Robust Visual Tracking with Deep Convolutional Neural Network based Object Proposals on PETS

Gao Zhu[1] Fatih Porikli[1,2,3] Hongdong Li[1,3]
Australian National University[1], NICTA[2]
ARC Centre of Excellence for Robotic Vision[3]
{gao.zhu,fatih.porikli,hongdong.li}@anu.edu.au *

Abstract

Tracking by detection based object tracking methods encounter numerous complications including object appearance changes, size and shape deformations, partial and full occlusions, which make online adaptation of classifiers and object models a substantial challenge.

In this paper, we employ an object proposal network that generates a small yet refined set of bounding box candidates to mitigate the this object model refitting problem by concentrating on hard negatives when we update the classifier. This helps improving the discriminative power as hard negatives are likely to be due to background and other distractions. Another intuition is that, in each frame, applying the classifier only on the refined set of object-like candidates would be sufficient to eliminate most of the false positives. Incorporating an object proposal makes the tracker robust against shape deformations since they are handled naturally by the proposal stage. We demonstrate evaluations on the PETS 2016 dataset and compare with the state-of-the-art trackers. Our method provides the superior results.

1. Introduction

Object tracking has been widely studied [31, 28, 19, 30] owing to its extensive applications from video surveillance to robotic vision. Robust and reliable tracking enables high level visual tasks. However, as we can see in Figure 1, real-life scenarios, especially surveillance videos, comprise many challenges:

- Pose and shape change – Humans have articulated bodies and they do not always stand up as depicted in the first row of Figure 1.
- Occlusion – In crowded scenes humans occlude each other frequently in the camera view as shown in the second row of Figure 1.
- Size change – Cameras are usually positioned to capture as wide viewing angles as possible. This causes objects to sustain significant size changes especially when they move along the imaging plane normal direction, e.g when humans enter the scene from faraway locations, as shown in the last row of Figure 1.

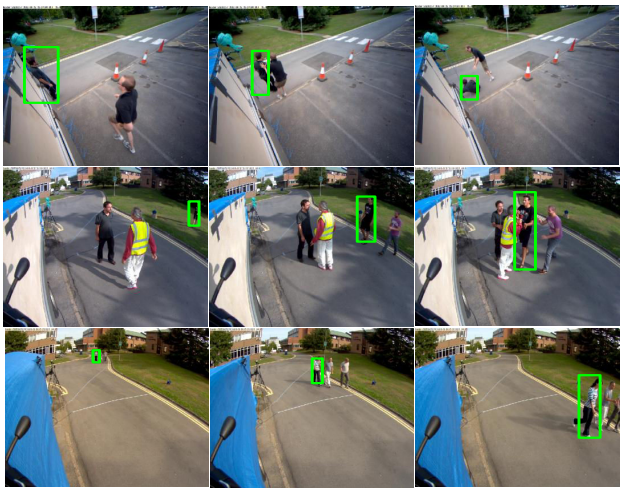

Figure 1: Typical video clips from the PETS 2016 dataset. Green bounding boxes denote ground truth annotations. As visible, objects undergo drastic size changes, deformations and partial occlusions, which cause obstacles for conventional object trackers. Our method handles these challenges naturally using object proposals generated by a deep convolutional neural network.

Most object trackers incorporate either a generative or a discriminative learning strategy to train their appearance models. Generative learning based models including the subspace learning [24] and sparse representation [21, 17, 15] mainly concentrate on how to construct an object rep-

*This work was supported under the Australian Research Councils Discovery Projects funding scheme (project DP150104645, DP120103896), Linkage Projects funding scheme (LP100100588), ARC Centre of Excellence on Robotic Vision (CE140100016) and NICTA (Data61).

resentation in specific feature spaces. In contrast, discriminative learning based appearance models aim to maximize the inter-class separability between the object and background regions using discriminative learning techniques, e.g. SVMs [2, 32, 35, 33], random forest [26], ridge regression [11, 6, 12] and multiple instance learning [3, 32], to name a few.

