Tracking customers in crowded retail scenes with Siamese Tracker

Pha A. Nguyen
University of Science
Vietnam National University Ho Chi Minh City
Ho Chi Minh City, Vietnam
nganhpha.hcmus@gmail.com

Son T. Tran
University of Science
Vietnam National University Ho Chi Minh City
Ho Chi Minh City, Vietnam
ttson@fit.hcmus.edu.vn

***Abstract*—Object tracking is a fundamental domain in computer vision, especially multiple human tracking, which is an important task to solve many key problems, including traffic counting, behavior analysis of in-store customers. It is very difficult to handle these problems in heavily crowded scenes because of pedestrian occlusions. Recent research works focus on tracking-by-detection, which breaks tracking problem down into two steps: detect objects then associate them with groups by motion or features. Recent object detectors still do not have the accuracy-speed balance and do not work perfectly in a crowded area. In this paper, we propose a new method for human tracking, which is less depended on the instability of the object detector and inherits the success of Siamese Visual Tracking in recent years. While Siamese-family trackers work on a single object, we extend the problem by proposing a strategy that manages across multiple tracker and combines a human detector as a semi-supervisor for tracker location correction. Additionally, the detector-tracker associating strategy adapts the transformation of human poses and the acquisition of new pedestrians during the tracking process. Our method outperforms the state-of-the-art algorithms on our crowded scenes dataset.**

***Index Terms*—Siamese Network, Multiple Object Tracking, Computer Vision**

I. Introduction

The problem of object detection, localization, and tracking has got significant attention in different research areas. Modern trackers could be roughly divided into two branches. The first one is *tracking-by-detection*, which has become the leading paradigm of multiple object tracking. SORT [1] and DeepSORT [2] are popular methods for online tracking algorithms. The simple idea is to employ a frame-by-frame data association using the Hungarian method with an association metric that measures the deep feature distance and bounding box overlap. However, they need an object detector that works all the time.

The other branch is *template-matching*, which uses a template of the target object and tries to match it to the regions of the later images. Several works in the *template-matching* method focus not only on deeper and wider networks, but also on searching as proposing regions to improve tracking robustness and accuracy on a single object. SiamFC [3] uses template embedding as the correlation filter for the search image to allow real-time performance. SiamRPN [4] uses region proposal networks that have been used by Faster R-CNN [5] to extract proposals from the correlation feature maps. Siamese instance search tracking (SINT) [6] trains a Siamese Network using the margin contrastive loss for the representation of features. By using these extracted features, a learned matching function compares the initial patch of the target image with the later patches that are centered at the previously detected location. The patch with the highest positive score is considered to be matching. GOTURN [7] centers around the previous location and regresses the next position of the bounding box, assuming that the object does not move too far [8].

Fig. 1. Sample of our **NgocHa Retail Store** dataset. All frames are marked by the bounding box representing different ID.

Inspired by SORT [1] and DeepSORT [2], we extend the target of Siamese Tracker into multiple object, especially on human, by adding the matching stage periodically after tracking. This approach can be applied to all common types of variations in appearance from tracking examples.

II. Multiple Pedestrian Tracking with Siamese Tracker

In this section, we present our strategy to track pedestrians in detail.

A. Simple Feature Updating

Since human pose changes all the time, we adopt a simple feature update step to an existing tracker when there is a new detected position. At the jth frame, the detect stage returns the cropped image of the ith object (denoted as z_{ij}). Let $\phi(z_{ij})$ denotes the deep convolutional appearance template from the Siamese Network and $Z_{ij} = \phi(z_{ij})$. In order to adapt object appearance, we compute $\phi(z_{ij})$ and update the new template

