This ICCV paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the version available on IEEE Xplore.

Online Object Tracking with Proposal Selection

Yang Hua   Karteek Alahari   Cordelia Schmid

Inria*

Abstract

Tracking-by-detection approaches are some of the most successful object trackers in recent years. Their success is largely determined by the detector model they learn initially and then update over time. However, under challenging conditions where an object can undergo transformations, e.g., severe rotation, these methods are found to be lacking. In this paper, we address this problem by formulating it as a proposal selection task and making two contributions. The first one is introducing novel proposals estimated from the geometric transformations undergone by the object, and building a rich candidate set for predicting the object location. The second one is devising a novel selection strategy using multiple cues, i.e., detection score and edgeness score computed from state-of-the-art object edges and motion boundaries. We extensively evaluate our approach on the visual object tracking 2014 challenge and online tracking benchmark datasets, and show the best performance.

1. Introduction

Over the past few years, the tracking-by-detection framework has emerged as one of the successful paradigms for tracking objects [33, 37, 44, 50]. It formulates the tracking problem as the task of detecting an object, which is likely to undergo changes in appearance, size, or become occluded, over time [3, 4, 17, 18, 19, 26, 31, 41, 47, 49]. These approaches begin by training an object detector with an initialization (in the form of a bounding box) in the first frame, and then update this model over time. Naturally, the choice of exemplars used to update and improve the object model is critical [23, 47, 51].

Consider the *motocross* example shown in Figure 1. Here, the target (biker and his motorbike) undergoes several deformations as the biker performs an acrobatics routine, which leads to significant changes in the aspect ratio as well as the rotation angle of the bounding box. State-of-the-art tracking methods, e.g., [23, 47], rely on axis-aligned bounding boxes and thus are ill-equipped to capture the accurate extents of the object in such scenarios. In this paper, we aim to address this issue by treating the tracking problem as the task of selecting the best proposal containing the object of interest from several candidates. To this end, we propose a novel method to generate candidates which capture the extents of the object accurately, by estimating the transformation(s) undergone by the object.

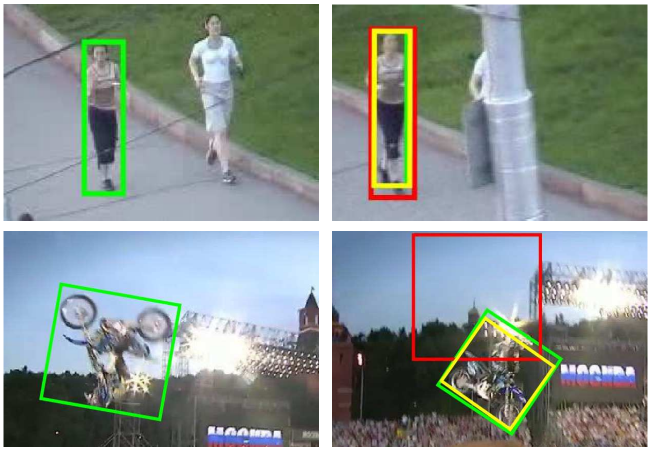
Figure 1. Sample frames (cropped) from the *jogging* (top row) and *motocross* (bottom row) sequences [33]. The ground truth annotation (green) in the first frame (left) is used to train our tracker and the winner [12] of VOT2014 challenge [33]. We show these two tracking results (right) on another frame in the sequence. Our method (yellow) successfully tracks objects undergoing deformations unlike [12] (red). *Best viewed in pdf.*

The focus of this paper is the problem of tracking a single target in monocular video sequences. The target is provided as a ground truth annotation in the first frame, as in a standard tracking-by-detection setup [19, 26]. We learn an object detector model from this annotation and evaluate it in subsequent frames to propose candidate locations that are likely to contain the target. While these proposals are sufficient to track objects undergoing a small subset of transformations over time, such as translation shown in Figure 1-top, they cannot handle generic transformations, e.g., similarity transformation shown in Figure 1-bottom. We address this problem by: (i) introducing novel additional proposals to improve the candidate location set, and (ii) using multiple cues to select the proposal that is most likely to contain the target. Additional proposals are computed

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

by robustly estimating the parameters of similarity transformation (i.e., scaling, translation, rotation) that the target is likely to have undergone, with a Hough transform voting scheme on the optical flow. With these parameters, we propose several candidate locations for the target. Thus, for every frame of the video sequence, our candidate set consists of object detector boxes as well as geometry estimation proposals. Note that state-of-the-art tracking-by-detection approaches are limited to detector proposals alone. The second contribution we make is in selecting the best proposal from the candidate set using multiple cues – objectness scores [53] computed with edge responses [13] and motion boundaries [48], and the detection score.

We evaluate the performance of our approach exhaustively on two recent benchmark datasets, namely the visual object tracking (VOT) 2014 challenge dataset [33] and the online tracking benchmark (OTB) [50]. Furthermore, we compare with all the 38 trackers evaluated as part of the VOT2014 challenge, and with the best performers [19, 25, 52] among the 29 methods analyzed in the OTB comparison paper [50]. Our method shows the top performance on both these challenging datasets, and is in particular 29.5% better than the current leader [12] of the VOT2014 challenge.

The remainder of the paper is organized as follows. In Section 2 we discuss related work. The details of our novel proposals and the multiple cues to select the best candidate are explained in Section 3. Section 4 discusses the implementation details of the methods. In Section 5 we present an extensive evaluation of the proposed method and compare it with the state of the art. This section also discusses the datasets and evaluation protocols. Concluding remarks are made in Section 6.