To adapt appearance changes due to deformations, size and shape changes, object models have to be processed and updated in an online manner using previous estimations. Since the appearance models are trained on the previous estimations and only a few positive training samples from the initial frames are reliable, it is considerably hard for these trackers to make accurate bounding box estimations when the target objects undergo drastic appearance changes. Once the tracker starts drifting, the location and overlap errors accumulate quickly distorting object model recursively and eventually leading up to a total tracking failure.

In this work, we introduce an object proposal generation procedure for handling the problem of model degradation. We obtain a small yet high-quality set of object proposals efficiently in the entire frame using a deep convolutional neural network (CNN) called region proposal network (RPN) [23] as shown in Figure 2. This network was trained offline using a large image dataset. When applied to a given image, it generates bounding boxes on the image regions that are likely to contain objects.

The benefits of using object proposals are fourfold:

- Since the extracted object proposals cover only "object-like" regions, a "regularity" for the tracking procedure is imposed by reducing the spurious false positives.
- Object proposals also suggest good negative samples for training as they correspond to possible distractions that may otherwise deteriorate the tracking process.
- The bounding boxes naturally accommodate size changes of the object.
- Tracking by object proposals enables tracking any object motion at any frame rate.

We validate the above arguments on the PETS 2016 [1] dataset comparing with several state-of-the-art trackers. Our method accomplishes the best *precision score* of 58.5 where the second best is achieved by EBT [34] with 52.6.

2. Related Work

We first review the current work using CNN features for visual tracking. Then, we give a simple introduction of object proposal methods and discuss some relative studies relevant to our method.

Convolutional Neural Networks for Tracking

Although significant advances have been attained by CNNs for object detection and classification tasks [25], there are comparably limited adaptations of CNNs for tracking task and most CNN based trackers use such networks to learn better features. In their pioneering work [16] employs a candidate pool of multiple CNNs as a data-driven model of different instances of the target object. Inspired by this, [20] interprets the hierarchies of convolutional layers as a nonlinear counterpart of an image pyramid representation and adaptively learns correlation filters on each convolutional layer to encode the target appearance. The recent work in [22] pretrains a CNN using a large set of videos with ground truth trajectories. The network is composed of shared layers and multiple branches of domain-specific layers. They train the network with respect to each domain iteratively to obtain generic target representations in the shared layers. In contrast, our method applies the CNN in a different fashion for both object proposal generation and feature extraction at the same time.

Object Proposals

As reported in [13, 36], use of proposal significantly improves the object detection benchmark along with the convolutional neural nets. Since, a subset of high-quality candidates are used for detection, object proposal methods boost not only the speed but also the accuracy by reducing false positives. The top performing detection methods [8, 29] for PASCAL VOC [7] use detection proposals.

The EdgeBoxes method [36] proposes object candidates based on the observation that the number of contours wholly enclosed by a bounding box is an indicator of the likelihood of the box containing an object. It is designed as a fast algorithm to balance between speed and proposal recall. BING [5] makes a similar observation that generic objects with well-defined closed boundary can be discriminated by looking at the norm of gradients. R-CNN [23] introduces the region proposal network (RPN), which is a fully end-to-end convolutional network that simultaneously predicts object bounds and objectness scores at each position. It shares full-image convolutional features with the detection network, thus enabling nearly cost-free region proposals.

Since it enables efficient extraction of object proposals and deep features, we employ RPN as the proposal generator in this paper.

Object Proposals for Tracking

A straightforward strategy based on linear combination of the original tracking confidence and an adaptive objectness score obtained by BING [5] is employed in [18]. In [14], a detection proposal scheme is applied as a post-processing step, mainly to improve the tracker's adaptability to scale and aspect ratio changes. More recent trackers [33, 34] are the most relevant approaches to ours. Here, we take the advantage of the deep networks and achieve better performance for PETS 2016 dataset.

