Learning to Track at 100 FPS with Deep Regression Networks - Supplementary Material

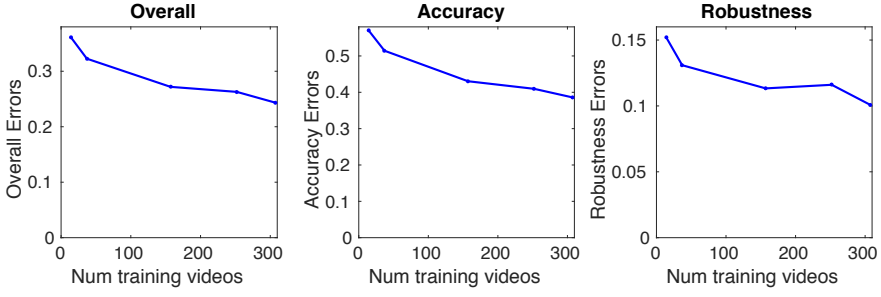
David Held, Sebastian Thrun, Silvio Savarese

Department of Computer Science
Stanford University
{davheld,thrun,ssilvio}@cs.stanford.edu

A Supplemental Video

Included in the Supplemental Material is a video demonstrating the performance of our tracker (GOTURN) on the VOT 2014 Tracking Challenge [6]. We compare our tracker to the 4 top performing baseline methods from the VOT 2014 Tracking Challenge [6]: SAMF, KCF, PLT_14, and DSST [2]. More details about the baseline methods can be found in the report on the VOT 2014 Tracking Challenge [6]. This supplemental video demonstrates the ability of our tracker to track novel objects under various motion changes, illumination changes, size changes, deformations, and minor occlusions. This video also demonstrates our failures, which can occur due to occlusions or overfitting to objects in the training set.

B Offline training



Supplementary Figure 1. Tracking performance as a function of the number of training videos (lower is better). This analysis indicates that large gains are possible by labeling more training videos.

Our tracker is able to improve its performance as it trains on more offline data. By observing more videos, GOTURN learns how the appearance of objects

change as they move. We further analyze the effect of the amount of training data on our tracker's performance in Supplementary Figure 1. We see that that the tracking errors drop dramatically as we increase the number of training videos. Our state-of-the-art results demonstrated in Section 6.1 of the main text were obtained after training on only 307 short videos, ranging from a few seconds to a few minutes in length, with an average of 52 annotations per video. Supplementary Figure 1 indicates that large gains could be achieved if the training set size were increased by labeling more videos.

C Online training

Previous neural network trackers for tracking generic objects have been trained online [8, 9, 12, 14, 13, 10, 15, 3, 7, 4]. Unfortunately, such trackers are very slow to train, ranging from 0.8 fps [8] to 15 fps [14], with the top performing neural-network trackers running at 1 fps [10, 3, 5]. Our tracker is trained offline in a generic manner, so no online training of our tracker is required. As a result, our tracker is able to track novel objects at 100 fps.

