Robust Visual Tracking Using Multiple Detectors by Trajectory Entropy Minimization

Wei Quan1, Yueping Liu1*, Xuemin Lu1*, Tianrui Li1, Ye Wang1, Chengjun Jia1, Jim X. Chen2

1 Southwest Jiaotong University, Chengdu, Sichuan 610031, China
2 George Mason University, Fairfax, Virginia 22030, USA
*Corresponding author. E-mail addresses: daliuyueping@163.com (Yueping Liu), b1160283279@qq.com (Xuemin Lu).

Abstract. We propose a tracking method based on minimization of multi-detector trajectory entropy. Multiple detectors are constructed online to track objects synchronously by mutual information-based feature selection. Each detector obtains its own tracking trajectory through continuous detection. The tracking results of all detectors are combined based on the minimization of trajectory entropy, and the optimal detector is selected from it to determine the target trajectory and achieve the target tracking task. Experimental results show that our tracker can handle more complex tracking environments and outperform many other state-of-the-art methods in terms of both success rate and precision.

1. Introduction

Visual tracking is one of the important research problems in the field of computer vision, with wide applications including intelligent surveillance, human-computer interaction and automatic control system, etc [1-3]. The core of tracking is to automatically determine its position and size in each subsequent frame based on the position and size of the given initial object. Although visual tracking has achieved many important progress in recent years [4-7], it still has great challenges in real-world scenes, such as background clutters, appearance changes and illumination variations, low image quality and frame hopping. An ideal tracking method should also consider three factors, namely, real-time, robustness and accuracy.

Visual tracking can be classified as two types: Generative model and Discriminative model. Generative method is searching the object that is most similar to the tracked target in the region [8-10], and Discriminative method is a classification problem which distinguishes the tracked target from the background [11-13]. Discriminative trackers utilize information from target and background both, which is also the main method to study visual tracking. Zhang et al. [14] proposed a tracking approach that was combined multiple classifiers to detect objects by using different adaptation rates and an entropy computing method was designed to fuse all the detection results for tracking. For the problem of the variations in scale and translation of moving objects, Ma et al. [15] and Martin et al. [16,17] decomposed the tracking task into object translation and scale estimation, and improved the accuracy and robustness of visual tracking based on correlation filters. However, these methods are still difficult to adapt to the more complex environments, which provides a possible solution to the analysis and processing of
tracking trajectory. Bertinetto et al. [18] proposed a feature fusion tracking algorithm, which made use of the complementarity between local feature HOG and global feature COLOR to improve tracking performance without affecting the tracking speed. In order to make a long-term tracking, Kalal et al. [19] proposed a P-N learning method based on online learning with positive samples and negative samples. Zhou et al. [20] proposed novel visual tracking algorithm based on an online discriminative dictionary learning technique to separate the target-of-interest from a cluttered background effectively. Lu et al. [21] and Supancic et al. [22] used dynamic programming to evaluate the track, including the calculation of each position in the track and the time domain correlation of continuous position between each other, then modified trajectory tracking and updated the appearance model according to the evaluation results, so as to achieve the purpose of inhibiting the propagation of errors. Lee et al. [23] tracked objects and generated multiple trace trajectories based on a variety of features, and calculated the reliability of each trajectory by the analysis of these forward trajectories and the corresponding backward trajectories, to choose the optimal trajectory as the target trajectory. To a certain extent, this tracking method detected and processed tracking errors, and improved the stability of tracking.

This paper proposed an object tracking method based on the minimization of trajectory entropy of multiple detectors. We constructed multiple detectors online by the mutual information feature selection, and each detector generated its own tracking trajectory by continuous detection, then by the method of minimization of trajectory entropy, the optimal detector was selected to determine the trajectory and achieve visual tracking. The differences between our work and Refs [14] are: (1) New detectors are constructed by feature selection, rather than preserving the states of the same detector at different times, thus improving the detector’s adaptability. (2) Each fern has a corresponding weight in each detector, and its size is directly proportional to the mutual information of the detector. By this method, the accuracy of detector to object detection is improved; (3) This paper proposes a method of trajectory entropy, which is calculated by reliability of positive and backward tracking trajectory. Experimental results based on real video sequences validate the effectiveness and stability of our tracker and our tracking performance is encouraged by comparing against state-of-the-art trackers.