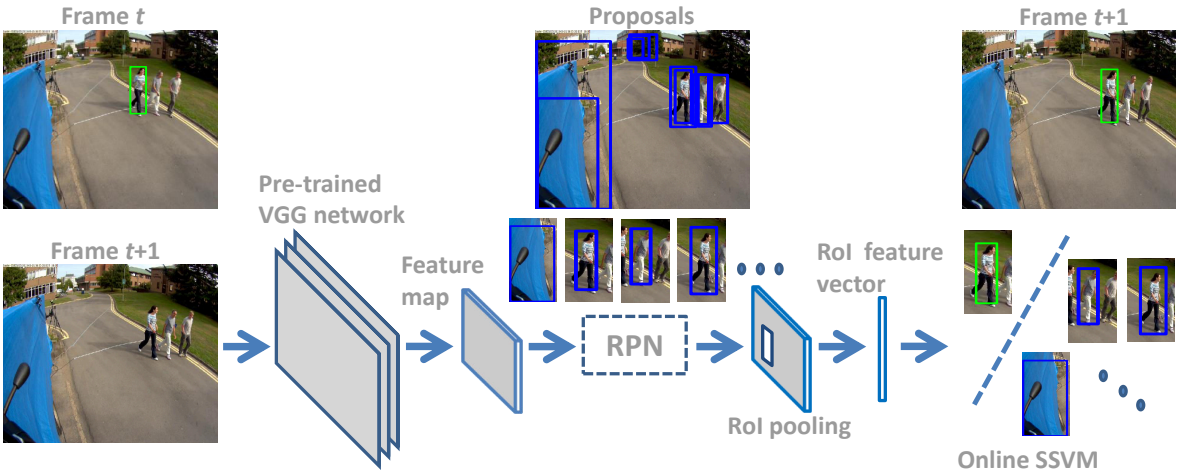

Figure 2: Framework of the proposed method. In a new frame $t+1$, an offline VGG network [27] is used to generate a feature map, which is then fed to the region proposal network (RPN) [23] to obtain candidate bounding boxes. Region of interest (RoI) pooling layer extracts feature vectors with a fixed size for the online structured support vector machine (SSVM) [4] that serves as the classifier. The proposal with the maximal response is assigned as the new location of the object. We call our method as Region Proposal Network Tracker: RPNT.

3. Proposed Tracker: RPNT

Our method follows a popular structured support vector machine (SSVM) based tracking-by-detection framework [4, 10]. The object location is initialized manually at the first frame $t = 1$. Denote B_t as a bounding box at frame t and can be represented as coordinates of its four corners. Then, given a classification function F_{t-1} trained on the previous frames, the current location of the object is estimated through:

$$B_t^\star = \arg\max_{B_t \in \mathcal{B}_t} F_{t-1}(B_t), \tag{1}$$

where $\mathcal{B}_t$ is a set of candidate samples at the current frame. To select samples, traditional trackers use heuristic search windows around the previously estimated object location for computational reasons. They apply each sample into a classifier. For example, a search radius of 30 pixels is used in [10]. Suppose the support vector set maintained by SSVM as $\mathcal{V}_{t-1}$ and the classification function can be written as a weighted sum of affinities:

$$F_{t-1}(B_t) = \sum_{B_{t-1}^i \in \mathcal{V}_{t-1}} w_{t-1}^i K(B_{t-1}^i, B_t), \tag{2}$$

where w_{t-1}^i is a scalar weight associated with the support vector B_{t-1}^i. Kernel function $K(B_{t-1}^i, B_t)$ calculates the affinity between two feature vectors extracted from B_{t-1}^i and B_t respectively.

The classifier will then revise its model with the new location of the object to adapt appearance changes. To update the support vector set $\mathcal{V}_{t-1} \rightarrow \mathcal{V}_t$ in an online fashion, a critical step is the selection of negative supports vector according to the following function:

$$B_t^- = \arg\max_{B_t \in \mathcal{B}_t \setminus B_t^\star} F_{t-1}(B_t) + L(B_t, B_t^\star), \tag{3}$$

where the loss function $L(B_t, B_t^\star) = 1 - (B_t \cap B_t^\star)/(B_t \cup B_t^\star)$ defines on the bounding box overlap. Optimization (3) corresponds to finding such a negative training sample that locates far from the positive one (high $L(B_t, B_t^\star)$) yet presents close appearance (high $F_{t-1}(B_t)$). For more details, refer to [10].

3.1. Region Proposal Network

The method proposed in this paper uses a similar framework as introduced above, yet we made one critical change to it. We recognize not all candidate bounding boxes $B_t \in \mathcal{B}_t$ should be treated equally (as the traditional trackers often do) since those boxes possess different "object-like" appearance, i.e. "objectness" characteristics, which should be taken into account.