2. Related Work

Two of the key elements of any tracking algorithm are, how the object of interest is represented, and how this representation is used to localize it in each frame. Several methods have been proposed on these two fronts. Object representation has evolved from classical colour histograms [10] to models learned from generative [30, 35, 42] or discriminative [3, 9, 19, 23] approaches. In the context of localizing the object given a representation, methods such as [5, 6, 24, 38, 39, 45, 46] have incorporated multiple cues and fused their results with Markov chain Monte Carlo [38] or particle filtering [24]. More recently, inspired by the success of object detection algorithms [14, 16], the tracking-by-detection approach [3, 4, 19, 26, 31, 41, 47, 51] has gained popularity. In fact, methods based on this paradigm are ranked among the top performers on evaluation benchmarks [37, 50].

Tracking-by-detection methods learn an initial discriminative model of the object from the first frame in the sequence (e.g., with a support vector machine (SVM) [19, 47]), evaluate it to detect the most likely object location in subsequent frames, and then update the object model with these new detections. *Struck* [19] is an interesting variant of this general framework, using a structured output formulation to learn and update the detector. This shows state-of-the-art results in several scenarios, but lacks the capability of handling severe occlusions and suffers from drift, as discussed in [23].

To avoid the well-known problem of drift [34], some methods explicitly detect tracking failures and occlusions with heuristics [52], combine detector- and tracker-based components [26], control the training set with learning [47], or maintain a pool of trackers [23, 29, 51]. Kalal et al. [26] combine predictions from a variant of the Lukas-Kanade tracker and an incrementally updated detector. More specifically, the results of the tracker provide training data to update the detector model, and the detector re-initializes the tracker when it fails. This interleaving of tracker and detector, although interesting, is restrictive as the object model is a set of fixed-size template patches, and thus, not robust to severe changes in object size.

Supancic and Ramanan [47] propose a self-paced learning scheme to select training examples for updating the detector model. A pre-fixed number of (top k) frames are chosen carefully according to an SVM objective function to avoid corrupting the training set. In other words, only frames with confident detections are used to update the model. This approach, however, lacks the ability to handle commonly-occurring scenarios where the object is undergoing a geometric transformation, such as the rotation shown in Figure 1-bottom. Hua et al. [23] address this by using long-term trajectories to estimate the transformations undergone by the object. However, their approach is: (i) inherently offline, requiring all the frames in the video sequence to track, (ii) not robust to significant changes in scale, (iii) maintains a pool of trackers, all of which need to be evaluated independently in every frame.

Despite the general success of tracking-by-detection approaches, they have an important shortcoming. They all rely solely on a detector, and are thus unable to cope with transformations an object can undergo. On the other hand, key point based trackers like CMT [36] detect key points in a frame, match them to the next frame and can estimate transformations, such as changes in scale and rotation. However, they are highly sensitive to the success of key point detection algorithms and lack a learned model. Indeed, this is evident from the poor performance on the VOT dataset (see Section 5.2). The main focus of this paper is to address the gap between these two paradigms. Inspired by successful object proposal methods for segmentation and detection [2, 8], we pose the tracking problem as the task of finding the best proposal from a candidate set. In contrast to previous work, we introduce novel proposals estimated from the geometric transformations undergone by the object of interest, and build a rich candidate set for predicting the object location, see Figure 2. In essence, our candidate set consists of proposals from the detector as well as the geometry estimation model. We then devise a novel scoring mechanism using standard object detector score, together with object edgeness [53] computed from state-of-the-art edge detector SED [13] and motion boundary detector [48] to find the most likely location of the object.

3. Proposal Selection for Tracking

The main components of our framework for online object tracking are: (i) learning the initial detector, (ii) building a rich candidate set of object locations in each frame, consisting of proposals from the detector as well as the estimated geometric transformations, (iii) evaluating all the proposals in each frame with multiple cues to select the best one, and (iv) updating the detector model. We now present all these components and then provide implementation details in Section 4.

3.1. Initial detector

We learn the initial detector model with a training set consisting of one positive sample, available as a bounding box annotation in the first frame, and several negative bounding box samples which are automatically extracted from the entire image. We use HOG features [11, 16] computed for these bounding boxes and learn the detector with a linear SVM, similar to other tracking-by-detection approaches [23, 47]. The detector is then evaluated on subsequent frames to estimate the candidate locations of the object. Rather than make a suboptimal decision by choosing the best detection as the object location in a frame, we extract the top k detections and build a candidate set. We augment this set with proposals estimated from the transformations undergone by the object, as described in the following section.

3.2. Estimating geometry and proposals

In the examples shown in Figure 2, the object of interest is undergoing a geometric transformation (rotation in these examples). None of the top detector proposals can capture the object accurately in this case. To address this issue, we explicitly estimate the transformation undergone by the object and then use it to enrich the candidate set with additional geometric proposals.

We represent the geometric transformation with a similarity matrix [20]. Note that other transformations like homography can also be used. The similarity transformation is defined by four parameters – one each for rotation and scale, and two for translation. In this work, we estimate them with a Hough transform voting scheme using frame-to-frame optical flow correspondences. We begin by computing the optical flow between frames $t-1$ and t with a state-of-the-art flow estimation method [7]. The pixels within the object bounding box with flow values in frame $t-1$ then give us corresponding pixels in frame t. With a pair of these matches, given by two pixels in $t-1$ which are sufficiently distant from each other, we estimate the four scalar parameters in the similarity matrix [20]. Every choice of a pair of corresponding matches gives us an estimate of the four parameters. We then use the Hough transform [22], wherein each pair of point-matches votes for a (4D) parameter set, to find the top k consistent parameter sets. This voting scheme allows us to filter out the incorrect correspondences due to common errors in optical flow estimation.