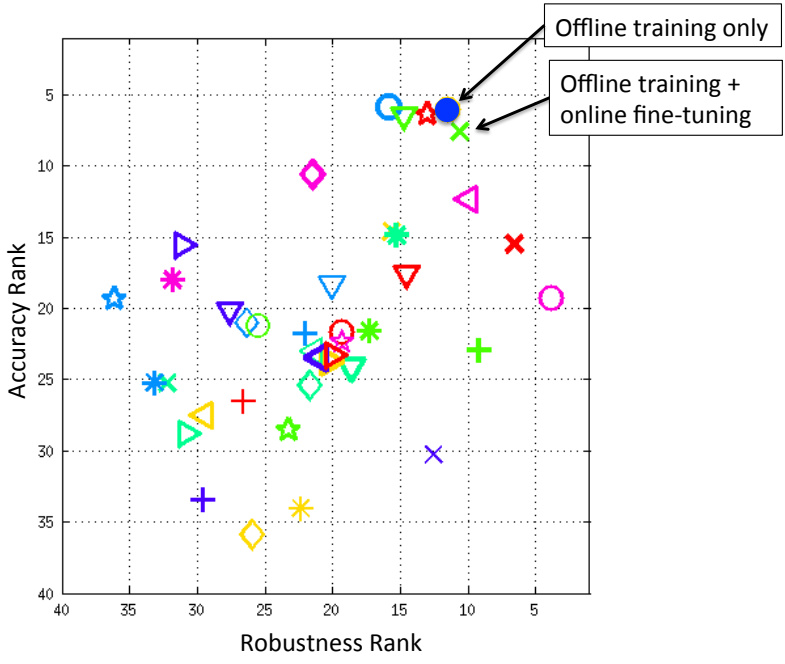
In Supplementary Figures 2 and 3, we explore the benefits of online training. We use cross-validation to choose the online learning rate to be 1e-9. Supplementary Figure 2 shows that online training does not significantly improve performance beyond our offline training procedure. As might be expected, there is a small increase in robustness from online training; however, this comes at a cost of accuracy, since online training tends to overfit to the first few frames of a video and would not easily generalize to new deformations or viewpoint changes. A more detailed analysis is shown in Supplementary Figure 3.

Our offline training procedure has seen many training videos with deformations, viewpoint changes, and other variations, and thus our tracker has already learned to handle such changes in a generic manner that generalizes to new objects. Although there might be other ways to combine online and offline training, our network has already learned generic target tracking from its offline training procedure and achieves state-of-the-art tracking performance without any online training required.

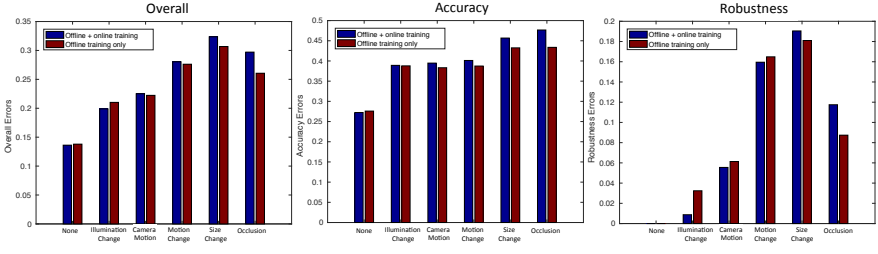
D Generality vs Specificity

In the main text, we analyze the generality of our tracker. We demonstrate that our tracker can generalize to novel objects not found in the training set. At the same time, a user can train our tracker to track a particular class of objects especially well by giving more training examples of that class of objects. This is useful if the tracker is intended to be used for a particular application where certain classes of objects are more prevalent.

We show more detailed results of this experiment in Supplementary Figure 4. Analyzing the accuracy and robustness separately, we observe an interesting pattern. As the number of training videos increases, the accuracy errors decreases equally both for object classes that appear in our training set and classes that

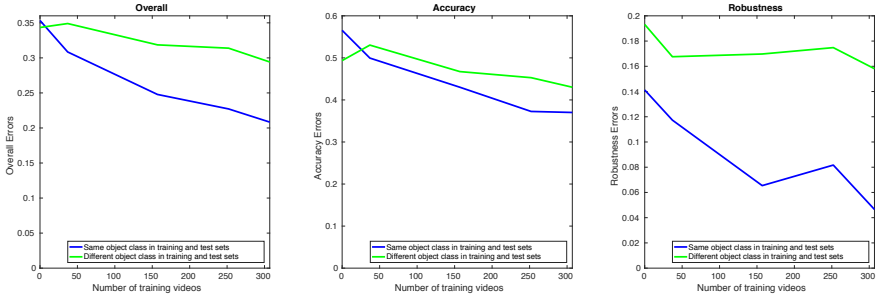


Supplementary Figure 2. Tracking results from the VOT 2014 tracking challenge. Our tracker's performance is indicated with a blue circle, outperforming all previous methods on the overall rank (average of accuracy and robustness ranks). A version of our tracker with online training is shown with a green X. Both versions achieve approximately the same performance, demonstrating that our offline training procedure has already taught the network how to track a variety of objects.