To this end, we have incorporated the region proposal network (RPN) [23] to generate a small set of candidate bounding boxes (as shown in Figure 2), denoted as $\mathcal{B}_t^C$. This network is basically a combined classification and regression layer built upon a feature map extracted from a pretrained CNN network such as VGG[27]. We use the implementation of RPN from [23] which is pretrained on the PASCAL VOC dataset [7].

Similar to [34], we additionally generate a bounding box set by sampling only around the previous object location as $\mathcal{B}_t^R$ (as in traditional methods). We find them useful to

help smoothen the tracking trajectory as the object proposal component works independently at each frame, which inevitably results temporally inconsistent proposals. Thus a combined set of $\mathcal{B}_t = \mathcal{B}_t^C \cup \mathcal{B}_t^R$ is used during the test stage. However, we only update the classifier when the estimated one is from $\mathcal{B}_t^C$ to resist potential corruption. More details in Section 3.4.

3.2. Deep Feature Extraction

Like other CNN trackers [16, 20, 22], we extract deep features from the pretrained deep network. It is convenient to apply the extraction in our framework as the RPN is built on the deep feature map. We employ a spatial pooling layer [9] to avoid warping each image patch independently corresponding to bounding box candidates of various sizes. The feature pooling layer takes all the candidates in a single shot and saves computational cost by sharing the feature map.

3.3. Candidates Classification

Instead of Eq.1, we use the following decision function to estimate the new location of the object:

$$B_t^\star = \arg\max_{B_t \in \mathcal{B}_t} F_{t-1}(B_t) + S(B_t, B_{t-1}^\star). \quad (4)$$

$S(B_t, B_{t-1}^\star)$ is a term representing the motion smoothness between the previous object location and the candidate box. This is important in our formulation as we are testing candidates generated from various locations at the frame. We use a simple weighting function in this paper: $S(B_t, B_{t-1}^\star) = \min(\sigma \| c(B_t) - c(B_{t-1}^\star) \|, 1)$, where $c(B_t)$ is the center of bounding box B_t and σ is a constant.

3.4. Online Updating with Proposals

During the update stage, we use both of $\mathcal{B}_t^C$ and $\mathcal{B}_t^R$ to choose negative support vector. $\mathcal{B}_t^C \backslash B_t^\star$ represents other good "object-like" regions and training with them increases the discriminative power among "objects-like" candidates. $\mathcal{B}_t^R$ is used for training as well as the negative sample space contains a lot more other negative samples.

As mentioned in Section 3.1, we treat the estimated result $B_t^\star$ as an indication for model updating. This is to say, when $B_t^\star \in \mathcal{B}_t^R$, we assume that there is no good object proposal and the current estimation is a compromise for trajectory smoothness, thus skipping the model updating. If $B_t^\star \in \mathcal{B}_t^C$, then it suggests a good estimation which has both maximal classifier response and high "objectness", then we update the object model immediately.

4. Experiments on PETS

We evaluate the proposed method on several video sequences from Performance Evaluation of Tracking and Surveillance (PETS) 2016 [1]:

- N1_ARENA-01_02_TRK_RGB_2: three humans walking in parallel towards the camera, significant size change, as shown in the last row of Figure 1.
- W1_ARENA-11_03_ENV_RGB_3: two humans crashed, one was pulled down by the other, articulated body deformation, as shown in Figure 4 (a).
- W1_ARENA-11_03_TRK_RGB_1: Another viewing angle of "W1_ARENA-11_03_ENV_RGB_3", as shown in Figure 4 (b).
- A1_ARENA-15_06_TRK_RGB_2: Four humans fighting, one was pulled down, occlusion and body deformation, as shown in Figure 4 (c-f).

We list the details of the four sequences in Table 1 with corresponding attributes labeled. As we can see, all sequences contain the attribute of size change, while sequence "A1_ARENA-15_06_TRK_RGB_2" is the most challenging video containing size change, deformation and occlusion.

All trackers are initialized at the same frame of each sequence using a human detector [23].