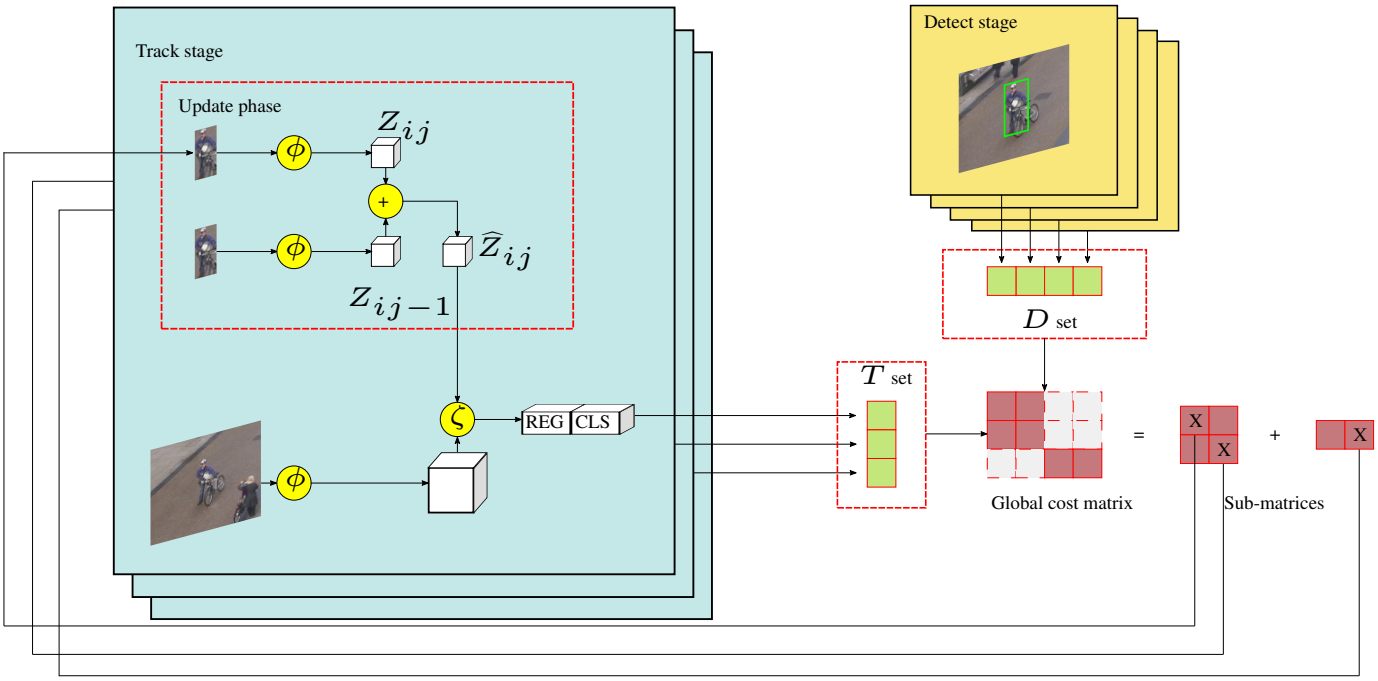

Fig. 2. The detector-tracker associating strategy.

feature with a small coefficient to balance two parts whenever it gets the detector to correct. This action may not always be continuous.

$$\widehat{Z}_{ij} = \begin{cases} Z_{ij} & , j = 0 \\ \gamma \widehat{Z}_{ij-1} + (1-\gamma) Z_{ij} & , j > 0 \end{cases} \tag{1}$$

The update function is a very simple linear function by using previous appearance templates [9]. The update rate γ is assigned to a fixed small value in range $[0, 1]$, following the assumption that the human pose changes quite much. During our experiments, we found that setting γ in the mid-range gives an excellent result.

B. Trackers and detections associating strategy

At any tracking step, the Siamese Network takes x_{ij} as the search field that is samples around the previous position of the target and then predicts the location of the target in the current frame. We employ the Region Proposal Network generation and selection framework that is first introduced in Faster-RCNN [5] and also used in SiamRPN [4]. Let function $\zeta(\widehat{Z}_{ij}, \phi(x_{ij}))$ denotes for the correlation function, which is computed on both two frameworks and returns result in two branches:

$$REG, CLS = \zeta(\widehat{Z}_{ij}, \phi(x_{ij})) \tag{2}$$

Let bounding box regression branch denotes as REG, the classification branch denotes as CLS, and the number of anchor box is k. The REG branch returns the output vector whose shape is $(21, 21, 4, k)$, each point whose shape is $(21, 21, 4)$ represents the distance between bounding boxes generated and a single relative anchor box. Similarly, the CLS branch returns the output vector whose shape is $(21, 21, 2, k)$ and each point in the CLS branch has $(21, 21, 2)$ shape, representing the positive score (denoted as ps) and negative score (denoted as ns) of these bounding boxes.

$$\begin{aligned} REG &= \{(dx_{\alpha\beta\kappa}, dy_{\alpha\beta\kappa}, dw_{\alpha\beta\kappa}, dh_{\alpha\beta\kappa})\} \\ CLS &= \{(ps_{\alpha\beta\kappa}, ns_{\alpha\beta\kappa})\} \end{aligned} \tag{3}$$

where $\alpha, \beta \in [0, 21)$, $\kappa \in [0, k)$.

