

GUSOT: Green and Unsupervised Single Object Tracking for Long Video Sequences

Zhiruo Zhou ^{*}, Hongyu Fu ^{*}, Suya You [†] and C.-C. Jay Kuo ^{*}

^{*}University of Southern California, Los Angeles, USA

[†]Army Research Laboratory, Maryland, USA

Abstract—Supervised and unsupervised deep trackers that rely on deep learning technologies are popular in recent years. Yet, they demand high computational complexity and a high memory cost. A green unsupervised single-object tracker, called GUSOT, that aims at object tracking for long videos under a resource-constrained environment is proposed in this work. Built upon a baseline tracker, UHP-SOT++, which works well for short-term tracking, GUSOT contains two additional new modules: 1) lost object recovery, and 2) color-saliency-based shape proposal. They help resolve the tracking loss problem and offer a more flexible object proposal, respectively. Thus, they enable GUSOT to achieve higher tracking accuracy in the long run. We conduct experiments on the large-scale dataset LaSOT with long video sequences, and show that GUSOT offers a lightweight high-performance tracking solution that finds applications in mobile and edge computing platforms.

Index Terms—object tracking, online tracking, single object tracking, unsupervised tracking, green tracking

I. INTRODUCTION

Video object tracking is a fundamental problem in computer vision and has a wide range of applications such as autonomous navigation and video recognition. A popular branch of visual tracking is single object tracking where the object marked in the first frame is tracked through the whole video. Deep-learning-based trackers, called deep trackers, have been popular in the last 7 years. Supervised deep trackers have been intensively studied. Its superior performance is achieved by exploiting a large amount of offline labeled data. While supervision is powerful in guiding the learning process, it casts doubt on the reliability of tracking unseen objects. Unsupervised deep trackers [1]–[6] have been developed to address this concern in recent years.

Research on supervised and unsupervised deep trackers has primarily focused on tracking performance. The high performance of deep trackers is accompanied with high computational complexity and a huge memory cost. Generally, they are difficult to deploy in resource-limited platforms such as mobile and edge devices. Specific examples include drones, autonomous vehicles, mobile phones, etc. Furthermore, some state-of-the-art trackers are short-term trackers since they cannot recover from object tracking loss automatically, which occurs frequently in long video tracking scenarios.

To tackle the unsupervised long-term tracking problem in a resource-constrained environment, we propose a green and unsupervised single-object tracker (GUSOT) in this work. Built upon a short-term tracking baseline known as UHP-SOT++ [6], we introduce two additional new modules to

GUSOT: 1) lost object recovery, and 2) color-saliency-based shape proposal. The first module helps recover a lost object with a set of candidates by leveraging motion in the scene and selecting the best one with local/global features (e.g., color information). The second module facilitates accurate and long-term tracking by proposing bounding-box proposals of flexible shape for the underlying object using low-cost yet effective segmentation. Both modules are lightweight and can be easily integrated with UHP-SOT++. They enable GUSOT to achieve higher tracking accuracy in the long run. We conduct experiments on a large-scale benchmark dataset, LaSOT [7], containing long video sequences and compare GUSOT with several state-of-the-art trackers. Experimental results show that GUSOT offers a lightweight tracking solution whose performance is comparable with that of deep trackers.

The rest of this paper is organized as follows. Related work is reviewed in Sec. II. The proposed GUSOT method is detailed in Sec. III. Experimental results are given in Sec. IV. Finally, concluding remarks are provided in Sec. V.

II. RELATED WORK

Supervised deep trackers. Supervised deep trackers [8]–[18] has progressed a lot in the last seven years and offer state-of-the-art tracking accuracy. Siamese-network-based trackers [19]–[23] are popular because of their simple architecture yet effective tracking performance. They extract features from the target object as well as a search window around the target using a shared deep network and, then, conduct cross-correlation between the two feature maps to generate a response map. The response map can be used to locate the object and further processed by some downstream heads for object classification and box regression. Recently, supervised transformer-based trackers have been developed in [24], [25] with excellent performance.