2. Tracker using multiple detectors by trajectory entropy minimization

2.1. Feature selection and detector construction based on mutual information

In this paper, the random fern [24] is used as the basic detector, and the two-point comparison feature is selected as the two-element of the detector. Compared with other classifier algorithms, the greatest advantage of random fern is simple construction, rapid training and detection, which is conducive to real-time tracking. However, two-element feature is randomly selected in most existing random fern methods. In order to achieve better classification results, more features and ferns need to be combined in these methods. In fact, those features that can discriminate objects and backgrounds well have relatively small Bayesian classification error rate. Therefore, a simpler and more effective fern can be obtained by adopting the appropriate feature selection method, which can be used to construct a more accurate random fern detector. This paper selects features based on the mutual information between feature response and type tags, and the mutual information can estimate the expected probability of correct classification of two-element classifier, so it can be used to measure the discriminant classification ability of the feature.

Let \( r \) represent the object position, \( W^T_r \) and \( W^B_r \) denote the object window containing the target object and the background window which is surrounded by the object, respectively. The feature set \( F \) predefined feature space \( V \) is obtained by extracting the features from both window regions. Let the probability of the type of the feature \( f \) in the target window be \( p(f \mid c = 1) \), and the probability of the type of the feature in the background window is \( p(f \mid c = 0) \), then the mutual information between features \( f \) and type tags \( c \) is calculated as:
\[ I(f; c) = \sum_c p(f, c) \log \frac{p(f, c)}{p(f) p(c)}, \quad (1) \]

where \( f \in \{0, 1\} \) is a two-element feature, \( p(f) = 0.5 \) represents that the probability of feature value is equal, \( c \in \{0, 1\} \) represents that object type. \( c = 1 \), background type \( c = 0 \), and \( p(c) = 0.5 \) represents that the probability of type tagging is equal. According to the probability formula, \( p(f, c) = p(f) p(c | f) \), \( p(c = 1 | f) \) can be calculated as:

\[ p(c = 1 | f) = 1 - p(c = 0 | f) = \frac{N_T}{N_T + N_B}, \quad (2) \]

where \( N_T \) represents the number of samples in which the value of the feature \( f \) in the background area is different from its value in the object area, \( N_B \) represents the number of samples in which the value of the feature \( f \) in the background area is consistent with its value in the object area. After calculating the mutual information of all the features in the feature set, then the features are sorted in ascending order of mutual information and the first \( M \) features are selected to construct a random fern detector. The detector training is then completed by extracting positive samples in the object area and negative samples in the background area and then used for object detection. The feature selection and detector construction for object detection is shown in Fig.1, in which the purple, green and blue dots in the red rectangle corresponding to the object indicate the selected pixel comparison feature, and the image blocks with red border and green border in the black rectangle represent the positive and negative samples extracted from the image, respectively.

For the object detection, the paper sets the weight for each fern in the random ferns according to the discriminant classification ability of the features, that is, the fern's discriminative classification ability is evaluated by its contained features. Let \( H_M = \{f_i\}_{i=1}^M \) denote the feature set for detector construction obtained by feature selection, and let \( B \) and \( D \) denote the number of ferns in the detector and the number of features contained in fern, respectively. And there is a numerical relationship: \( B \times D = M \). From \( H_M \) we select \( D \) number of features to construct fern and obtain \( B \) number of ferns, then the conditional probability of the detector about the object type is calculated as:

\[ P(H_M | c = 1) = \frac{1}{B} \sum_{k=1}^B W_k P(R_k | c = 1), \quad (3) \]

where \( R_k = \{f_{j + D(k-1)}\}_{j=1}^D \) denotes the k-th fern, and \( f_{j + D(k-1)} \) is the j-th feature, which corresponds to the \( j + D(k-1) \)-th feature of \( H_M \), the weight of \( R_k \) is \( W_k = \left(\sum_{j=1}^D I(f_{j + D(k-1)}; c)/G\right) \), in which the normalized factor is \( G = \sum_{j=1}^M I(f_j; c) \). Therefore, the greater the mutual information of the feature in the fern, the greater the weight of the fern.
2.2. Detector selection based on trajectory entropy minimization

For the same object, different detectors will generate different tracking trajectories, and at the same time corresponding to different object positions. Let \( Q = \{ F_j \}_{j=1}^N \) represents detector set, \( \varphi_{F_j} \) represents loss function of \( F_j \), where \( F_j \) and \( N \) represent \( j \)-th detector and the number of detectors respectively. Then the optimal detector is calculated as:

\[
F^* = \arg \min_{F_j \in Q} \varphi_{F_j}, \quad (4)
\]

in other words, the detector with the smallest loss value is selected.

In this paper, Partial-Label Learning (PLL) [25] is used to solve the loss function, which is to solve the problem of partial mark learning by maximizing the posterior probability of the model parameters. For more details, please refer to literature [25]. Let \( S = \{ X_i \}_{i=1}^N \) denote a set of trajectories, \( X_i = \{ x_{ik} \}_{k=1}^H \) denote the \( i \)-th tracking trajectory, where \( x_{ik} \) is the position of the \( k \)-th frame in the included time segment of \( X_i \), \( H \) is the number of frames contained in the trajectory. Let \( y_i = (c_i, v_i) \) denote compound mark of \( X_i \), \( c_i \in \{1, 0\} \) (\( c_i = 1 \) represents that \( X_i \) is object trajectory, \( c_i = 0 \) represents that the non-object trajectory) denote track mark, \( v_i \) denote the end position of the track of \( X_i \). \( Y = \{ y_{ij} \}_{i,j=1}^N \)——compound mark set of object real trajectories of \( S \) where may only be one object real trajectory of all the tracking trajectories is contained in \( Z = \{ Y_k \}_{k=1}^M \)——the possible composite mark set, where there is \( Y_k = \{ y_{ik} \}_{i=1}^N = \{ (c^k_i, v^k_i) \}_{i=1}^N \), there is \( v^k_i = v_i \), and \( c^k_i = 1 \) when \( i = k \). For the tracking problem, the loss function for the detector selection can be calculated as:

\[
\phi_{F} (S, Z) = -L(F_j; S, Z) + \lambda H(Y \mid S, Z; F_j), \quad (5)
\]

where \( L = (F_j; S, Z) \) represents the log-likelihood probability whose the model parameter is \( F_j \), \( H(Y \mid S, Z; F_j) \) represents the type-mark empirical condition entropy with respect to the training data and the mark set, and \( \lambda \) is the ratio between the two coefficients. For the mark set, the model parameters with lower uncertainty also have smaller entropy. In other words, if one model has a higher probability of one of the markers and a smaller probability of the other, the model will have a lower entropy; and if both of the two markers of a model have the same probability, the model will have a larger entropy. Here the log likelihood is defined as:
\[ L(F_j; S, Z) = \max_{Y \in Z} \log p(Y \mid S; F_j) , \]  
entrophy is calculated as:

\[ p(Y \mid S; F_j) = \prod_i p(c_i, v_i \mid X_i; F_j) = \prod_i p(v_i \mid c_i)p(c_i \mid X_i; F_j) , \]  
and \( p(Y \mid S; F_j) \) can be calculated as:

\[ p(Y \mid S; F_j) = \prod_i p(c_i, v_i \mid X_i; F_j) = \prod_i p(v_i \mid c_i)p(c_i \mid X_i; F_j) , \]  
where \( p(v_i \mid c_i) = p(v_i \mid c_i, X_i) \) is the spatial prior probability of the trajectory and \( p(c_i \mid X_i; F_j) \) is the posterior probability of the trajectory. \( p(v_i \mid c_i) \) is defined as the average value of the probability of the detector classification represented by the image block corresponding to each image block in the trajectory, calculated as \( p(v_i \mid c_i) = \frac{1}{H} \sum_{k=1}^H \rho_k^i \), where \( \rho_k^i \) is the classification probability of the detector in the k-th frame of the track. And \( p(c_i \mid X_i; F_j) \) is obtained from the trajectory reliability based on the image block overlap rate, calculated as:

\[ p(c_i \mid X_i; F_j) = \sum_{k=1}^H \delta_k^j \]  
where \( \delta_k^j \) is image block overlap rate between the forward tracking trajectory \( \{x_i^j\}_{k=1}^H \) and reverse tracking trajectory \( \{\overline{x}_i^j\}_{k=1}^H \) at the k-th frame, calculated as:

\[ \delta_k^j = \frac{\Omega(x_i^j, \overline{x}_k^j)}{\Omega(x_i^j) + \Omega(\overline{x}_k^j)} , \]  
\( \Omega(x_i^j) \) and \( \Omega(\overline{x}_k^j) \) represent the area of the image block of \( x_i^j \) and \( \overline{x}_k^j \) at the k-th frame, respectively, \( \Omega(x_i^j, \overline{x}_k^j) \) represent the area of overlap of two image blocks at the k-th frame. Finally, \( p(Y \mid S, Z; F_j) \) is calculated as the Kullback-Leibler projection on \( p(Y \mid S; F_j) \):

\[ p(Y \mid S, Z; F_j) = \frac{\delta_k^j p(Y \mid S; F_j)}{\sum_{Y \in \psi} \delta_k(Y) p(Y \mid S; F_j)} , \]  
if \( Y \in Z \), then \( \delta_k^j(Y) = 1 \), else \( \delta_k^j(Y) = 0 \), \( \psi \) denotes the vector space where \( Y \) and \( Z \) are. The optimal detector can be obtained by formula (5).

### 2.3. Object Tracking

Based on section 2.1, this paper constructs multiple detectors online, which locate the objects simultaneously and generate their own tracking trajectories in a continuous detection method. After the initial construction of the first detector, a new detector is constructed every other frame, which is characterized by reselection, and the earliest constructed detector is automatically deleted when the number of detectors is exceeded \( \theta_y \), that is to say, the number of detectors can not exceed \( \theta_y \). All detectors generate the tracking trajectory with a length of \( \theta_y \) frames every \( \theta_y \) frame. The optimal detector can be calculated according to section 2.2, and the corresponding tracking trajectory is taken as the object trajectory in this period. Before continuing tracking, all detectors are incrementally updated based on the new positive and negative samples taken from the determined object trajectory. The multi-detector trajectory entropy minimization object tracking process is shown in Fig.2, where the blue boxes
represent the online-constructed detector, the image block above the detector and the black curve below represent the corresponding object appearance when detector is constructed and the resulting tracking trace, respectively. The red boxes and the red curves represent the detector selection and the corresponding object trace respectively. The scale of object is determined by calculating the similarity of candidate object image blocks of different scales based on NCC (Normalized Cross-Correlation) and selecting the optimal image block from them. More specifically, for object of each frame in the current object track, let the candidate object sizes are 70%, 80%, 90%, 100%, 110%, 120%, 130% of original size, and then calculate the average NCC image block similarity between each candidate object and all the objects contained in the previous object trajectory. Finally, the candidate object with the maximum average similarity is used as the corresponding object in the current object trajectory.