From the highest positive score index in the CLS branch, we get the next location of the ith object in the REG branch as the same index as the score:

$$\begin{aligned} A, B, K &= argmax(softmax(CLS)[:, :, 0, :]) \\ t_l &= convert\_boundingbox(REG[A, B, :, K]) \\ t_s &= softmax(CLS)[A, B, 0, K] \end{aligned} \tag{4}$$

where t_l is the tracker location and t_s is the tracker score. The 0 index in the CLS branch represents the positive scores ps.

We use the softmax function to normalize each pair (ps, ns) into a vector of two values that follows a probability distribution whose total sums up to 1 and convert the distance with the corresponding anchor to the bounding box of the target. We map a collection of objects whose each object t is tracked as mentioned above in parallel, combine to set T and denote $|T| = n$.

Besides, the object detector detects object bounding boxes (denotes as set D and $|D| = m$) in the following format: $D = \{(x_\alpha, y_\alpha, w_\alpha, h_\alpha) | \alpha \in [0, m)\}$. To match each pair from the tracker predictions set T and detections set D, we compute an intersection over union association cost:

$$c(t_l, d) = 1 - \frac{t_l \cap d}{t_l \cup d} \tag{5}$$

TABLE I
PERFORMANCE COMPARISONS ON TWO DATASETS

Methods	NgocHa Retail Store									
	$frame\_interval = 1$					$frame\_interval = 10$				
	IDF1	IDP	IDR	MOTA	MOTP	IDF1	IDP	IDR	MOTA	MOTP
GOTURN	22.0	22.0	22.0	-4.5	0.349	-	-	-	-	-
SiamRPN	25.9	25.9	25.9	-25.1	0.330	-	-	-	-	-
DeepSORT	64.0	62.6	65.5	94.9	0.033	53.1	52.0	54.2	84.5	**0.084**
Modified SiamRPN (Ours)	**69.8**	**69.9**	**69.8**	**99.8**	**0.000**	**56.1**	**56.5**	**55.7**	**95.6**	0.117
	TownCentre									
Methods	$frame\_interval = 1$					$frame\_interval = 10$				
	IDF1	IDP	IDR	MOTA	MOTP	IDF1	IDP	IDR	MOTA	MOTP
GOTURN	28.1	28.2	27.9	-2.1	0.343	-	-	-	-	-
SiamRPN	39.1	39.1	39.1	-12.5	0.376	-	-	-	-	-
DeepSORT	80.5	76.5	**85.0**	87.7	0.019	70.4	69.3	71.6	76.6	**0.104**
Modified SiamRPN (Ours)	**85.4**	**85.8**	**85.0**	**98.6**	**0.000**	**79.9**	**81.8**	**78.2**	**89.4**	0.167

where $t \in T$ and $d \in D$.

The global cost matrix can be considered to formulate as:

$$C(T,D)_{n\times m} = \begin{pmatrix} c(t_{l1},d_1) & c(t_{l1},d_2) & \cdots & c(t_{l1},d_m) \\ c(t_{l2},d_1) & c(t_{l2},d_2) & \cdots & c(t_{l2},d_m) \\ \vdots & \vdots & \ddots & \vdots \\ c(t_{ln},d_1) & c(t_{ln},d_2) & \cdots & c(t_{ln},d_m) \end{pmatrix} = (c(t_{l\alpha},d_\beta)) \in \mathbb{R}^{n\times m} \tag{6}$$

Additionally, the positive score is considered as an important factor to prioritize predictions. We select the subset of trackers $\{t \in T | t_s = score\}$ that have the same score and unmatched subset of detections $\widehat{D}$ that have not been associated with any trackers in the previous matches. Selected $score$ is descending sorted from $max(\{t_s | t \in T\})$ to $min(\{t_s | t \in T\})$. Then the sub cost matrices between each pair of two sets at considered score can be formulated as:

$$\begin{gathered} C(\{t \in T | t_s = score\}, \widehat{D}) \\ \textbf{for } score \in descending\_sort(\{t_s | t \in T\}) \end{gathered} \tag{7}$$

In the implementation, we linearly rescaled values $t_{l\alpha}$ into a new arbitrary range 0 to 100 so that values can be observed easily. After computing sub cost matrices, we also mark certain impossible pairings. By separating global cost matching problem into several sub cost matches, we reduce the computation cost of linear assignment problem from $O(|T|^3)$ to $O(a^3) + O(b^3) + \ldots$ where $a + b + \ldots = n$. The values of $a, b, \ldots$ depend on the degree of occlusion of data scenes.