Unsupervised deep trackers. Research on unsupervised deep trackers focuses on training deep trackers without labels on offline datasets [1]–[4]. UDT [1] offers a pioneering solution along this direction. It trains a network by tracking forward and backward in video with cycle consistency. Later, efforts have been made to improve the learning capability of UDT. Examples include mining positive and negative samples in ResPUL [2], mining moving objects via optical flow in USOT [3], or improving cycle training with better long-short term features or loss re-weighting in ULAST [4]. Yet, many of them only have a limited performance gain as compared

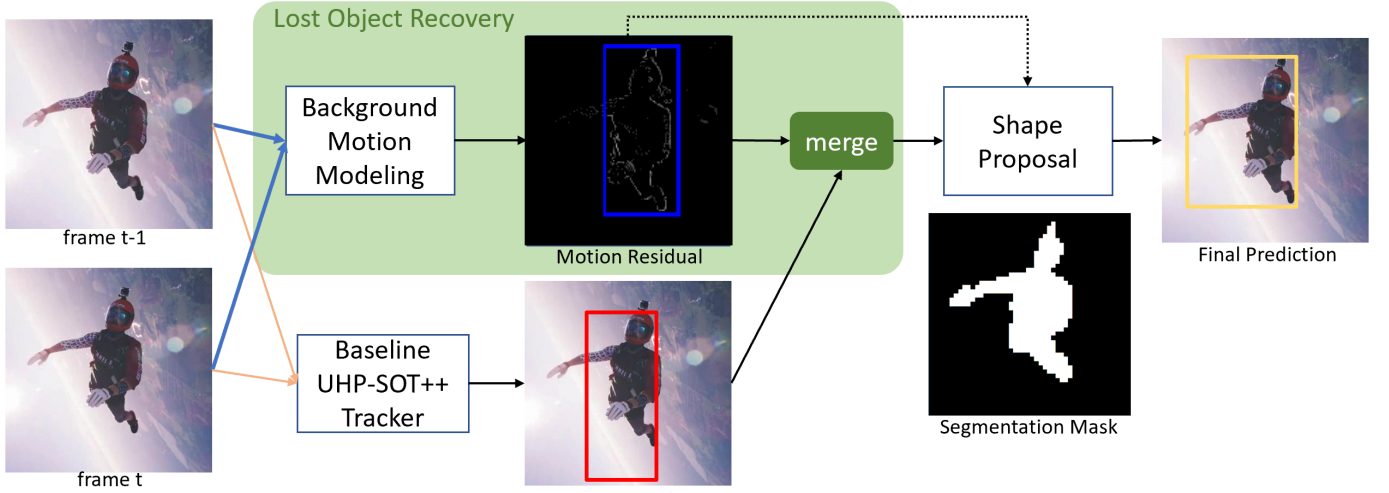


Fig. 1. An overview of the proposed GUSOT tracker, where the red and blue boxes denote the baseline and the motion proposals, respectively. The one with higher appearance similarity is chosen to be the location proposal. Then, the third proposal, called the shape proposal, is used to adjust the shape of the location proposal. The final predicted box is depicted by the yellow box.

with conventional trackers based on discriminative correlation filters (DCF). Furthermore, they are not lightweight trackers and cannot be adopted in resource-constrained devices.

Unsupervised conventional trackers. Before the deep learning era, most conventional trackers are unsupervised. One class of them are based on DCFs [9], [26]–[33]. DCF trackers learn a template, \mathbf{f} , from the first frame with regression and update the template via

$$\arg \min_{\mathbf{f}} \frac{1}{2} \left\| \sum_{d=1}^D \mathbf{x}^d * \mathbf{f}^d - \mathbf{y} \right\|^2, \quad (1)$$

where \mathbf{x} is the representation of the object patch with D features, \mathbf{y} is the pre-defined regression label, and $*$ is the feature-wise spatial convolution. The correlation between the learned template and the search region is implemented by the fast Fourier transform (FFT). It can be executed efficiently even on CPUs. Handcrafted features such as the histogram of oriented gradients (HOG) and colornames [34] are used for feature extraction. Yet, the performance of DCF trackers is inferior to that of supervised deep trackers by a significant margin.