Figure 2. Tracking process using trajectory minimization of entropy by multi-detector

3 Experimental results
The feature set of the paper contains 200 randomly generated two-point comparison features initially, and M = 25 features are selected by feature selection for the construction of random fern detectors, each detector contains B = 5 ferns, and each fern contains D = 5 features. Training and updating of detectors is based on the positive and negative samples of the normalized 25 × 25 pixel size, which are extracted from the object area and the background area in the same size as the object size. The experimental results show that the processing time of the tracking system increases rapidly with the increase of the number of detectors θₙ. However, the stability of the tracking system increases slightly after θₙ = 5. Therefore, considering the computational efficiency and tracking stability, let θₙ = 5 here. In the implementation of this paper, θₙ - the length of the tracking trajectory is 15, θ₅ₙ - the time interval for constructing a new detector is 45, and the scaling factor λ in the loss function is 10. For 320 × 240 images, the average speed of the tracking system based on C# multithreaded programming is 5fps on an ordinary dual core PC. The video sequence used in the experiment comes from Babenko [26] sequence, which is widely used in the world, and Jumping, Pedestrian 1, Motocross and Carchase sequences.

In this paper, the tracking method is tracked and tested. Fig.3 shows the tracking quantification results of the proposed method under the representative David sequence. It can be seen that the proposed method maintains a high tracking stability during the tracking process compared with other tracking algorithms. Here the tracking stability is the percentage of objects that are tracked correctly in the whole video sequence. Specifically, if 30% of the tracking rectangle in a frame does not contain the object or 30% of the object is not included in the tracking rectangle, the frame is considered as tracking error; otherwise, the tracking is correct. Other tracking algorithms used for comparison include OAB[27], ORF[28], FT[29], MILT[26], CDCT[30], PNT[31] and CT[32] and MEEM[14]. We list the statistical results of the tracking method based on Babenko sequences in Tab.1, where the best tracking results are underlined in bold and the second-best results are underlined in italics. It is Fig.4 that shows the average
center error of each tracking method under the Babenko sequence. The sequence of the video sequences is the same as that in Table 1. It can be seen that the method in this paper is generally superior to other tracking algorithms. Fig.5 shows the tracking results (example frame images) of the proposed method under Babenko et al. sequence, where the red box indicates the object of tracking. The Babenko sequence is used to track baselines for algorithm comparisons. The Jumping and Pedestrian 1 sequences are used to test the ability to track fast-moving objects while the Motocross and Carchase sequences are used to test the ability to perform long-term tracking.

At present, the memory of the tracking system based on C# is about 75M at initialization and 120M in the tracking process. Compared with the representative literatures [21] and [32], the tracking system memory based on C++ is 150M and 110M. It can be concluded that the method is one order of magnitude better than other methods in terms of memory usage in this paper. Tab.2 lists the tracking efficiency of each method (average frame rate). Because in terms of the tracking method, on the one hand, multiple detectors are constructed at different times by feature selection to adapt to the change of the object, on the other hand, we can select the trajectory minimization of object generated by these detectors and from which chose the optimal tracking trajectory to achieve the tracking performance improvement.

Table 1. Percentage of tracking correctly using each method under Babenko and other sequences