4.1. Compared Trackers and Evaluation Metrics

Our method is denoted as RPNT and we compare it with several state-of-the-art trackers: EBT [34], MUSTer [12], SRDCF [6], KCF [11], Struck [10] and MEEM [32]. Most of them have been ranked at the top positions in recent large benchmarks [31, 28, 19, 30]. Among these trackers, EBT uses a similar proposal framework with EdgeBox [36] used as candidate generation algorithm. MUSTer and SRDCF improve the KCF tracking system. MEEM is an improved support vector machine tracker. Stuck uses the same SSVM classification model as ours, with a local uniform sampling scheme. For all the trackers, we use their default settings.

Evaluation metrics and code are provided by the benchmark [31, 30]. We employ the one-pass evaluation (OPE) and use two metrics: *precision plot* and *success plot*. The former one calculates the percentage (*precision score*) of frames whose center location is within a certain threshold distance with the ground truth. A commonly used threshold is 20 pixels. The latter one calculates a same percentage but based on bounding box overlap threshold. We utilize the area under curve (AUC) as an indicative measurement for it.

Parameters For the RPN setting, non-maximum suppression parameter is fixed at 0.7. The maximal number of proposal is 600. Notice that a typical number of proposals actually generated for PETS 2016 is about 400. For the setting in SSVM, we use the same parameters as used in EBT [34]. For the smooth motion function $S(B_t, B_{t-1}^\star)$ in Eq.4, σ is set as the diagonal length of the initialized bounding box.

videos	#humans	#frames	size change	deformation	occlusion
N1_ARENA-01_02_TRK_RGB_2	3	115	Yes	No	No
W1_ARENA-11_03_ENV_RGB_3	2	107	Yes	Yes	No
W1_ARENA-11_03_TRK_RGB_1	2	101/148	Yes	Yes	No
A1_ARENA-15_06_TRK_RGB_2	3	582/198/195	Yes	Yes	Yes

Table 1: Details of the tested video sequences from the PETS 2016 dataset.

Notice that, we fix all of the parameters for all of the sequences. We do not fine-tune parameter values for optimal performance.

4.2. Quantitative Experimental Results

The quantitative experimental results can be found in Table 2 and Figure 3. In Table 2, we have details of the tracking results for each human being tracked in the four videos. We report *precision score* (center distance threshold at 20 pixels) and area under curve (AUC) of the *success plot* respectively. In Figure 3, results at various thresholds can be found and they are ranked using *precision score* or AUC of *success plot*.

Our method, PRNT, achieves the best overall performance among the compared trackers. Especially when compared with EBT, which is the closest method to ours, we have a 1.5% improvement in AUC of *success plot* and 5.9% in term of *precision score*. This shows the advantage of using deep CNN based feature and object proposal method, comparing to the contour based proposal approach (RPN vs EdgeBox) and crafted feature (deep feature vs intensity histogram). More discussion is in Section 4.5.

The results also demonstrates that the object proposal based framework works noticeably better than conventional trackers which explore only the information from the video itself, on the PETS 2016. The EBT and proposed RPNT outperforms the best non-proposal tracker, SRDCF, 4%+ in AUC metric and 1.3%+ in *precision score*. It indicates the robustness of the incorporated "objectness" cues when the videos contain drastic appearance changes.

4.3. Size Adaption

From the results, we can see the importance of the tracker's ability to adapt the size change. As from Table 1, all four video sequences contain the attribute of size change.

While in the participated trackers, ours method and EBT adapt the size change naturally during proposing the object candidates. MUSTer and SRDCF additionally build a correlation filter in the scale space and select the one with the maximal response. In contrast, KCF, Struck and MEEM are all fixing the bounding box size thorough the tracking.

It is clear to see the differences when looking at the AUC metric, as the best non-size-adapt tracker, MEEM, achieves 35.6%, comparing to 51.5% of RPNT. While in term of *precision score*, KCF manages to achieve 50.6%, comparing to 58.5% of RPNT. This is surprising, as it does not lose the object totally. However, it would be better if the size change could be adapted from the first place.

4.4. Qualitative Experimental Analysis

We illustrate several qualitative results in Figure 4 from top ranked trackers: RPNT, EBT, SRDCF and MUSTer, for an intuitive analysis. The demonstrated results are from "W1_ARENA-11_03_ENV_RGB_3", "W1_ARENA-11_03_TRK_RGB_1" and "A1_ARENA-15_06_TRK_RGB_2". Especially, the last one contains occlusion, deformation as well as size change.