UHP-SOT and UHP-SOT++. UHP-SOT was proposed in [5] to boost the performance of DCF trackers. It adopted a DCF tracker, STRCF [31], as the baseline and updated the template in each frame by

$$\arg \min_{\mathbf{f}} \left\{ \frac{1}{2} \left\| \sum_{d=1}^D \mathbf{x}_t^d * \mathbf{f}^d - \mathbf{y} \right\|^2 + \frac{1}{2} \sum_{d=1}^D \|\mathbf{w} \cdot \mathbf{f}^d\|^2 + \frac{\mu}{2} \|\mathbf{f} - \mathbf{f}_{t-1}\|^2 \right\}, \quad (2)$$

where \mathbf{w} is the spatial weight on the template to suppress background, \mathbf{f}_{t-1} is the template learned from time $t-1$, and μ is a constant regularization coefficient. With the STRCF baseline, two modules (i.e., background motion modeling and trajectory-based box prediction) were added for performance

improvement. UHP-SOT achieved a significant performance gain over STRCF on OTB-2015 [35]. Its enhanced version, UHP-SOT++ [6], has been tested on more datasets with further performance improvement. We adopt UHP-SOT++ as the baseline in GUSOT as elaborated in the next section.

III. PROPOSED GUSOT METHOD

An overview of the proposed GUSOT method is given in Fig. 1. GUSOT adds two new modules to the baseline UHP-SOT++ tracker for higher performance in long video tracking: 1) lost object recovery and 2) color-saliency-based shape proposal. While the baseline offers a box proposal (the red box), the recovery module yields a motion residual map and provides the second box proposal of the same size, called the motion proposal (the blue box). The two proposals are compared and the one with higher appearance similarity with a trusted template is chosen as the location proposal. Finally, another proposal, called the shape proposal, is used to adjust the shape of the location proposal to yield the final prediction (the yellow box). For the UHP-SOT++ baseline, we refer to [6]. The operations of the two new modules are detailed below.

Lost Object Recovery. When the baseline tracker works properly, the lost object recovery module simply serves as a backup. Yet, its role becomes critical when the baseline tracker fails. Recall that most trackers rely on similarity matching between the learned template \mathbf{f} at frame $(t-1)$ and the search region in frame t to locate the object. However, if the object gets lost due to tracking error or deformations, it is difficult to capture it again as the search region often drifts away from the true object location. Exhaustive search over the whole frame is infeasible in online tracking. Besides, the learned template could be contaminated during the object loss period and it cannot be used to search the object anymore. This module is used to find object locations of higher likelihood efficiently and robustly.

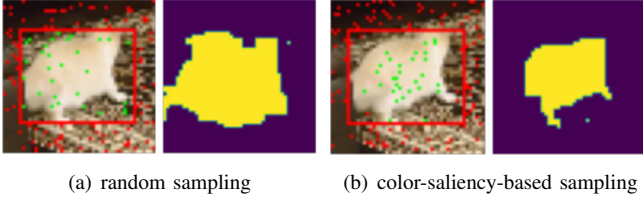


Fig. 2. Comparison of two sampling scheme and their segmentation results, where green and red dots represent foreground and background initial points, respectively. The red box is the reference box which indicates the object location.

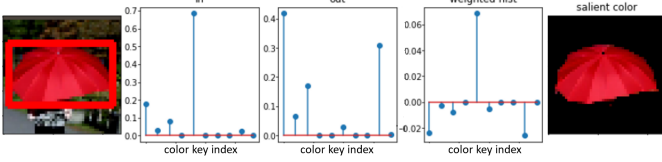


Fig. 3. Determination of salient color keys.

This module is built upon background motion estimation and compensation [5]. Based on the correspondence of sparsely sampled background salient points between frame $(t-1)$ and frame t , we can estimate the global motion field of the scene. Next, we can apply the motion field to all pixels in frame $(t-1)$, which is called the motion compensated frame, and find the difference between frame t and motion-compensated frame $(t-1)$, leading to a motion residual map. Afterwards, a box proposal, which is of the same size as that of frame $(t-1)$ and covering the largest amount of motion residuals, is computed. It serves as a good proposal for the object since the object can be revealed by residuals if 1) the object and its background take different motion paths and 2) background motion is compensated.