| Video Sequence | Frames | OAB | ORF | FT  | MILT | CDCT | PNT  | CT   | MEEM | Ours |
|----------------|--------|-----|-----|-----|------|------|------|------|------|------|
| Coke           | 292    | 18.4| 17.1| 10.6| 46.1 | 43.2 | 41.3 | 44.5 | 64.8 | 69.3 |
| David          | 462    | 30.8| 87.2| 73.8| 82.7 | 79.5 | 90.5 | 92.2 | 95.7 | 98.6 |
| Face 1         | 888    | 72.9| 97.5| 97.5| 89.6 | 93.7 | 95.6 | 96.8 | 98.0 | 99.3 |
| Face 2         | 820    | 76.3| 68.9| 63.7| 92.9 | 93.3 | 94.9 | 94.9 | 97.3 | 99.0 |
| Girl           | 502    | 45.8| 92.6| 87.4| 84.2 | 91.6 | 94.3 | 95.6 | 96.2 | 96.2 |
| Sylvester      | 1345   | 56.1| 69.2| 82.6| 81.4 | 83.1 | 85.7 | 87.9 | 91.4 | 94.8 |
| Tiger 1        | 354    | 31.7| 27.1| 26.5| 73.9 | 73.9 | 67.4 | 74.8 | 80.9 | 84.9 |
| Tiger 2        | 364    | 35.5| 20.7| 23.5| 76.5 | 46.5 | 71.8 | 83.7 | 85.8 | 89.5 |
| Jumping        | 313    | 57.3| 68.5| 48.2| 75.6 | 73.5 | 80.7 | 87.6 | 87.6 | 92.9 |
| Pedestrian 1   | 140    | 53.6| 65.9| 44.9| 72.1 | 69.8 | 19.3 | 84.3 | 86.0 | 88.7 |
| Motocross      | 2665   | 2.5 | 2.3 | 5.15| 18.1 | 81.9 | 89.9 | 92.5 | 93.7 | 95.8 |
| Carchase       | 5149   | 6.8 | 7.1 | 6.9 | 6.9  | 72.6 | 83.5 | 86.8 | 88.7 | 89.4 |

Table 2. Each method tracks efficiency (Average frame rate, unit: frame / second)

| OAB | ORF | FT  | MILT | CDCT | PNT | CT  | MEEM | Ours |
|-----|-----|-----|------|------|-----|-----|------|------|
| 13  | 10  | 5   | 11   | 8    | 12  | 10  | 9    | 5    |
4. Conclusion

In this paper, we proposed a tracking method based on minimization of multi-detector trajectory entropy. The results show that the proposed method can handle more complex tracking environments and our tracker outperforms many other state-of-the-art methods. As the future work, we would like to study more refined trajectory analysis methods so that we can obtain more stable and fast-tracking performance with less image information.

References

[1] D.Y. Lee, J.Y. Sim, and C.S. Kim. Visual tracking using pertinent patch selection and masking. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), pp. 3486–3493, 2014.
[2] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), pp. 2411–2418, 2013.
[3] F. Henriques, R. Caseiro, P. Martins, and J. Batista. Exploiting the Circulant Structure of Tracking-by-Detection with Kernels. 2012
[4] S. Hare, A. Saffari, and P. H. S. Torr. Struck: structured output tracking with kernels. IEEE Int’l Conf. Computer Vision (ICCV), pp. 263-270, 2011.
[5] J. Fan, X. Shen, and Y. Wu. Scribble tracker: a matting-based approach for robust tracking. IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 34, no. 8, pp. 1633–1644, 2012.
[6] Hare S, Golodetz S, Saffari A, et al. Struck: Structured output tracking with kernels[J]. IEEE TPAMI, 2016.
[7] Danelljan M, Bhat G, Khan F S, et al. ECO: Efficient Convolution Operators for Tracking [C]. Computer Vision and Pattern Recognition (CVPR), 2017.
[8] V. Belagiannis, F. Schubert, N. Navab, and S. Ilic. Segmentation based particle filtering for real-time 2d object tracking. European Conference on Computer Vision (ECCV), pp. 842–855, 2012.
[9] J. Xing, J. Gao, B. Li, W. Hu, and S. Yan. Robust object tracking with online multi-lifespan dictionary learning. IEEE Int’l Conf. Computer Vision (ICCV), pp. 665–672, 2013.
[10] Vojir T, Noskova J, Matas J. Robust scale-adaptive mean-shift for tracking [J]. Pattern Recognition Letters, 2014
[11] Z. Zuo, G. Wang, B. Shuai, L. Zhao, and Q. Yang. Exemplar based deep discriminative and shareable feature learning for scene image classification. Pattern Recognition, vol. 48, no. 10, pp. 3004-3015, 2015.
[12] Goded, M., Sternig, S., Roth, P.M., Bischof, H. Context-driven clustering by multi-class classification in an active learning framework. IEEE Conf. Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 19–24, 2010.
[13] LI Kang, HE Fa-zhi, PAN Yi-teng, SUN Hang. Multi-Classifier Object Tracking Based on Cluster
Similarity. Acta Electronica Sinica, 2016, 44(4): 821-825.