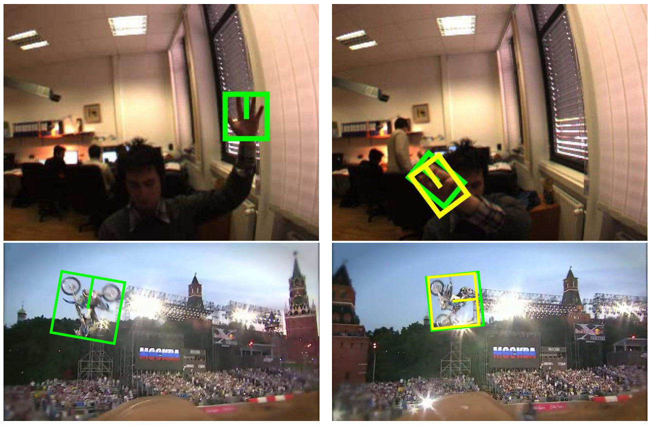

Figure 2. Two sequences from the VOT dataset [33]. In each row, we show the ground truth (in green) in the first frame (left) and the best geometry proposal in a subsequent frame (in yellow in the right image). We show a vertical line on each box to visualize the rotation angle with respect to image edges.

To sum up, we estimate the k most likely geometric transformations undergone by the object as the parameters with the top k votes. Each one of these transformations results in a candidate location of the object in t, by applying the corresponding similarity matrix to the object bounding box in $t-1$. Before adding all these k proposals to the candidate set, we perform an additional filtering step to ensure further robustness. To this end, we discard all the proposals: (i) which have a low confidence, given by the number of votes, normalized by the total number of point-matches, and (ii) those where the estimated angle is significantly different from estimations in earlier frames. We determine both these threshold values with Kalman filtering [27]. A threshold τ_t, used for filtering proposals in frame $t+1$, is given by:

$$\begin{aligned} \tau_t &= \alpha\tau_{t-1} + k_t(d_t - \alpha\beta\tau_{t-1}), && (1)\\ k_t &= (p_{t-1}+q)/(p_{t-1}+q+r), && (2)\\ p_t &= (1-k_t)(p_{t-1}+q), && (3) \end{aligned}$$

where α, β, q, r are scalar parameters set empirically (see Section 4), k_t is the Kalman gain, p_t is the error estimate. The filter is initialized with $\tau_0 = 0$, $p_0 = 0.1$. Here, d_t is either the confidence score or the estimated angle of the selected proposal in frame t to determine the respective threshold.

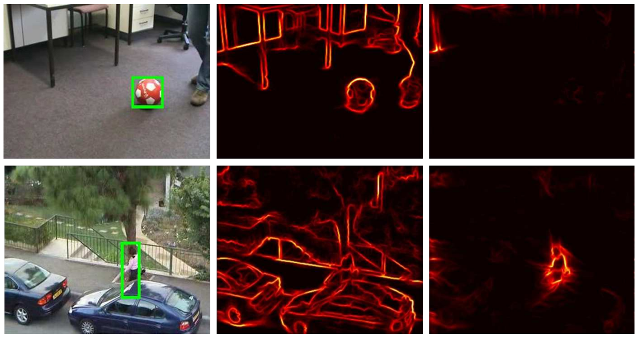

Figure 3. Two examples of edge and motion boundary responses. In each row, from left to right, we show the original image (with the object in a green box), edges and motion boundaries. The edges provide a stronger response in the example in the first row while motion boundaries are more effective in the example in the second row.

3.3. Selecting proposals

At the end of the geometry estimation step, we have k proposals from the detector and (at most) k proposals from the similarity transformation in frame t. Now, the task is to find the *best* proposal most likely to contain the object in this frame. We use three cues, detection confidence score, objectness measures computed with object edges and motion boundaries, for this task. We first use the (normalized) detection confidence score computed for each proposal box with the SVM learned from the object annotation in the first frame and updated during tracking. This provides information directly relevant to the object of interest in a given sequence.

In cases where the detection scores of all the candidate boxes are statistically similar, we use cues extracted from object edges [13] and motion boundaries [48] in the image, referred to as edgeness scores. In other words, when the detection scores are inconclusive to choose the best proposal, we rely on edgeness measure. Here, all the top candidates contain the object of interest to some extent, and the task is to choose the one that best contains the object—a scenario well-suited for edgeness cue because a proposal box which accurately localizes the entire object has a stronger edge response compared to a box which overlaps partially with the object, thus resulting in a weaker edge score. The edgeness score is given by the number of edges (or motion boundaries) within a proposal box after discarding contours that intersect with the box's boundary, as in [53]. We compute edgeness scores with edges and motion boundaries separately and use the one with a stronger response. As shown in the examples in Figure 3, these two cues are complementary. Then, the proposal with the best edgeness score is selected.

We follow this approach, rather than using a combined score, e.g., a weighted combination of SVM and edgeness scores, because setting this weight manually is suboptimal, and learning it is not possible due to the lack of standard training-validation-test datasets for tracking. In contrast, our approach uses object-specific SVM and generic edgeness scores effectively, and circumvents the need for a manually set weight to combine scores.

3.4. Updating the detector

Having computed the best proposal containing the object, box_t, we use it as a positive exemplar to learn a new object model. In practice, we update the current model incrementally with this new sample; see Section 4. This new detector is then evaluated in frame $t+1$, to continue tracking the object. Algorithm 1 summarizes our overall approach.

4. Implementation Details

Detector. The initial detector is trained with the one positive sample (provided as ground truth box in the first frame) and several negative examples extracted from the frame. We harvest bounding boxes that overlap less than 50% with the ground truth for the negative set. The regularization parameter in the SVM cost function is set to 0.1 in all our experiments. This cost function is minimized with LIBLINEAR [15]. We perform three rounds of hard negative mining to refine the detector. The detector scores are calibrated with Platt's method [40] using jittered versions of the positive sample (100 positive and negative samples each). For the VOT dataset, an additional step is introduced to train with rotated bounding box annotations. We estimated the rotation angle between one edge of the box and the corresponding image axis,[1] rectified the image with this angle, and extracted an axis-aligned box as the positive sample.