Supplementary Figure 3. Comparison of our tracker with and without online training (lower is better). Both versions achieve approximately the same performance, demonstrating that our offline training procedure has already taught the network how to track a variety of objects. Online training can lead to overfitting to the first few frames of a video, leading to more errors.

do not appear in our training set. On the other hand, the decrease in robustness errors is much more significant for object classes that appear in our training set compared to classes that do not.

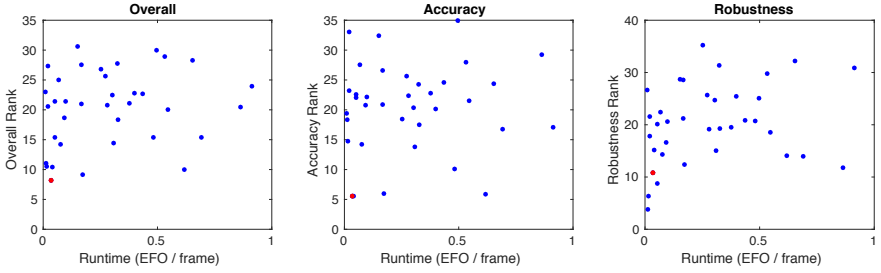


Supplementary Figure 4. Overall tracking errors for different types of objects in our test set as a function of the number of videos in our training set (lower is better). Class labels are not used by our tracker; these labels were obtained only for the purpose of this analysis.

Thus our tracker is able to learn generic properties about objects that enable it to accurately track objects, i.e. to accurately denote the borders of the object with a bounding box. On the other hand, the tracker's ability to generalize robustness is more limited; the tracker has a hard time tracking the motion of unknown objects when faced with difficult tracking situations. This analysis points towards future work to increase the robustness of the tracker by labeling more videos or by learning to train on unlabeled videos.

E Speed analysis

In the main text, we showed the speed of our tracker as a function of the overall rank (computed as the average of accuracy and robustness ranks) and showed that we have the lowest overall rank while being one of the fastest trackers. In Supplementary Figure 5 we show more detailed results, demonstrating our tracker's speed as a function of the accuracy rank and the robustness ranks. Our tracker has the second-highest accuracy rank, one of the top robustness ranks, and the top overall rank, while running at 100 fps. Previous neural-network trackers range from 0.8 fps [8] to 15 fps [14], with the top performing neural-network trackers running at only 1 fps GPU [10, 3, 5], since online training of neural networks is slow. Thus, by performing all of our training offline, we are able to make our neural network tracker run in real-time.



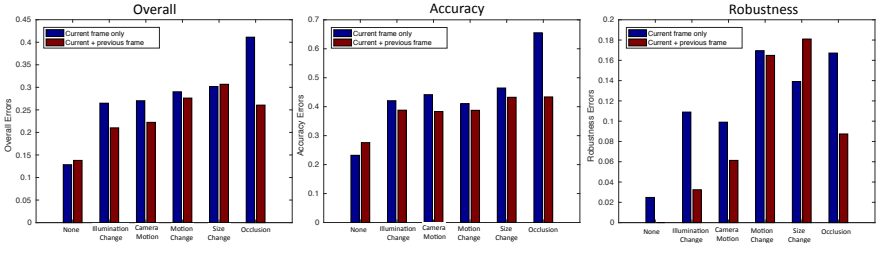
Supplementary Figure 5. Rank vs runtime of our tracker (red) compared to the 38 baseline methods from the VOT 2014 Tracking Challenge (blue). Each blue dot represents the performance of a separate baseline method (best viewed in color).

F How does it work?

In the main text, we explored how our tracker works as a combination of two hypotheses:

1. The network compares the previous frame to the current frame to find the target object in the current frame.
2. The network acts as a local generic "object detector" and simply locates the nearest "object."