[14] J. Zhang, S. Ma, and S. Sclaroff. MEEM: robust tracking via multiple experts using entropy minimization. European Conference on Computer Vision (ECCV), vol. 8694, pp. 188-203, 2014.

[15] C. Ma, X. Yang, C. G. Zhang, and M. H. Yang. Long-term Correlation Tracking. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), pp. 5388-5396, 2015.

[16] Bertinetto L, Valmadre J, Henriques J F, et al. Fully-convolutional siamese networks for object tracking [C]. European Conference on Computer Vision (ECCV), 2016.

[17] David Held, Sebastian Thrun, Silvio Savarese. Learning to Track at 100 FPS with Deep Regression Networks [C]. European Conference on Computer Vision (ECCV), 2016.

[18] Bertinetto L, Valmadre J, Golodetz S, et al. Staple: complementary learners for real-time tracking[C]/Proceedings of IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas, NV, United States: IEEE, 2016:1401-1409.

[19] Z. Kalal, K. Mikolajczyk, and J. Matas. Tracking-learning-detection. IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 34, no. 7, pp. 1409–1422, 2012.

[20] Tao Zhou, Fanghui Liu."Online discriminative dictionary learning for robust object tracking." Neurocomputing 275: 1801-1812, 2018.

[21] J. Supancic and D. Ramanan. Self-paced learning for long-term tracking. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), pp. 2379–2386, 2013.

[22] Y. Lu, T. Wu, and S.-C. Zhu. Online object tracking, learning, and parsing with and-or graphs. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), pp. 3462–3469, 2014.

[23] D. Lee, J. Sim and C. Kim. Multihypothesis trajectory analysis for robust visual tracking. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), pp. 5088 – 5096, 2015.

[24] M. Ozuysal, P. Fua, and V. Lepetit. Fast keypoint recognition in ten lines of code. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), pp. 1-8, 2007.

[25] Y. Grandvalet, and Y. Bengio. Semi-supervised learning by entropy minimization. Neural Information Processing Systems (NIPS), vol. 5, no. 2, pp. 529-536, 2005.

[26] B. Babenko, M.-H. Yang, and S. Belongie. Visual tracking with online multiple instance learning. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), pp. 983-990, 2009.

[27] H. Grabner and H. Bischof. On-line boosting and vision. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), vol. 1, pp. 260–267, 2006.

[28] A. Saffari, C. Leistner, J. Sanntner, M. Godec, and H. Bischof. On-line random forests. IEEE Int’l Conf. Computer Vision (ICCV), WS on On-Line Learning for Computer Vision, pp. 1393-1400, 2009.

[29] A. Adam, E. Rivlin, and I. Shimshoni. Robust fragments-based tracking using the integral histogram. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), pp. 798–805, 2006.

[30] Godec, M., Sternig, S., Roth, P.M., Bischof, H. Context-driven clustering by multi-class classification in an active learning framework. IEEE Conf. Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 19–24, 2010.

[31] Z. Kalal, K. Mikolajczyk, and J. Matas. Tracking-learning-detection. IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 34, no. 7, pp. 1409–1422, 2012.

[32] T.B. Dinh, N. Vo, and G. Medioni. Context tracker: exploring supporters and distracters in unconstrained environments. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), pp. 1177–1184, 2011.