The detector was evaluated at seven scales: $\{0.980, 0.990, 0.995, 1.000, 1.005, 1.010, 1.020\}$, in every frame. This is done densely in each scale, with a step size of 2 pixels. In the rotated bounding box case, we rectify the image with the angle estimated during training, before evaluating the detector. We set $k = 5$ to select the top 5 detection results and also (at most) 5 geometry proposals to form the candidate set. The best proposal box_t selected from all the candidates in a frame is used to update the detector with a warm-start scheme [15, 47]. We then perform hard negative mining and calibrate the scores to complete the detector update step. For computational efficiency, the detector is evaluated in a region around the previously estimated object location, instead of the entire image.

Hough voting. The geometric transformation is estimated only when the mean ℓ_2 norm of optical flow in box_{t-1} is larger than 0.5. This ensures that there is sufficient motion in the frame to justify estimating a transformation. We use two pixels in $t-1$ and their corresponding points in t to vote for the geometric transformation parameters. In practice,

[1] Note that this estimation was done only in the first frame or whenever the tracker is restarted after a failure, i.e., when we have access to the ground truth box.

```
Algorithm 1 : Our approach for online object tracking.
Data: Image frames $1 \dots n$, Ground truth $\text{box}_1$ in frame 1
Result: Object location $\text{box}_t$ in frames $t = 2 \dots n$
Learn initial detector model in frame 1 (§3.1)
for $t = 2 \dots n$ do
    candidate set $\mathcal{C}_t \leftarrow$ Top-$k$ detections in frame $t$ (§3.1)
    $\mathcal{H}_t \leftarrow$ Geometry proposals in frame $t$ (§3.2)
    $\mathcal{C}_t \leftarrow \mathcal{C}_t \cup \mathcal{H}_t$
    $\text{box}_t \leftarrow$ Best proposal in $\mathcal{C}_t$ (§3.3)
    Update detector model with $\text{box}_t$ (§3.4)
end
```

pixels that are at least 25 pixels apart from each other in $t-1$ were chosen randomly for a reliable estimation. To aggregate the votes, a pseudo-random hash function of bin values is used to vote into a one-dimensional hash table, instead of a four-dimensional array, similar to [32]. The bin size for scale is set to 0.1, and 2.0 for angle and the two translation parameters. Finally, we take the least-squares solution of all the candidates that vote for a particular bin, making our geometry estimation further robust.

Pruning proposals. The parameters of the Kalman filter to discard weak geometry proposals are set as: $\alpha = 1$, $\beta = 1$, process error variance $q = 0.001$, and measurement error variance $r = 0.01$ in all our experiments. To find the best proposal from a candidate set, we propose two cues: detection and edgeness scores. Edgeness scores are used to determine the best proposal when the detection scores are inconclusive, i.e., when one or more proposals differ from the best detection score by at most 1%. However, edgeness scores can often be corrupted by noise, e.g., low-quality images that are common in benchmark datasets. We ensure that these scores are used only when they are comparable to the mean score of the previous 5 frames, i.e., the difference between the highest edgeness score and the mean is less than twice the variance. This filtering step is performed separately for edgeness computed with object edges and motion boundaries.

5. Experiments

We now present our empirical evaluation on two state-of-the-art benchmark datasets and compare with several recent methods. The results of all the methods we compare with are directly taken from the respective toolkits. Additional results and videos are provided on the project website [1].

5.1. Datasets and Evaluation

Much attention has been given to developing benchmark datasets and measures to evaluate tracking methods in the past couple of years [43]. For example, Wu et al. [50] and Song and Xiao [44] introduced benchmark datasets and evaluated several state-of-the-art tracking algorithms. Kristan et al. [33] have organized visual object tracking (VOT) challenges in 2013 and 2014, where several challenging sequences were used to evaluate methods. We have evaluated our method on the VOT2014 [28] and the online tracker benchmark (OTB) [50] datasets, following their original protocols.

VOT2014 dataset. As part of the visual object tracking challenge held at ECCV 2014 [33], the organizers released a set of 25 challenging video sequences. They were chosen from a candidate set of 394 sequences, composed of examples used in several previous works such as VOT2013 benchmark dataset, ALOV [43], OTB [50], as well as a few unpublished ones. This choice was made by: (i) discarding sequences shorter than 200 frames, and (ii) ensuring that the selected sequences contain well-defined targets and represent challenging scenarios, e.g., clutter, object deformation, change in aspect ratio. The frames in all the sequences are manually annotated. In a departure from annotations in other datasets, where the bounding box enclosing the target is axis-aligned, this benchmark uses rotated boxes in order to handle targets that are deforming, rotating or elongated, as shown in Figure 2. This new annotation makes the dataset a more challenging setting to evaluate trackers. In addition to this, all the frames are labelled with visual attributes: occlusion, illumination change, object motion, object size change, camera motion and neutral.

We follow the protocol described in [28] to evaluate our tracking algorithm, and compare it with several state-of-the-art trackers. This evaluation scheme uses accuracy and robustness measures to compare trackers. Accuracy is computed as the mean intersection over union score with the ground truth bounding box over the entire sequence (while discarding a few frames immediately following a tracking failure), and robustness is the number of times the tracker has failed. A tracking failure is signalled in a frame s if the predicted box does not overlap with the ground truth annotation. In this case, the tracker is restarted from scratch with the ground truth annotation in frame $s+5$.