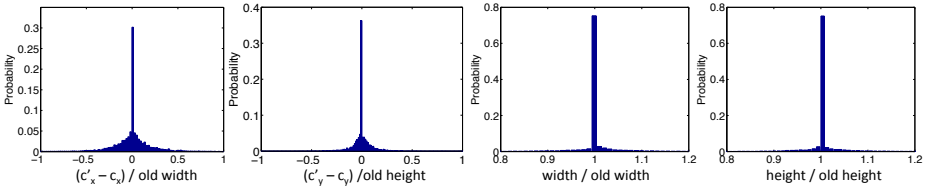
We distinguished between these hypotheses by comparing the performance of our network to the performance of a network which does not receive the previous frame as input. In Supplementary Figure 6 we show more details of this experiment, showing also accuracy and robustness rankings. For a more detailed interpretation of the results, see Section 6.2 of the main text.



Supplementary Figure 6. Tracking errors for our network which receives as input both the current and previous frame, compared to a network which receives as input only the current frame (lower is better). This comparison allows us to disambiguate between two hypotheses that can explain how our neural-network tracker works (see Section 6.2 of the main text).

G Motion Smoothness Distribution

In Section 4.2 of the main text, we describe how we use random cropping to implicitly encode the idea that small motions are more likely than large motions. To determine which distribution to use to encode this idea, we analyze the distribution of object motion found in the training set. This motion distribution can be seen in Supplementary Figure 7. As can be seen from this figure, each of these distributions can be modeled by Laplace distributions. Accordingly, we use Laplace distributions for our random cropping procedure. Note that the training set was only used to determine the shape of the distribution (i.e. Laplace); we use our validation set to determine the scale parameters for the distributions.



Supplementary Figure 7. Statistics for the change in bounding box size and location across two consecutive frames in our training set.

In more detail, suppose that the bounding box in frame $t-1$ is given by (c_x, c_y, w, h) where c_x and c_y are the coordinates of the center of the bounding box and w and h are the width and height accordingly. Then the bounding box at time t can be seen as drawn from a distribution:

$$c'_x = c_x + w \cdot \Delta x \tag{1}$$
$$c'_y = c_y + h \cdot \Delta y \tag{2}$$
$$w' = w \cdot \gamma_w \tag{3}$$
$$h' = h \cdot \gamma_h \tag{4}$$

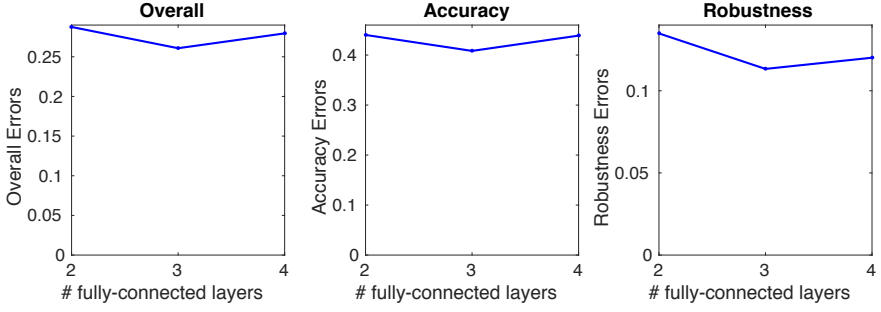
with random variables Δx, Δy, γ_w, and γ_h, where (c'_x, c'_y, w', h') parameterize the bounding box at time t using the same representation described above. In terms of the random variables, we can rewrite these expressions as

$$\Delta x = (c'_x - c_x)/w \tag{5}$$
$$\Delta y = (c'_y - c_y)/h \tag{6}$$
$$\gamma_w = w'/w \tag{7}$$
$$\gamma_h = h'/h \tag{8}$$

The empirical distributions of these random variables over the training set are shown in Supplementary Figure 7.