A linear sum assignment problem solver for dense matrices is applied to get suitable pairs on each matrix. After getting matched pairs, we update detections to trackers as described in (1). On the other hand, trackers that are not matched with any detections are considered to be occluded and still predict shortly until the correction is made or after a long time. Also, the unmatched observations will be initialized as new trackers.

The strategy process flow is depicted in Fig. 2. The Track stage leverages the Siamese Tracker's work and is along with the Update phase to adapt human pose change by a simple update function. $\widehat{Z}_{ij} = Z_{ij} + Z_{ij-1}$ is a simple representation of (1). In implementation, we compute the accumulated $\widehat{Z}_{ij-1}$ instead of Z_{ij-1}. Each object in T set associates with each object in D set by (5). The global cost matrix, which may contain impossible match pairs, is split into many sub-matrices and assignment sum cost is optimized on these matrices. Compatible pairs, which are marked by **X** letter in Fig. 2, will be used to update the status of the objects.

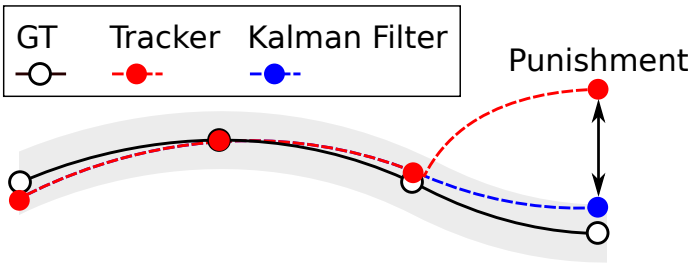

Fig. 3. A punishment occurs when the tracker suddenly changes direction.

C. Kalman Filter Penalty

We break the assignment problem in a global cost matrix into a series of sub-matrices as described in II-B. Additionally, with some uncertainty or inaccuracy in prediction, the rescaled $score$ is multiplied by a penalty to decrease the priority of suddenly direction changed trackers, also expand the score range. We use a simple Kalman Filter with the assumption that those objects are moving at the constant velocity motion. We take the bounding box center position as direct observation of the object state. The distance between the Kalman Filter prediction $p = (x, y)$ and Siamese Tracker prediction center $center(t_l) = (x, y)$ over the frame size (w, h), which map to the interval $[0, 1]$, is computed as the punishment weight:

$$\widehat{t_s} = t_s \times (1 - norm(\frac{center(t_l) - p}{image\_size})) \tag{8}$$

The punishment is illustrated in Fig. 3. For simplicity, we plot ground truth trajectory with solid curve and the predictions with dashed ones. The gray area indicates the matching threshold [10]. The red curve represents the Siamese Tracker, and the blue one is the Kalman Filter output.

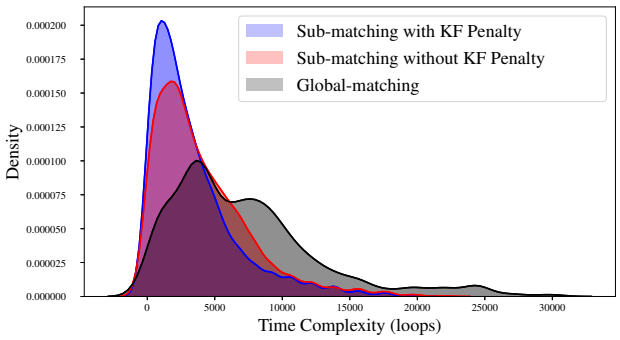

Fig. 4. Histogram time complexity distribution between three types of matching. Best viewed on color display.

D. Training phase

Since all pretrained Siamese convolutional appearance networks (denoted as ϕ) are trained on a large number of object types, we retrain Siamese convolutional network on the **Market-1501** [11] dataset with AlexNet [12]. Each positive pair is generated by 2-permutations of n intra-trackers, where n is the length of a tracker. From 751 trackers, 593876 positive pairs are generated with a large number of human poses, cloth colors. Negative samples are also picked by randomly choosing two people. Data augmentation methods are considered applicable, we apply horizontal flip, grayscale, blur, shift scale, and color distortions on both template images and search images.