The scores are averaged over several runs of the tracker to account for any stochastic behaviour. The overall rank of a tracker is determined by first ranking trackers on accuracy and robustness separately on each attribute set, then averaging the rank over the attributes, and then taking the mean rank over the two performance measures. If a set of trackers show similar performance (which is measured with statistical significance and practical difference tests), their ranks are averaged. This rank correction is performed separately for accuracy and robustness, and the overall average rank is recomputed as mentioned above. We used the toolkit[2] provided by the organizers of the challenge to compute the scores and the ranking, allowing us to compare with 38 methods. The interested reader can find more details of the evaluation protocol in [28].

[2] http://www.votchallenge.net/howto/perfeval.html

No.	Method	Acc.	Robust.	Avg.
1	Our-ms-rot	**6.07**	**8.58**	**7.33**
2	Our-ms	**4.73**	10.13	**7.43**
3	DSST [12]	6.78	13.99	**10.39**
4	SAMF	6.46	15.65	11.06
5	DGT	12.67	10.13	11.4
6	KCF [21]	**6.16**	16.71	11.44
7	PLT_{14}	16.04	**6.98**	11.51
8	PLT_{13}	19.74	**4.00**	11.87
9	eASMS	15.37	15.1	15.24
10	Our-ss	16.11	14.47	15.29
11	ACAT	15.1	16.78	15.94
12	MatFlow	23.82	9.67	16.74
13	HMMTxD	11.08	22.51	16.8
14	MCT	18.47	15.14	16.81
15	qwsEDFT	19.06	21.15	20.11
16	ACT	22.68	18.1	20.39
17	ABS	22.34	20.49	21.42
18	VTDMG	23.22	19.94	21.58
19	LGTv1	31.05	12.68	21.87
20	BDF	24.92	19.39	22.15

Table 1. Performance of the top 20 methods on the VOT2014 dataset. We show the ranking on accuracy (Acc.) and robustness (Robust.) measures, as well as the overall rank (Avg.). The top performer in each measure is shown in red, and the second and third best are in blue and green respectively. We show three variants of our method – Our-ms-rot: uses multiscale detector and geometry proposals, Our-ms: uses only multiscale detector proposals, and Our-ss: uses only single-scale detector proposals. The performance of other methods from the VOT2014 challenge is shown on the project website [1].

OTB dataset. The online tracking benchmark dataset [50] is a collection of 50 of the most commonly used tracking sequences, which are fully annotated. It contains sequences where the object varies in scale, has fast motion, or is occluded. The tracking performance on this benchmark is evaluated with a precision score and area under the curve (AUC) of a success plot, as defined in [50]. Precision score is measured as the percentage of frames whose predicted object location (center of the predicted box) is within a distance of 20 pixels from the center of the ground truth box. Precision plots are also computed by varying the distance threshold between 0 and 50. Success plots show the percentage of frames whose intersection over union overlap with the ground truth annotation is over a threshold varying between 0 and 1. We used the toolkit[3] provided by Wu et al. [50] to generate the plots and compute the scores.

5.2. Results

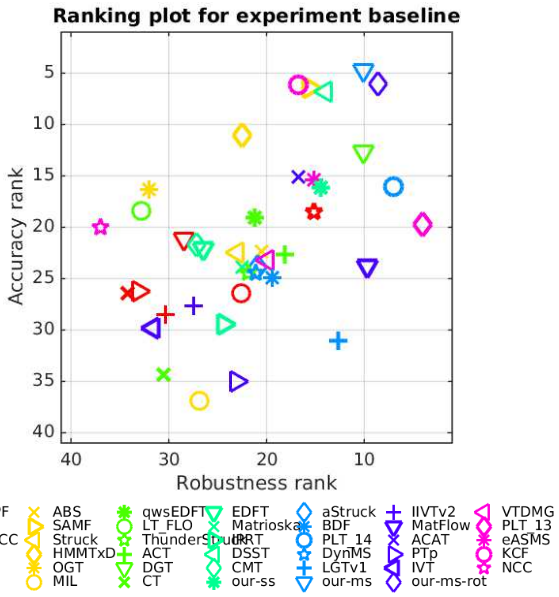

Figure 4. AR ranking plot of all the trackers from the VOT2014 challenge as well as our approaches. Trackers close to the top right corner of the plot are among the top performers. Our multiscale approaches ("Our-ms-rot" and "Our-ms") are closest to the top-right corner compared to all the other methods.

VOT. The results in Table 1 correspond to the baseline experiment in the toolkit, where each tracker is executed 15 times with the same ground truth annotation in the first frame, and the overall rank is computed as described in Section 5.1. Overall, our proposal selection method using detector and geometry proposals ("Our-ms-rot" in the table) performs significantly better than DSST [12], the winner of the VOT2014 challenge, with an average rank of 7.33 vs 10.39. We also evaluated two variants of our approach. The first one "Our-ms" uses only the multiscale detector proposals and the second variant "Our-ss" is limited to proposals from the detector evaluated only at a single scale. The overall rank of these two variants is lower than our full method: 7.43 and 15.29 respectively. The variant based on multiscale detector proposals performs better on the accuracy measure compared to the full method, but is significantly worse on the robustness measure. In other words, this variant fails on many more frames than the full method, and benefits from the resulting reinitializations. Due to significant changes in scale that occur in several frames in the dataset, the variant based on a single-scale detector performs rather poorly. CMT [36], the key point based tracker which estimates rotation and scale of the object, has an average rank of 24.43 and is ranked 28th on the list (see [1]). With the evaluation protocol of reinitializing the tracker when it fails, we found that using edgeness measure did not show a significant difference compared to the detection score. Thus, to keep the comparison with respect to the large number of trackers (38) reasonable, we show a subset of variants of our methods on VOT.