H Number of layers

In Supplementary Figure 8 we explore the effect of varying the number of fully-connected layers on top of the neural network on the tracking performance. These fully-connected layers are applied after the initial convolutions are performed on each image. This figure demonstrates that using 3 fully-connected layers performs better than using either 2 or 4 layers. However, the performance is similar for 2, 3, or 4 fully-connected layers, showing that, even though 3 fully-connected layers is optimal, the performance of the tracker is not particularly sensitive to this parameter.

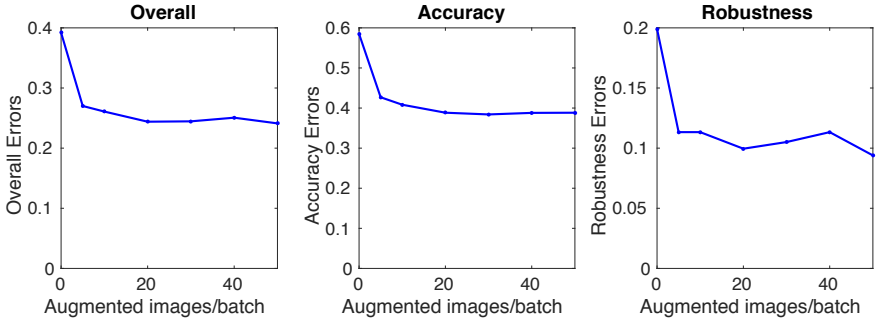


Supplementary Figure 8. Tracking performance as a function of the number of fully-connected layers in the neural network (lower is better).

I Data augmentation

In Supplementary Figure 9 we explore the effect of varying the number of augmented images created for each batch of the training set. Note that new augmented images are created on-the-fly for each batch. However, varying the number of augmented images varies the percentage of each batch that consists of real images compared to augmented images. Our batch size is 50, so we can vary the number of augmented images in each batch from 0 to 49 (to leave room for at least 1 real image).

As shown in Supplementary Figure 9, best performance is achieved when 49 augmented images are used per batch, i.e. only 1 real image is used, and the remainder are augmented. However, performance is similar for all values of augmented images greater than 20. In our case (with a batch size of 50), this indicates that performance is similar as long as at least 40% of the images in the batch are augmented. The augmented images show the same examples as the real images, but with the target object translated or with a varying scale. Augmented images thus teach the network how the bounding box position changes due to translation or scale changes.



Supplementary Figure 9. Tracking performance as a function of the number of augmented images in each batch (lower is better). Note that new augmented images are created on-the-fly for each batch.

J Training Set

Our training set was taken from ALOV300++ [11]. To ensure that there was no overlap with our test set, we removed 7 videos from our training set. These videos are:

- 01-Light_video00016
- 01-Light_video00022
- 01-Light_video00023
- 02-SurfaceCover_video00012
- 03-Specularity_video00003
- 03-Specularity_video00012
- 10-LowContrast_video00013

After removing these 7 overlapping videos, there is no overlap between the videos in the training and test sets.

K Detailed Results

The detailed results of our method compared to the 38 other methods that were submitted to the VOT 2014 Tracking Challenge [6] are shown in Supplementary Table 1. The VOT 2014 Tracking Challenge consists of two types of experiments. In the first experiment, the trackers are initialized with an exact ground-truth bounding box ("exact"). In the second experiment, the trackers are initialized with a noisy bounding box, which is shifted slightly off of the target object ("noisy"). For the noisy initialization experiment, the same 15 noisy initializations are used for each tracker, and the results shown are an average of the tracking performance across these initializations. This experiment allows us to determine the robustness of each tracker to errors in the initialization. This

noisy initialization procedure imitates that of a noisy automatic initialization process or noisy human initializations.

The trackers are evaluated using two standard tracking metrics: accuracy and robustness [6, 1]. Each frame of the video is annotated with a number of attributes: occlusion, illumination change, motion change, size change, and camera motion. The trackers are ranked in accuracy and robustness separately for each attribute, and the rankings are then averaged across attributes to get a final accuracy and robustness ranking for each tracker. The accuracy and robustness rankings are averaged to get an overall ranking, shown in Supplementary Table 1.