As we can see from the illustrated figures, the proposed RPNT adapts the size and appearance change naturally in most circumstances. For the challenging "A1_ARENA-15_06_TRK_RGB_2" video, the RPNT shows strong robustness while most of other trackers start to drift.

4.5. Proposal and Feature

We also test two PRNT variants: (1) using EdgeBox proposal and deep feature; (2) using RPN with histogram of intensity (HoI) feature. The results are included in Table 3. We can see that the first variant using "EdgeBox + Deep feature" generates a similar result to RPNT while the latter one, "RPN + HoI" generates a relatively inferior result. This demonstrates the importance of applying the deep feature while EdgeBox performs as well as RPN.

Table 3: Performance of two variants.

	EB + Deep feature	RPN + HoI feature
PETS (AUC/PS)	51.1/57.6	50.5/53.5

4.6. Computational Cost

The running speed of the proposed RPNT is reported in Table 2. The RPN and feature pooling network are implemented using GPU. They are running at a similar speed as EdgeBox proposal method. Overall, our method runs a close speed to EBT and Struck. As the SSVM classifier is currently implemented in C++ and an interface between the proposal component and SSVM is needed, RPNT could be further speed up with code optimization.

videos	Pro. RPNT	EBT [34]	MUSTer [12]	SRDCF [6]	KCF [11]	Struck [10]	MEEM [32]
N1_ARENA-01_02_TRK_RGB_2	73.6/99.1	68.0/72.2	74.7/97.4	72.8/100.0	58.6/99.1	57.3/100.0	57.8/53.9
	61.2/95.7	65.9/86.1	47.4/91.3	64.2/80.0	44.8/93.9	47.9/96.5	47.2/92.2
	59.4/87.8	69.3/89.6	64.6/94.8	67.1/94.8	47.0/99.1	16.7/20.9	40.7/22.6
W1_ARENA-11_03_ENV_RGB_3	64.0/100.0	49.5/100.0	56.3/98.1	57.5/100.0	43.8/85.0	41.7/85.0	44.4/82.2
	42.9/81.3	43.9/82.2	21.3/47.7	21.8/57.0	25.1/63.6	48.2/93.5	40.8/100.0
W1_ARENA-11_03_TRK_RGB_1	9.6/6.9	9.2/4.0	12.3/5.9	8.5/5.9	11.1/5.9	24.4/24.6	9.4/5.9
	39.2/25.7	45.6/15.5	18.5/5.4	21.2/2.0	20.9/2.0	45.8/12.2	11.7/1.4
A1_ARENA-15_06_TRK_RGB_2	51.4/21.5	46.4/8.4	4.4/1.0	44.2/2.7	4.5/1.0	24.4/24.6	4.5/0.9
	57.9/29.2	47.9/11.7	40.6/12.4	34.2/10.7	64.4/28.5	63.0/17.8	63.4/15.1
	55.8/37.9	54.0/56.4	63.0/61.5	68.4/60.0	31.3/28.2	18.4/15.4	31.7/24.6
Overall	51.5/58.5	50.0/52.6	40.3/51.6	46.0/51.3	35.1/50.6	35.3/44.8	35.6/40.1
fps	3.8	4.4	2.1	7.8	70.9	4.8	8.8

Table 2: Area Under Curve (AUC) of *success plot* and *precision score* (20 pixels threshold) reported on the tested video sequences in PETS 2016 (AUC/PS). Note every video contains several targets. fps: frames per second.

