springer, Heidelberg (2010)
8. Brox, T., Malik, J.: Large displacement optical flow: Descriptor matching in variational motion estimation. PAMI 33(3), 510–513 (2011)
9. Rother, C., Kolmogorov, V., Blake, A.: Grabcut: Interactive foreground extraction using iterated graph cuts. ACM Trans. Graphics (2004)
10. Collins, R.T., Liu, Y., Leordeanu, M.: Online selection of discriminative tracking features. PAMI 27(10), 1631–1643 (2005)
11. Comaniciu, D., Ramesh, V., Meer, P.: Kernel-based object tracking. PAMI 25(5), 564–577 (2003)
12. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: CVPR (2005)
13. Du, W., Piater, J.: A probabilistic approach to integrating multiple cues in visual tracking. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part II. LNCS, vol. 5303, pp. 225–238. Springer, Heidelberg (2008)
14. Everingham, M., Sivic, J., Zisserman, A.: Taking the bite out of automatic naming of characters in TV video. Image and Vision Computing 27(5) (2009)
15. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The Pascal Visual Object Classes (VOC) Challenge. IJCV 88(2), 303–338 (2010)
16. Fan, R.E., Chang, K.W., Hsieh, C.J., Wang, X.R., Lin, C.J.: LIBLINEAR: A library for large linear classification. JMLR 9, 1871–1874 (2008)
17. Felzenszwalb, P.F., Girshick, R.B., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part based models. PAMI 32(9), 1627–1645 (2010)
18. Grabner, H., Leistner, C., Bischof, H.: Semi-supervised on-line boosting for robust tracking. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part I. LNCS, vol. 5302, pp. 234–247. Springer, Heidelberg (2008)
19. Hammer, P.L.: Some network flow problems solved with pseudo-boolean programming. Operations Research 13, 388–399 (1965)
20. Hare, S., Saffari, A., Torr, P.H.S.: Struck: Structured output tracking with kernels. In: ICCV (2011)
21. Hartley, R.I., Zisserman, A.: Multiple View Geometry in Computer Vision, 2nd edn. Cambridge University Press (2004)
22. Isard, M., Blake, A.: ICONDENSATION: Unifying low-level and high-level tracking in a stochastic framework. In: Burkhardt, H.-J., Neumann, B. (eds.) ECCV 1998. LNCS, vol. 1406, pp. 893–908. Springer, Heidelberg (1998)
23. Jepson, A.D., Fleet, D.J., Maraghi, T.F.E.: Robust online appearance models for visual tracking. PAMI 25(10), 1296–1311 (2003)
24. Kalal, Z., Mikolajczyk, K., Matas, J.: Tracking-learning-detection. PAMI 34(7), 1409–1422 (2012)
25. Kolmogorov, V., Zabih, R.: What energy functions can be minimized via graph cuts? PAMI 26(2), 147–159 (2004)
26. Lee, K., Ho, J., Yang, M., Kriegman, D.: Visual tracking and recognition using probabilistic appearance manifolds. CVIU 99(3), 303–331 (2005)
27. Leibe, B., Schindler, K., Cornelis, N., van Gool, L.: Coupled object detection and tracking from static cameras and moving vehicles. PAMI 30(10), 1683–1698 (2008)

28. Lezama, J., Alahari, K., Sivic, J., Laptev, I.: Track to the future: Spatio-temporal video segmentation with long-range motion cues. In: CVPR (2011)
29. Liu, B., Huang, J., Kulikowski, C., Yang, L.: Robust visual tracking using local sparse appearance model and k-selection. PAMI 35(12), 2968–2981 (2013)
30. Lucas, B., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: IJCAI (1981)
31. Malisiewicz, T., Gupta, A., Efros, A.: Ensemble of exemplar-svms for object detection and beyond. In: ICCV (2011)
32. Matthews, I., Ishikawa, T., Baker, S.: The template update problem. PAMI 26(6), 810–815 (2004)
33. Mei, X., Ling, H.: Robust visual tracking and vehicle classification via sparse representation. PAMI 33(11), 2259–2272 (2011)
34. Moreno-Noguer, F., Sanfeliu, A., Samaras, D.: Dependent multiple cue integration for robust tracking. PAMI 30(4), 670–685 (2008)
35. Ochs, P., Malik, J., Brox, T.: Segmentation of moving objects by long term video analysis. PAMI 36(6), 1187–1200 (2014)
36. Pang, Y., Ling, H.: Finding the best from the second bests - inhibiting subjective bias in evaluation of visual tracking algorithms. In: ICCV (2013)
37. Park, D.W., Kwon, J., Lee, K.M.: Robust visual tracking using autoregressive hidden Markov model. In: CVPR (2012)
38. Pérez, P., Vermaak, J., Blake, A.: Data fusion for visual tracking with particles. Proc. IEEE 92(3), 495–513 (2004)
39. Platt, J.C.: Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In: NIPS (1999)
40. Ramanan, D., Forsyth, D., Zisserman, A.: Tracking people by learning their appearance. PAMI 29(1), 65–81 (2007)
41. Ross, D.A., Lim, J., Lin, R., Yang, M.: Incremental learning for robust visual tracking. IJCV 77(1), 125–141 (2008)
42. Song, S., Xiao, J.: Tracking revisited using RGBD camera: Unified benchmark and baselines. In: ICCV (2013)
43. Spengler, M., Schiele, B.: Towards robust multi-cue integration for visual tracking. Machine Vis. App. 14, 50–58 (2003)
44. Stenger, B., Woodley, T., Cipolla, R.: Learning to track with multiple observers. In: CVPR (2009)
45. Sundaram, N., Brox, T., Keutzer, K.: Dense point trajectories by GPU-accelerated large displacement optical flow. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part I. LNCS, vol. 6311, pp. 438–451. Springer, Heidelberg (2010)
46. Supancic, J.S., Ramanan, D.: Self-paced learning for long-term tracking. In: CVPR (2013)
47. Wu, B., Nevatia, R.: Detection and tracking of multiple, partially occluded humans by Bayesian combination of edgelet based part detectors. IJCV (2007)
48. Wu, Y., Lim, J., Yang, M.H.: Online object tracking: A benchmark. In: CVPR (2013)
49. Yilmaz, A., Javed, O., Shah, M.: Object tracking: A survey. ACM Comput. Surv. 38(4) (2006)