Now, we have two bounding box proposals: 1) the baseline proposal from UHP-SOT++ and 2) the motion proposal as discussed above. We need to assess their quality and select a better one. This is achieved by similarity measure. To avoid template degradation from tracking loss, we store a trusted template \mathbf{f}^* . It could be the initial template or a learned template from a high-confidence frame. Let \mathbf{x} be the candidate proposal. Then, the similarity measure between \mathbf{f}^* and \mathbf{x} can be defined as their correlation coefficient:

$$s_1(\mathbf{f}^*, \mathbf{x}) = \frac{\langle \mathbf{f}^*, \mathbf{x} \rangle}{\|\mathbf{f}^*\| \|\mathbf{x}\|}, \quad (3)$$

where $\langle \cdot, \cdot \rangle$ is the vector inner product, \mathbf{f}^* and \mathbf{x} are feature representations (rather than pixel values) of a trusted template or a candidate. Commonly used features are the histogram of oriented gradients (HOG) and colormaps. Here, we calculate a normalized histogram v of color keys in colormaps. Then, the similarity of two histograms is measured using the Chi-square distance:

$$s_2(\mathbf{f}^*, \mathbf{x}) = \sum_i \frac{(v_{\mathbf{f}^*, i} - v_{\mathbf{x}, i})^2}{v_{\mathbf{f}^*, i} + v_{\mathbf{x}, i}}. \quad (4)$$

The two histograms are more similar if the Chi-square distance is smaller. We replace the baseline proposal, \mathbf{x}_2 , with the motion proposal, \mathbf{x}_1 , if

$$s_1(\mathbf{f}^*, \mathbf{x}_1) > s_1(\mathbf{f}^*, \mathbf{x}_2) \text{ and } s_2(\mathbf{f}^*, \mathbf{x}_1) \leq s_2(\mathbf{f}^*, \mathbf{x}_2).$$

Otherwise, we keep the baseline proposal. For a faster tracking speed, the lost object recovery module is turned on only when the similarity score of the baseline proposal is low or the predicted object centers show abnormal trajectories (e.g., getting stuck to a corner).

Color-Saliency-Based Shape Proposal. Good shape estimation can benefit the tracking performance in the long run as it leads to tight bounding boxes and thus reduces noise in template learning. Deep trackers use the region proposal network to offer flexible boxes. Here, we exploit several low-cost segmentation techniques for box shape estimation in online tracking.

Given an image of size $W \times H$, the foreground/background segmentation is to assign binary labels $l_p = l(x, y) \in \{0, 1\}$ to pixels $p = (x, y)$. The binary mask, \mathbf{I} , can be estimated using the Markov Random Field (MRF) optimization framework:

$$\mathbf{I}^* = \arg \min_{\mathbf{I}} \sum_p \rho(p, l_p) + \sum_{\{p, q\} \in \mathbb{N}} w_{pq} \|l_p - l_q\|, \quad (5)$$

where $\rho(p, l_p)$ is the cost of assigning l_p to pixel p , which is defined as the negative log-likelihood of p in the Gaussian mixtures, \mathbb{N} is the four-connected neighborhood, w_{pq} is the weight of mis-matching penalty which is calculated as the Euclidean distance between p and q . The first term of Eq. (5) ensures that similar pixels get the same label while its second term forces label continuity among neighbors. The fast algorithm implemented in [36], [37] can work on a 32×32 image in 50ms. Thus, we crop a small patch centered at the predicted location and conduct coarse segmentation.

The MRF optimization problem is solved by iteration, which is to be initialized by a set of foreground/background points of good quality. Random sampling inside/outside the box results in noisy initialization as shown in Fig. 2(a). It tends to lead to undesired errors and/or longer iterations. To address it, we exploit salient foreground and background colors. With the predicted box and its associated image patch, all colors on the patch are quantized into N color keys. The distributions of color keys in and out of the object box, denoted by p_{in} and p_{out} , are calculated. Then, the color saliency score (CSS) of color keys k_i , $i = 1, \dots, N$, is the weighted difference between the two distributions:

$$\text{CSS}(k_i) = \frac{\sum_{j \neq i} \exp \|k_i - k_j\|^2}{Z} (p_{in}(k_i) - p_{out}(k_i)). \quad (6)$$

Z is the normalization factor for the sum of exponentials. The weights take the color difference into consideration and favor color keys standing out among all colors. If $\text{CSS}(k_i)$ is a positive (or negative) number of large magnitude, k_i is a foreground (or background) color key. A visualization example is provided in Fig. 3. The histogram bin number, N , is adaptively determined based on the first frame. It first

tries the color keys used in colormnames features and takes the corresponding number if there exists clear salient colors in this setting. Otherwise, clustering of all colors and merging of clusters are conducted to determine N . Finally, we sample points according to their color saliency score. As shown in Fig. 2, color-saliency-based sampling provides better initial points for iteration, thus leading to a better segmentation mask.