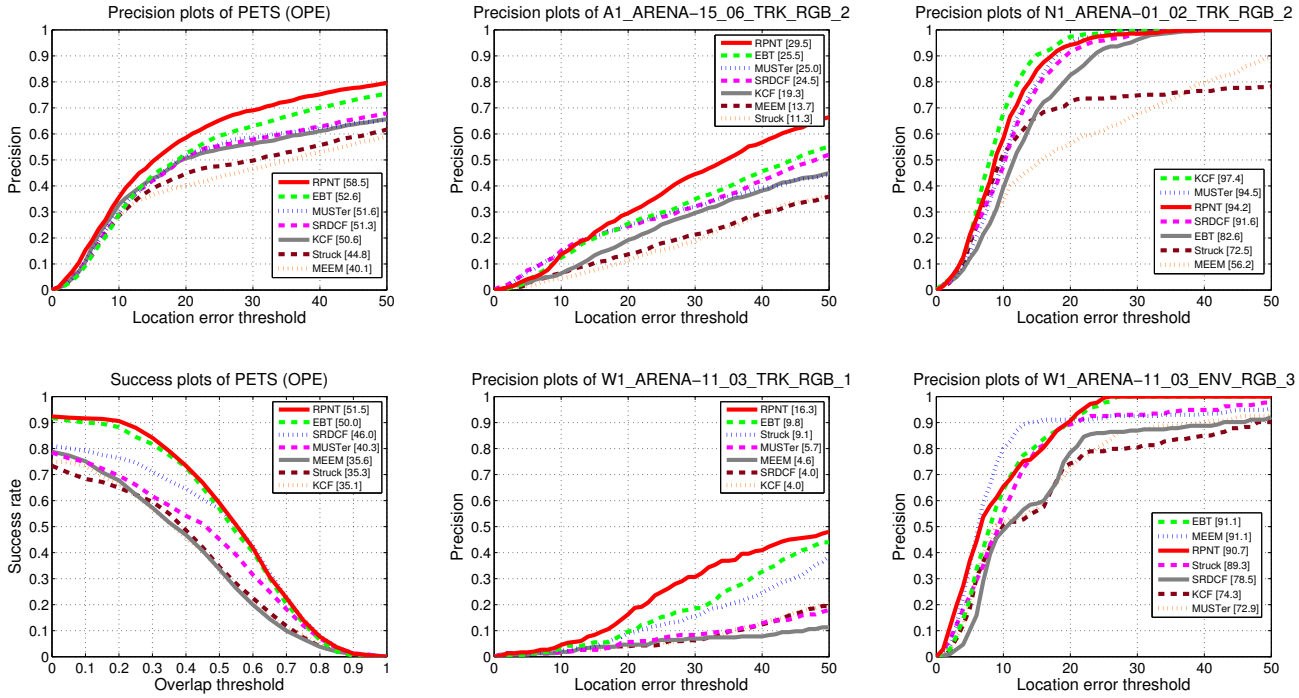

Figure 3: *Success plot* and *precison plot* on the PETS 2016 dataset of OPE. The number followed by the algorithm name is the area under the curve (AUC) and the *precision score* (PS) at the location error threshold of 20 pixels, respectively. Our method has an overall better performance.

5. Conclusion

This paper presented a robust object tracking method that naturally addresses the size change, deformation and occlusion problems common in surveillance videos through an object proposal procedure. The proposal network was built on the deep convolutional neural network and it enabled efficient deep feature extraction. It was evaluated on the PETS 2016 dataset and comparisons with recent state-of-the-art trackers was provided.

References

[1] IEEE International Workshop on Performance Evaluation of Tracking and Surveillance, PETS 2016. http://www.pets2016.net/. 2, 4

[2] S. Avidan. Support vector tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2001. 2

Figure 4: Qualitative comparisons with the state-of-the-art trackers on the PETS 2016 surveillance videos. Our method exhibits robustness in challenging scenarios such as size change, deformation and occlusion.

[3] B. Babenko, M. H. Yang, and S. Belongie. Visual tracking with online multiple instance learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2011. 2

[4] M. B. Blaschko and C. H. Lampert. Learning to localize objects with structured output regression. In *European Conference on Computer Vision (ECCV)*, 2008. 3

[5] M. Cheng, Z. Zhang, W. Lin, and P. H. S. Torr. BING: binarized normed gradients for objectness estimation at 300fps. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 2

[6] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg. Learning spatially regularized correlation filters for visual tracking. In *IEEE International Conference on Computer Vision (ICCV)*, 2015. 2, 4, 6

[7] M. Everingham, S. M. A. Eslami, L. V. Gool, C. K. I. Williams, J. M. Winn, and A. Zisserman. The visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 2015. 2, 3