On the accuracy measure, our approaches ("Our-ms-rot"

[3] https://sites.google.com/site/trackerbenchmark/benchmarks/v10

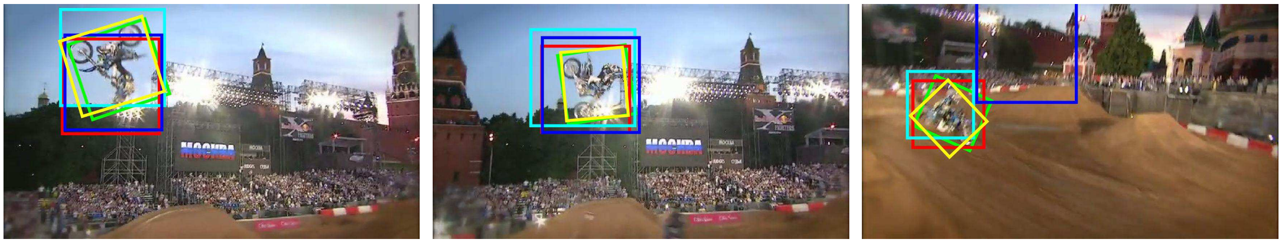

Figure 5. Sample result on the *motocross* sequence. We compare with several state-of-the-art trackers. Ground truth: green, Our result: yellow, DSST [12]: red, PLT_{14}: cyan, Struck [19]: blue. *Best viewed in pdf.*

and "Our-ms") are ranked first and second. KCF [21] is third, with SAMF and DSST [12] also performing well. The AR ranking plot in Figure 4 shows the similar performance of KCF, SAMF and DSST, where they form a cluster. KCF is a correlation filter based tracker. It is essentially a regression method trained with HOG features computed on several patches densely sampled around the object. This method, however, is significantly inferior on robustness measure (16.71 compared to our result 8.58) as it cannot handle object deformations other than translation. SAMF and DSST are extensions of the KCF approach. SAMF proposes an adaptive scale for the patches and also integrates HOG and colour attribute features. It is more robust than KCF, but still poorer than our result (15.65 vs 8.58). DSST [12], the VOT2014 challenge winner, improves the correlation filtering scheme by estimating the scale of the target and combining intensity and HOG features. It trains two separate filters: one for estimating translation and another for scale. While DSST shows better performance than both SAMF and KCF, it also cannot handle cases where the target is rotating. Our method addresses this issues and significantly improves over DSST (8.58 vs 13.99 in robustness rank).

Our tracker is ranked third on the robustness measure. PLT_{14} and PLT_{13}, which are two variants of *Struck* [19], are first and second respectively. PLT_{13}, the winner of the VOT2013 challenge, uses object-background colour histograms to weight the features within the bounding box during SVM training. In essence, it attempts to refine the object representation from an entire bounding box to an object segmentation. PLT_{14} is the multiscale version of PLT_{13}, and it shows better accuracy at the cost of a lower robustness. The segmentation cue used in these methods makes them accurate because it provides a more precise object representation than a bounding box. However, in many situations, e.g., fast motion, lack of strong object-background colour priors, it is likely to fail. This is evident from the significantly lower accuracy of the PLT methods – 16.04, 19.74 compared to 6.07 of our approach (see Table 1).

We present a sample qualitative result in Figure 5, see [1] for additional results. The *motocross* sequence is considered as one of the most challenging sequences in this benchmark, based on the number of trackers that fail on it. Our approach performs significantly better other methods. We compare to the results of DSST and PLT_{14}, which performed well on the VOT2014 challenge, and also *Struck*, the top performer on several datasets.

OTB. The results in Figure 6 correspond to the one-pass evaluation (OPE) of the toolkit [50]. The trackers are tested on more than 29,000 frames in this evaluation. We show the success and precision plots of our method in Figure 6 and compare it with the top 10 methods. Comparison with all the 28 trackers evaluated is shown in [1]. Our overall method "Our-ms-rot-em" and its four variants outperform all the trackers on both the measures. More precisely, our tracker achieves 0.798 precision score compared to 0.656 of *Struck* [19], and 0.580 AUC of success plot compared to 0.499 of SCM [52] – the top two performers on this benchmark dataset [50].

Our single-scale tracker "Our-ss" is inferior to the other four variants: 0.733 precision score and 0.523 AUC of success plot due to significant changes in scale and/or rotation in several sequences. The performance of our multiscale version without geometry proposals "Our-ms", and the multiscale one with geometry proposals "Our-ms-rot" is very similar. This is expected, despite "Our-ms-rot" performing better than "Our-ms" on the VOT dataset, because the ground truth annotations in the OTB dataset are axis-aligned and not rotated boxes, as is the case in VOT. Thus, a rotating object can be enclosed (although imprecisely) by a larger axis-aligned box. This makes "Our-ms" as effective as "Our-ms-rot" following the OTB evaluation protocol. In Figure 6 we also show the influence of scores computed with edges "e" and motion boundaries "m". The precision result using (multiscale) detector confidence alone to select from the candidate set ("Our-ms-rot") is 0.754. Incorporating the measure computed with edges ("Ours-ms-rot-e") improves this by 2.1% to 0.770, and the cue computed with motion boundary ("Our-ms-rot-em") gives a further 3.6% improvement to 0.798. A similar improvement pattern can be seen on the success plot.