Complicated objects may not have clear salient colors. If the MRF optimization generates an abnormal output, we switch to a simple yet effective shape proposal by merging superpixels with guidance from motion and baseline. As shown in Fig. 4, after superpixel segmentation [38], we assign binary labels to superpixels and find the enclosing box, B_s , for the foreground superpixels. With different label assignments, different B_s can be generated.

Output Proposal. Let B_b , B_m and B_s denote the baseline proposal, the motion proposal and the shape proposal, respectively. Usually, we stick to baseline proposal B_b if its similarity score is high. We will only consider B_m , B_s , and B^* if the baseline gets a low similarity score. When this happens, we compute the following proposal as the predicted tracking box at frame t :

$$B^* = \arg \max_{B_s} \text{IoU}(B_s, B_b) + \text{IoU}(B_s, B_m), \quad (7)$$

where IoU is the intersection-over-union between two boxes.

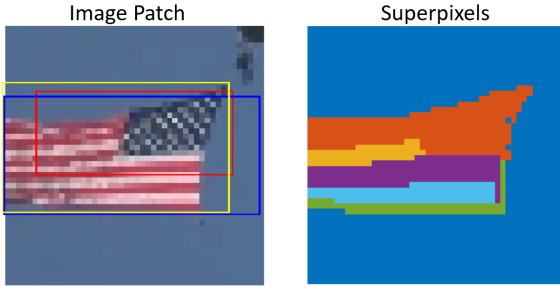


Fig. 4. Illustration of shape proposal derivation based on superpixel segmentation, where the red, blue and yellow boxes correspond to the baseline, motion, and shape proposals, respectively. The size of the motion proposal here is determined by clipping integral curves horizontally and vertically on the motion residual map.

IV. EXPERIMENTS

Experimental Setup. We conduct experiments on the large-scale single object tracking dataset LaSOT [7]. It has 280 long test videos of around 685K frames. Evaluation metrics for tracking performance include: 1) the distance precision (DP) measured at the 20-pixel threshold and 2) the area-under-curve (AUC) score for the overlap precision. We use the same hyperparameter settings for the baseline tracker. The segmentation module is activated when the appearance score of the baseline tracker is less than 0.2. Almost all template matching trackers can provide the appearance score for an object proposal. It is simply the correlation score in DCF-trackers. The patch size used in segmentation is 48×48 . The superpixel segmentation

TABLE I
COMPARISON OF UNSUPERVISED AND SUPERVISED TRACKERS ON LASOT, WHERE S AND P INDICATE SUPERVISED AND PRE-TRAINED, RESPECTIVELY. BACKBONE DENOTES THE PRE-TRAINED FEATURE EXTRACTION NETWORK.

	S	P	DP	AUC	GPU	Backbone
ECO-HC [8]	×	×	27.9	30.4	×	N/A
STRCF [31]	×	×	29.8	30.8	×	N/A
UHP-SOT++ [6]	×	×	32.9	32.9	×	N/A
LUDT [39]	×	✓	-	26.2	✓	VGG [40]
USOT [3]	×	✓	32.3	33.7	✓	ResNet50 [41]
ULAST [4]	×	✓	40.7	43.3	✓	ResNet50 [41]
SiamFC [19]	✓	✓	33.9	33.6	✓	AlexNet [42]
ECO [8]	✓	✓	30.1	32.4	✓	VGG [40]
SiamRPN [20]	✓	✓	38.0	41.1	✓	AlexNet [42]
GUSOT (Ours)	×	×	36.1	36.8	×	N/A

exploits a Gaussian blur of $\sigma_{blur} = 0.6$ and the minimal superpixel size is set to 50.

Performance Evaluation. We compare tracking accuracy and model complexity of GUSOT against state-of-the-art supervised and unsupervised trackers in Table I. We have the following observations. First, the improvement over the UHP-SOT++ baseline is around 10% in DP and 12% in AUC. It demonstrates the capability of GUSOT in handling tracking loss and shape deformation in long video sequences. Second, GUSOT outperforms all previous DCF-trackers (e.g., supervised ECO and unsupervised ECO-HC and STRCF) in the first group by a large margin. Third, GUSOT outperforms two unsupervised deep trackers with pre-training (e.g. LUDT and USOT) in the second group. Fourth, there is a significant performance gap between GUSOT and the latest unsupervised deep tracker ULAST. Yet, the latter has to be pre-trained by a large amount of video data with pseudo labels generated from optical flows. Its large feature backbone, ResNet-50, could be heavy for small devices. Finally, GUSOT surpasses two supervised deep trackers (i.e., SiamFC and ECO) in the last group, narrowing the performance gap to the supervised deep tracker SiamRPN.