[8] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 2

[9] R. B. Girshick. Fast R-CNN. *CoRR*, 2015. 4

[10] S. Hare, A. Saffari, and P. H. S. Torr. Struck: Structured output tracking with kernels. In *IEEE International Conference on Computer Vision (ICCV)*, 2011. 3, 4, 6

[11] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2015. 2, 4, 6

[12] Z. Hong, Z. Chen, C. Wang, X. Mei, D. Prokhorov, and D. Tao. Multi-store tracker (MUSTer): A cognitive psychology inspired approach to object tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 2, 4, 6

[13] J. Hosang, R. Benenson, and B. Schiele. How good are detection proposals, really? In *British Machine Vision Conference (BMVC)*, 2014. 2

[14] D. Huang, L. Luo, M. Wen, Z. Chen, and C. Zhang. Enable scale and aspect ratio adaptability in visual tracking with detection proposals. In *British Machine Vision Conference (BMVC)*, 2015. 2

[15] X. Jia, H. Lu, and M. H. Yang. Visual tracking via adaptive structural local sparse appearance model. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 1

[16] H. Li, Y. Li, and F. Porikli. DeepTrack: Learning discriminative feature representations by convolutional neural networks for visual tracking. In *British Machine Vision Conference (BMVC)*, 2014. 2, 4

[17] H. Li, C. Shen, and Q. Shi. Real-time visual tracking using compressive sensing. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011. 1

[18] P. Liang, C. Liao, X. Mei, and H. Ling. Adaptive objectness for object tracking. *CoRR*, 2015. 2

[19] M. Kristan et al. The visual object tracking VOT2014 challenge results. In *European Conference on Computer Vision (ECCV) Visual Object Tracking Challenge Workshop*, 2014. 1, 4

[20] C. Ma, J. B. Huang, X. Yang, and M. H. Yang. Hierarchical convolutional features for visual tracking. In *International Conference on Computer Vision (ICCV)*, 2015. 2, 4

[21] X. Mei and H. Ling. Robust visual tracking using l1 minimization. In *IEEE International Conference on Computer Vision (ICCV)*, 2009. 1

[22] H. Nam and B. Han. Learning multi-domain convolutional neural networks for visual tracking. *CoRR*, 2015. 2, 4

[23] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Neural Information Processing Systems (NIPS)*, 2015. 2, 3, 4

[24] D. A. Ross, J. Lim, R. S. Lin, and M. H. Yang. Incremental learning for robust visual tracking. *International Journal on Computer Vision*, 2008. 1

[25] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 2015. 2

[26] J. Santner, C. Leistner, A. Saffari, T. Pock, and H. Bischof. PROST: Parallel robust online simple tracking. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010. 2

[27] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, 2014. 3

[28] A. W. M. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah. Visual tracking: An experimental survey. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 2014. 1, 4

[29] X. Wang, M. Yang, S. Zhu, and Y. Lin. Regionlets for generic object detection. In *IEEE International Conference on Computer Vision (ICCV)*, 2013. 2

[30] Y. Wu, J. Lim, and M. Yang. Object tracking benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2015. 1, 4

[31] Y. Wu, J. Lim, and M. H. Yang. Online object tracking: A benchmark. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013. 1, 4

[32] J. Zhang, S. Ma, and S. Sclaroff. MEEM: Robust tracking via multiple experts using entropy minimization. In *European Conference on Computer Vision (ECCV)*, 2014. 2, 4, 6

[33] G. Zhu, F. Porikli, and H. Li. Tracking randomly moving objects on edge box proposals. *CoRR*, 2015. 2

[34] G. Zhu, F. Porikli, and H. Li. Beyond local search: Tracking objects everywhere with instance-specific proposals. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016,. 2, 3, 4, 6

[35] G. Zhu, F. Porikli, Y. Ming, and H. Li. Lie-Struck: Affine tracking on Lie groups using structured SVM. In *IEEE Winter conference on Applications of Computer Vision (WACV)*, 2015. 2

[36] C. L. Zitnick and P. Dollár. Edge boxes: Locating object proposals from edges. In *European Conference on Computer Vision (ECCV)*, 2014. 2, 4