A couple of recent papers [23, 47] also evaluated their trackers on the OTB dataset. However, they are not online trackers, in that they use the information for subsequent frames to infer the object location in the current frame. We, nevertheless, compare to them using mean F_1 score (with

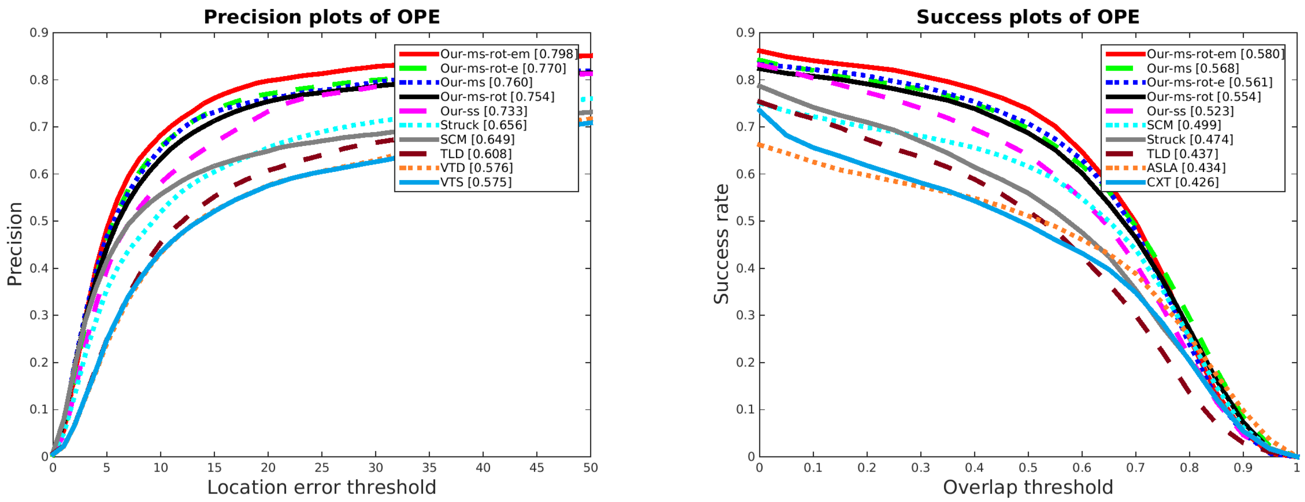

Figure 6. Precision (left) and success rate (right) plots on the OTB dataset. For clarity we compare to the top 10 algorithms in each figure. The complete plots are on the project website [1]. The precision plots are summarized with the precision score at 20 pixel error threshold and the success plots with the area under the curve. These values are shown in the legend. *Best viewed in pdf.*

50% overlap threshold), following the protocol in [23]. Our tracker ("Our-ms-rot-em") performs significantly better with a score of 0.757 compared to 0.657 and 0.661 of [23] and [47] respectively.

In Figure 7 we evaluate the performance of our approach ("Our-ms-rot-em") with respect to the number of candidate proposals on the OTB dataset. Here, we measure the performance as mean overlap between the predicted and the ground truth bounding boxes over the entire sequence. We observe that the mean overlap score increases initially with the number of proposals, but then quickly saturates. In other words, we require a minimum number of proposals (5 each for detection and geometry, which we use in this paper) for tracking an object effectively. Adding further candidate proposals beyond this does not affect the performance, as the new candidates are either very similar to or less accurate than those already in the proposal set.

Computation time. Our Matlab and mex implementation is un-optimized and not real-time. It performs as follows (average fps on VOT+OTB datasets): our-ss: 6.4, our-ms: 4.1, our-ms-rot: 3.1, our-ms-rot-e: 2.1, our-ms-rot-em: 1.9 on a cluster node with 20 cores. The last three variants require pre-computed optical flow [7], which runs at 0.3 fps on a GPU. Our implementation can certainly be sped up in several ways, e.g., by scaling the images down, speeding up the feature extraction process.

6. Summary

In this paper, we present a new tracking-by-detection framework for online object tracking. Our approach begins with building a candidate set of object location proposals extracted using a learned detector model. It is then augmented with novel proposals computed by estimating the geometric transformation(s) undergone by the object. We localize the object by selecting the best proposal from this candidate set using multiple cues: detection confidence score, edges and motion boundaries. The performance of our tracker is evaluated extensively on the VOT2014 challenge and the OTB datasets. We show state-of-the-art results on both these benchmarks, significantly improving over the top performers of these two evaluations.

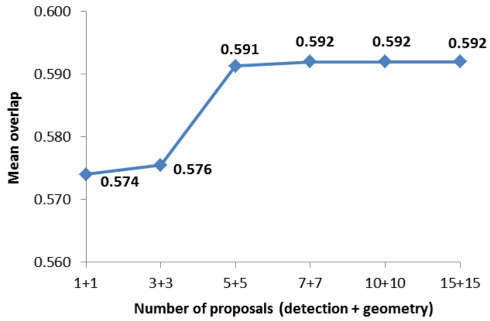

Figure 7. Influence of the number of (detection and geometry) proposals on the tracking performance on the OTB dataset.

Acknowledgements. This work was supported in part by the MSR-Inria joint project, Google Faculty Research Award, and the ERC advanced grant ALLEGRO.

References

[1] http://lear.inrialpes.fr/research/pstracker.

[2] B. Alexe, T. Deselaers, and V. Ferrari. Measuring the objectness of image windows. *PAMI*, 2012.

[3] S. Avidan. Ensemble tracking. *PAMI*, 2007.

[4] B. Babenko, M.-H. Yang, and S. Belongie. Robust object tracking with online multiple instance learning. *PAMI*, 2011.

[5] V. Badrinarayanan, P. Pérez, F. Le Clerc, and L. Oisel. Probabilistic color and adaptive multi-feature tracking with dynamically switched priority between cues. In *ICCV*, 2007.

[6] S. Birchfield. Elliptical head tracking using intensity gradients and color histograms. In *CVPR*, 1998.