Six examples are selected and shown in Fig. 5 for qualitative comparison. UHP-SOT++ does not perform well in any of them. In contrast, GUSOT offers the best performance in all six cases. It shows the power of the two newly added modules. GUSOT offers flexible box shapes, yield robust tracking against occlusion and fast motion, and works well on small objects (e.g., yoyo) and large objects (e.g., person). USOT could be distracted by background clutters (see pool, umbrella, yoyo and person). SiamFC also fails in three right subfigures (e.g., airplane, person and yoyo).

We also compare the DP and AUC scores of GUSOT, UHP-SOT++ and USOT against various tracking attributes in Fig. 6. While USOT performs best in deformation due to its box regression neural network, GUSOT has leading performance in all other attributes. The improvement of GUSOT over UHP-SOT++ is more obvious in fast motion, out-of-view and viewpoint change, which aligns well with the contributions of the lost object recovery and the color-saliency-based shape

REFERENCES

- [1] G. Wang, Y. Zhou, C. Luo, W. Xie, W. Zeng, and Z. Xiong, "Unsupervised visual representation learning by tracking patches in video," in *Proceedings of IEEE/CVF Computer Vision and Pattern Recognition*, 2021, pp. 2563–2572.
- [2] Q. Wu, J. Wan, and A. B. Chan, "Progressive unsupervised learning for visual object tracking," in *Proceedings of IEEE/CVF Computer Vision and Pattern Recognition*, 2021, pp. 2993–3002.
- [3] J. Zheng, C. Ma, H. Peng, and X. Yang, "Learning to track objects from unlabeled videos," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 13 546–13 555.
- [4] Q. Shen, L. Qiao, J. Guo, P. Li, X. Li, B. Li, W. Feng, W. Gan, W. Wu, and W. Ouyang, "Unsupervised learning of accurate siamese tracking," in *Proceedings of IEEE/CVF Computer Vision and Pattern Recognition*, 2022, pp. 8101–8110.
- [5] Z. Zhou, H. Fu, S. You, C. C. Borel-Donohue, and C.-C. J. Kuo, "UHP-SOT: An unsupervised high-performance single object tracker," in *2021 International Conference on Visual Communications and Image Processing (VCIP)*. IEEE, 2021, pp. 1–5.
- [6] Z. Zhou, H. Fu, S. You, and C.-C. J. Kuo, "Unsupervised lightweight single object tracking with UHP-SOT++," *arXiv:2111.07548*, 2021.
- [7] H. Fan, L. Lin, F. Yang, P. Chu, G. Deng, S. Yu, H. Bai, Y. Xu, C. Liao, and H. Ling, "Lasot: A high-quality benchmark for large-scale single object tracking," in *Proceedings of IEEE/CVF Computer Vision and Pattern Recognition*, 2019, pp. 5374–5383.
- [8] M. Danelljan, G. Bhat, F. Shahbaz Khan, and M. Felsberg, "Eco: Efficient convolution operators for tracking," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 6638–6646.
- [9] M. Danelljan, A. Robinson, F. S. Khan, and M. Felsberg, "Beyond correlation filters: Learning continuous convolution operators for visual tracking," in *European conference on computer vision*. Springer, 2016, pp. 472–488.
- [10] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang, "Hierarchical convolutional features for visual tracking," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 3074–3082.
- [11] Y. Qi, S. Zhang, L. Qin, H. Yao, Q. Huang, J. Lim, and M.-H. Yang, "Hedged deep tracking," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4303–4311.
- [12] G. Bhat, J. Johnander, M. Danelljan, F. S. Khan, and M. Felsberg, "Unveiling the power of deep tracking," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 483–498.
- [13] N. Wang, W. Zhou, Q. Tian, R. Hong, M. Wang, and H. Li, "Multi-cue correlation filters for robust visual tracking," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4844–4853.
- [14] Y. Sun, C. Sun, D. Wang, Y. He, and H. Lu, "Roi pooled correlation filters for visual tracking," in *Proceedings of IEEE/CVF on Computer Vision and Pattern Recognition*, 2019, pp. 5783–5791.
- [15] X. Lu, C. Ma, B. Ni, X. Yang, I. Reid, and M.-H. Yang, "Deep regression tracking with shrinkage loss," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 353–369.
- [16] H. Nam and B. Han, "Learning multi-domain convolutional neural networks for visual tracking," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4293–4302.
- [17] S. Pu, Y. Song, C. Ma, H. Zhang, and M.-H. Yang, "Deep attentive tracking via reciprocative learning," *arXiv preprint arXiv:1810.03851*, 2018.