	Overall Ranks		Accuracy Ranks		Robustness Ranks		Speed
Method name	Exact	Noisy	Exact	Noisy	Exact	Noisy	Frames/EFO
GOTURN (Ours)	**8.206944**	**8.588319**	5.544841	7.227564	10.869048	9.949074	29.928769
SAMF	9.970153	9.297234	5.866667	5.685897	14.073638	12.908571	1.617264
KCF	10.368056	9.341055	5.533730	5.583333	15.202381	13.098776	24.226259
DSST	9.193519	9.393977	5.979630	5.855556	12.407407	12.932399	5.803051
PLT_14	10.526710	9.412576	14.720087	13.726667	6.333333	5.098485	62.846506
DGT	10.633462	9.880582	11.719306	9.318182	9.547619	10.442982	0.231538
PLT_13	11.045249	11.066132	18.340498	17.298932	3.750000	4.833333	75.915548
eASMS	14.267836	12.838634	14.220760	11.327036	14.314912	14.350232	13.080900
HMMTxD	15.398256	14.663101	10.070087	9.727810	20.726425	19.598391	2.075963
MCT	15.376874	15.313581	16.806659	17.576278	13.947090	13.050884	1.447154
ABS	19.651999	15.340186	20.666961	15.344515	18.637037	15.335856	0.623772
ACAT	14.438846	16.338981	13.796118	17.769841	15.081575	14.908122	3.237589
MatFlow	15.393888	16.910356	21.996109	19.142094	8.791667	14.678618	19.083821
LGTv1	20.504135	18.189239	29.225131	26.533460	11.783138	9.845018	1.158273
ACT	18.676877	18.692439	20.756783	22.184568	16.596972	15.200311	10.858222
VTDMG	19.992574	18.835055	21.481942	20.647094	18.503205	17.023016	1.832097
qwsEDFT	18.365675	19.776101	17.495604	18.545589	19.235747	21.006612	3.065546
BDF	20.535189	19.905596	23.242965	21.731090	17.827413	18.080103	46.824844
Struck	21.038417	20.129413	20.868501	21.424688	21.208333	18.834137	5.953411
ThunderStruck	21.389674	20.333286	22.612468	21.989153	20.166880	18.677419	19.053603
DynMS	21.141005	20.479737	22.815739	21.510423	19.466270	19.449050	2.650560
aStruck	20.780963	21.465762	22.409722	20.878854	19.152203	22.052670	3.576635
SIR_PF	22.705212	22.413896	24.537547	22.331205	20.872878	22.496587	2.293901
Matrioska	21.371144	23.119954	22.115980	21.947863	20.626308	24.292044	10.198580
EDFT	22.516498	23.176905	20.338931	22.141689	24.694066	24.212121	3.297059
OGT	22.463076	23.528818	14.810633	16.893364	30.115520	30.164271	0.393198
CMT	22.788164	23.852773	20.098007	22.612765	25.478321	25.092781	2.507500
FoT	23.003472	24.375915	19.388889	21.623392	26.618056	27.128439	114.643138
IIVTv2	25.669987	24.610138	25.651061	25.400309	25.688913	23.819967	3.673112
IPRT	25.014620	25.081882	27.564283	26.643535	22.464957	23.520229	14.688296
PTp	27.288300	25.208133	33.046296	30.268937	21.530303	20.147328	49.892214
LT_FLO	23.958402	26.020573	17.075617	20.843334	30.841186	31.197811	1.096522
FSDT	28.275770	26.805519	24.378835	24.318730	32.172705	29.292308	1.529770
IVT	28.892955	27.820781	27.952576	27.432765	29.833333	28.208796	1.879526
IMPNCC	27.566645	29.293698	26.570580	29.349962	28.562711	29.237434	5.983489
CT	30.585835	29.377864	32.462103	30.823647	28.709566	27.932082	6.584306
FRT	27.800316	29.554293	24.300128	27.199856	31.300505	31.908730	3.093665
NCC	26.831924	30.305656	18.497180	23.444646	35.166667	37.166667	3.947948
MIL	30.007762	30.638921	34.934175	35.527778	25.081349	25.750064	2.012286