We use the loss function in Faster R-CNN [5] to train the network on both the classification and the regression branch. The loss for classification is the cross-entropy loss and smooth L1 loss for regression. Finally, we optimize the two losses combination:

$$L = L_{cls} + \lambda L_{reg} \tag{9}$$

III. Experiments

A. State-of-the-art algorithms comparison

We evaluate our proposed method on two datasets. **NgocHa Retail Store**, which is recorded at retail shopping sites and focuses on heavily occluded humans and unpredictable movement trajectories. Sample is shown as Fig. 1. This dataset contains 38015 human bounding boxes within 7580 images. And **TownCentre** [13] dataset is recorded in a busy town center street. Up to 230 pedestrians were tracked simultaneously in 4500 frames, contains 71460 bounding boxes. Also, we only focus on our matching framework by using ground truth bounding boxes as input. We further compare our method results with the original SiamRPN [4], DeepSORT [2] and GOTURN [7] tracking methods on the MOT metrics [10] as shown in Table I.

Seeing detector as a semi-advisor, we evaluate the tracker predictions within a $frame\_interval$ then get the matching

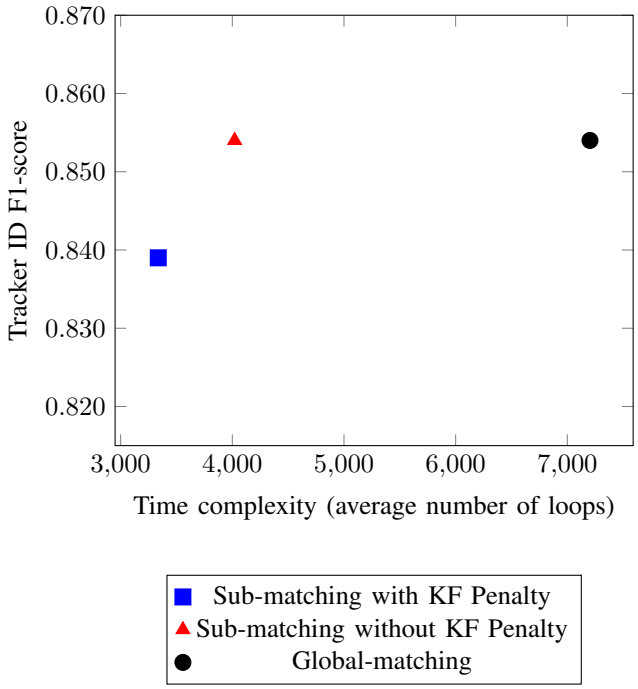

Fig. 5. Complexity and F1-score compares between three methods.

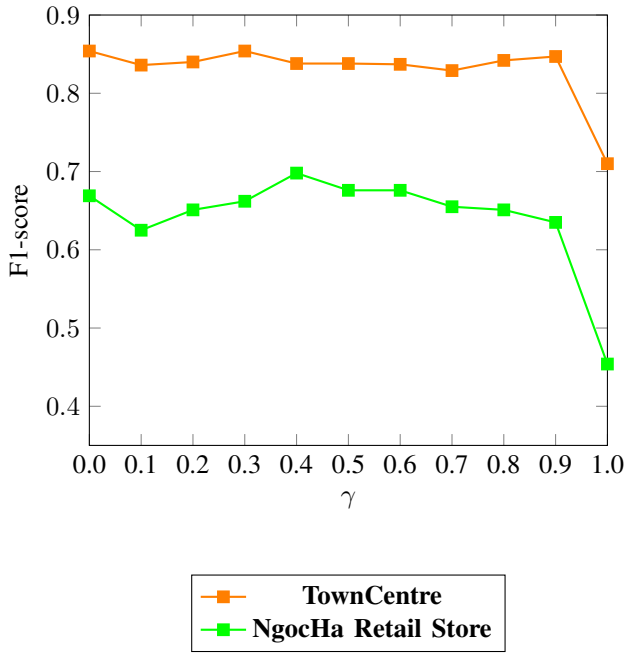

Fig. 6. The change of F1-score is affected by γ. Best viewed on color display.