- [18] Y. Song, C. Ma, L. Gong, J. Zhang, R. W. Lau, and M.-H. Yang, "Crest: Convolutional residual learning for visual tracking," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2555–2564.
- [19] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr, "Fully-convolutional siamese networks for object tracking," in *European conference on computer vision*. Springer, 2016, pp. 850–865.
- [20] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu, "High performance visual tracking with siamese region proposal network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8971–8980.
- [21] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan, "Siamrpn++: Evolution of siamese visual tracking with very deep networks," in *Proceedings of IEEE/CVF Computer Vision and Pattern Recognition*, 2019, pp. 4282–4291.
- [22] R. Tao, E. Gavves, and A. W. Smeulders, "Siamese instance search for tracking," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1420–1429.
- [23] Z. Zhu, Q. Wang, B. Li, W. Wu, J. Yan, and W. Hu, "Distractor-aware siamese networks for visual object tracking," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 101–117.
- [24] N. Wang, W. Zhou, J. Wang, and H. Li, "Transformer meets tracker: Exploiting temporal context for robust visual tracking," in *Proceedings of IEEE/CVF Computer Vision and Pattern Recognition*, 2021, pp. 1571–1580.
- [25] X. Chen, B. Yan, J. Zhu, D. Wang, X. Yang, and H. Lu, "Transformer tracking," in *Proceedings of IEEE/CVF Computer Vision and Pattern Recognition*, 2021, pp. 8126–8135.
- [26] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *2010 IEEE computer society conference on computer vision and pattern recognition*. IEEE, 2010, pp. 2544–2550.
- [27] 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*, vol. 37, no. 3, pp. 583–596, 2014.
- [28] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg, "Convolutional features for correlation filter based visual tracking," in *Proceedings of the IEEE international conference on computer vision workshops*, 2015, pp. 58–66.
- [29] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg, "Discriminative scale space tracking," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 8, pp. 1561–1575, 2016.
- [30] L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik, and P. H. Torr, "Staple: Complementary learners for real-time tracking," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1401–1409.
- [31] F. Li, C. Tian, W. Zuo, L. Zhang, and M.-H. Yang, "Learning spatial-temporal regularized correlation filters for visual tracking," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4904–4913.
- [32] T. Xu, Z.-H. Feng, X.-J. Wu, and J. Kittler, "Learning adaptive discriminative correlation filters via temporal consistency preserving spatial feature selection for robust visual object tracking," *IEEE Transactions on Image Processing*, vol. 28, no. 11, pp. 5596–5609, 2019.
- [33] Y. Li, C. Fu, F. Ding, Z. Huang, and G. Lu, "Autotrack: Towards high-performance visual tracking for uav with automatic spatio-temporal regularization," in *Proceedings of IEEE/CVF Computer Vision and Pattern Recognition*, 2020, pp. 11 923–11 932.
- [34] M. Danelljan, F. Shahbaz Khan, M. Felsberg, and J. Van de Weijer, "Adaptive color attributes for real-time visual tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1090–1097.
- [35] Y. Wu, J. Lim, and M.-H. Yang, "Object tracking benchmark," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1834–1848, 2015.
- [36] Q. Chen and V. Koltun, "Fast mrf optimization with application to depth reconstruction," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 3914–3921.
- [37] Z. Zhang, "A python implementation of block gradient decent for optimizing markov random fields," https://github.com/zhzhang/MRF_BCD, 2019.
- [38] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *International journal of computer vision*, vol. 59, no. 2, pp. 167–181, 2004.
- [39] N. Wang, W. Zhou, Y. Song, C. Ma, W. Liu, and H. Li, "Unsupervised deep representation learning for real-time tracking," *International Journal of Computer Vision*, vol. 129, no. 2, pp. 400–418, 2021.
- [40] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," *arXiv preprint arXiv:1405.3531*, 2014.
- [41] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [42] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.