Supplementary Table 1. Full results from the VOT 2014 tracking challenge, comparing our method (GOTURN) to the 38 other methods submitted to the competition. We initialize the trackers in two different ways: with the exact ground-truth bounding box ("Exact") and with a noisy bounding box ("Noisy").

References

1. Cehovin, L., Kristan, M., Leonardis, A.: Is my new tracker really better than yours? In: Applications of Computer Vision (WACV), 2014 IEEE Winter Conference on. pp. 540–547. IEEE (2014)
2. Danelljan, M., Häger, G., Khan, F., Felsberg, M.: Accurate scale estimation for robust visual tracking. In: British Machine Vision Conference, Nottingham, September 1-5, 2014. BMVA Press (2014)
3. Danelljan, M., Hager, G., Shahbaz Khan, F., Felsberg, M.: Learning spatially regularized correlation filters for visual tracking. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 4310–4318 (2015)
4. Hong, S., You, T., Kwak, S., Han, B.: Online tracking by learning discriminative saliency map with convolutional neural network. In: Proceedings of the 32nd International Conference on Machine Learning, 2015, Lille, France, 6-11 July 2015 (2015)
5. Kristan, M., Matas, J., Leonardis, A., Felsberg, M., Cehovin, L., Fernandez, G., Vojir, T., Hager, G., Nebehay, G., Pflugfelder, R.: The visual object tracking vot2015 challenge results. In: Proceedings of the IEEE International Conference on Computer Vision Workshops. pp. 1–23 (2015)
6. Kristan, M., Pflugfelder, R., Leonardis, A., Matas, J., Čehovin, L., Nebehay, G., Vojíř, T., Fernandez, G., Lukežič, A., Dimitriev, A., et al.: The visual object tracking vot2014 challenge results. In: Computer Vision-ECCV 2014 Workshops. pp. 191–217. Springer (2014)
7. Kuen, J., Lim, K.M., Lee, C.P.: Self-taught learning of a deep invariant representation for visual tracking via temporal slowness principle. Pattern Recognition 48(10), 2964–2982 (2015)
8. Li, H., Li, Y., Porikli, F.: Deeptrack: Learning discriminative feature representations by convolutional neural networks for visual tracking. In: Proceedings of the British Machine Vision Conference. BMVA Press (2014)
9. Li, H., Li, Y., Porikli, F.: Deeptrack: Learning discriminative feature representations online for robust visual tracking. arXiv preprint arXiv:1503.00072 (2015)
10. Nam, H., Han, B.: Learning multi-domain convolutional neural networks for visual tracking. arXiv preprint arXiv:1510.07945 (2015)
11. Smeulders, A.W., Chu, D.M., Cucchiara, R., Calderara, S., Dehghan, A., Shah, M.: Visual tracking: an experimental survey. Pattern Analysis and Machine Intelligence, IEEE Transactions on 36(7), 1442–1468 (2014)
12. Wang, L., Ouyang, W., Wang, X., Lu, H.: Visual tracking with fully convolutional networks. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 3119–3127 (2015)
13. Wang, N., Li, S., Gupta, A., Yeung, D.Y.: Transferring rich feature hierarchies for robust visual tracking. arXiv preprint arXiv:1501.04587 (2015)
14. Wang, N., Yeung, D.Y.: Learning a deep compact image representation for visual tracking. In: Advances in neural information processing systems. pp. 809–817 (2013)
15. Zhang, K., Liu, Q., Wu, Y., Yang, M.H.: Robust visual tracking via convolutional networks. arXiv preprint arXiv:1501.04505 (2015)