[7] T. Brox and J. Malik. Large displacement optical flow: Descriptor matching in variational motion estimation. *PAMI*, 33(3):510–513, 2011.

[8] J. Carreira and C. Sminchisescu. Constrained parametric min-cuts for automatic object segmentation. In *CVPR*, 2010.

[9] R. T. Collins, Y. Liu, and M. Leordeanu. Online selection of discriminative tracking features. *PAMI*, 2005.

[10] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *PAMI*, 25(5):564–577, 2003.

[11] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

[12] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg. Accurate scale estimation for robust visual tracking. In *BMVC*, 2014.

[13] P. Dollár and C. L. Zitnick. Structured forests for fast edge detection. In *ICCV*, 2013.

[14] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL visual object classes (VOC) challenge. *IJCV*, 2010.

[15] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A library for large linear classification. *JMLR*, 9:1871–1874, 2008.

[16] 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.

[17] M. Godec, P. M. Roth, and H. Bischof. Hough-based tracking of non-rigid objects. In *ICCV*, 2011.

[18] H. Grabner, C. Leistner, and H. Bischof. Semi-supervised on-line boosting for robust tracking. In *ECCV*, 2008.

[19] S. Hare, A. Saffari, and P. H. S. Torr. Struck: Structured output tracking with kernels. In *ICCV*, 2011.

[20] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2004.

[21] J. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *PAMI*, 2015.

[22] P. V. C. Hough. Method and means for recognizing complex patterns, 1962. US Patent 3,069,654.

[23] Y. Hua, K. Alahari, and C. Schmid. Occlusion and motion reasoning for long-term tracking. In *ECCV*, 2014.

[24] M. Isard and A. Blake. ICONDENSATION: Unifying low-level and high-level tracking in a stochastic framework. In *ECCV*, 1998.

[25] X. Jia, H. Lu, and M.-H. Yang. Visual tracking via adaptive structural local sparse appearance model. In *CVPR*, 2012.

[26] Z. Kalal, K. Mikolajczyk, and J. Matas. Tracking-learning-detection. *PAMI*, 34(7):1409–1422, 2012.

[27] R. E. Kalman. A new approach to linear filtering and prediction problems. *J. Fluids Engineering*, 1960.

[28] M. Kristan, J. Matas, A. Leonardis, T. Vojir, R. Pflugfelder, G. Fernandez, G. Nebehay, F. Porikli, and L. Cehovin. A Novel Performance Evaluation Methodology for Single-Target Trackers. *arXiv:1503.01313*, 2015.

[29] J. Kwon and K. M. Lee. Tracking by sampling trackers. In *ICCV*, 2011.

[30] K. Lee, J. Ho, M. Yang, and D. Kriegman. Visual tracking and recognition using probabilistic appearance manifolds. *CVIU*, 99(3):303–331, 2005.

[31] B. Leibe, K. Schindler, N. Cornelis, and L. van Gool. Coupled object detection and tracking from static cameras and moving vehicles. *PAMI*, 30(10):1683–1698, 2008.

[32] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[33] M. Kristan et al. The visual object tracking VOT2014 challenge results. In *ECCV Visual Object Tracking Challenge Workshop*, 2014.

[34] I. Matthews, T. Ishikawa, and S. Baker. The template update problem. *PAMI*, 26(6):810–815, 2004.

[35] X. Mei and H. Ling. Robust visual tracking and vehicle classification via sparse representation. *PAMI*, 2011.

[36] G. Nebehay and R. Pflugfelder. Consensus-based matching and tracking of keypoints for object tracking. In *WACV*, 2014.

[37] Y. Pang and H. Ling. Finding the best from the second bests - inhibiting subjective bias in evaluation of visual tracking algorithms. In *ICCV*, 2013.

[38] D. Park, J. Kwon, and K. Lee. Robust visual tracking using autoregressive hidden Markov model. In *CVPR*, 2012.

[39] J. Pérez, P. Vermaak and A. Blake. Data fusion for visual tracking with particles. *Proc. IEEE*, 92(3):495–513, 2004.

[40] J. C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *NIPS*, 1999.

[41] D. Ramanan, D. Forsyth, and A. Zisserman. Tracking people by learning their appearance. *PAMI*, 29(1):65–81, 2007.

[42] D. A. Ross, J. Lim, R. Lin, and M. Yang. Incremental learning for robust visual tracking. *IJCV*, 2008.

[43] A. Smeulders, D. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah. Visual tracking: An experimental survey. *PAMI*, 36(7):1442–1468, 2014.

[44] S. Song and J. Xiao. Tracking revisited using RGBD camera: Unified benchmark and baselines. In *ICCV*, 2013.

[45] M. Spengler and B. Schiele. Towards robust multi-cue integration for visual tracking. *Machine Vis. App.*, 2003.

[46] B. Stenger, T. Woodley, and R. Cipolla. Learning to track with multiple observers. In *CVPR*, 2009.

[47] J. S. Supancic and D. Ramanan. Self-paced learning for long-term tracking. In *CVPR*, 2013.

[48] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid. Learning to detect motion boundaries. In *CVPR*, 2015.

[49] B. Wu and R. Nevatia. Detection and tracking of multiple, partially occluded humans by Bayesian combination of edgelet based part detectors. *IJCV*, 2007.

[50] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. In *CVPR*, 2013.

[51] J. Zhang, S. Ma, and S. Sclaroff. MEEM: robust tracking via multiple experts using entropy minimization. In *ECCV*, 2014.

[52] W. Zhong, H. Lu, and M.-H. Yang. Robust object tracking via sparsity-based collaborative model. In *CVPR*, 2012.

[53] C. L. Zitnick and P. Dollár. Edge boxes: Locating object proposals from edges. In *ECCV*, 2014.