Fig. 7. Qualitative comparison result on two methods. Red bounding boxes are DeepSORT's visualization and green bounding boxes are the visualization of our method. Our method performs tracking better than DeepSORT after a complete occlusion. The 9th ID changes to the 11th ID, compares with the 8th ID, whose prediction is more stable. Best viewed on color display.

strategy as the correction, so it only occurs on detection-based methods. Since the original idea of GOTURN [7] and SiamRPN [4] only works on a single object, we have to add some modifications to make it work on multiple object. By adding the initialization phase based on the first time new object appears from the ground truth and letting it track till the end without any correction, we see it change ID easily in crowded environments and it is vulnerable at occlusion scenes.

Thanks to the deep convolutional feature of Siamese Network, our method can perform not only a better quantitative result on ID precision, ID recall, ID F1-score, but also better qualitative comparison[1], as demonstrated in Fig. 7.

B. Time complexity

We benchmark the assignment time complexity of our cascade matching strategies on the **TownCentre** [13] dataset as described in II-B. In Fig. 4 and Fig. 5, the blue area and the blue square mark represent the matching strategy which applies with (7) and Kalman Filter penalty (8), the red area and red triangle mark represent the strategy applies with (7) but does not use penalty (8), and the black area and black dot use the naivest approach, which simply assigns on the global cost matrix (6) without penalty (8). The time complexity, which is counted by the number of loops, significantly decreases from the black to the red to the blue. The number of iterations of the highest density point in the blue area is less than the number of two others. However, this results to a trade-off for a slight drop of F1-score as in Fig. 5. In this work, we only benchmark on small dataset. On a large and extreme occlusion dataset, the gap could be wider.

[1]Full tracking visualization by DeepSORT [2] is uploaded at https://youtu.be/Jp-mjmrO0uU, and our modified Siamese Tracker is at https://youtu.be/OeN46gICi2w.

C. The update rate observation

The update rate γ in (1) is crucial to learn a good appearance for human tracking. In Fig. 6, we observe the change of F1-score by the increase of γ in the range $[0, 1]$. By choosing a value which makes two parts balance and focusing on new features, we achieve the highest F1-score on **NgocHa Retail Store** at $\gamma = 0.4$. And with the dataset whose human moving trajectories are stable and linear as in the **TownCentre** [13] dataset, the γ coefficient does not affect to excess, the best result is also obtained when γ is in the mid-range. Oppositely, F1-score decreases dramatically when γ reaches 1. It means that the template feature is only initialized for the first time, and it never gets updated later.

IV. Conclusion and Future Work

With a good method to track in-store customers, these questions are easier to approach: What time is the store most crowded? How much time do customers spend? And how will these affect sales? We focus in this work not only on the real in-store data but also on the efficient method to handle the Siamese Tracker and the object detector in a crowded environment. During the process, the feature extraction branch is trained by a custom human database **Market-1501** [11]. Meanwhile, our tracking system shows good performance in tracking precision and recall.

We remark that the object deep feature might be an important factor to be considered when matching tracker prediction with detection besides the overlapping area. A new branch in the network architecture with a fuse loss function training may be used to perform appearance representation besides classification and regression branch.

In this paper, we trained our backbone on the **Market-1501** [11] dataset, which contains cropped pedestrian images. By collecting more dataset with full-frame availability, we are hopeful that the regression branch can perform better results.

Since several deeper and wider networks are created and boost not only the neural network depth but also the performance of all the computer vision tasks, we also regard them as major improvements in tracking problem. But with the limitation of resources, we only present AlexNet [12] performance in this work. In the future, we will attempt to take full advantage of the neural network.

ACKNOWLEDGMENT

We would like to thank Tuan Hue Thi for his great support and discussion.

REFERENCES

[1] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple Online and Realtime Tracking," *arXiv e-prints*, p. arXiv:1602.00763, Feb 2016.

[2] N. Wojke, A. Bewley, and D. Paulus, "Simple Online and Realtime Tracking with a Deep Association Metric," *arXiv e-prints*, p. arXiv:1703.07402, Mar 2017.

[3] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. S. Torr, "Fully-Convolutional Siamese Networks for Object Tracking," *arXiv e-prints*, p. arXiv:1606.09549, Jun 2016.

[4] B. Q. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu, "High performance visual tracking with siamese region proposal network," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8971–8980, 2018.

[5] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," 2015.

[6] R. Tao, E. Gavves, and A. W. M. Smeulders, "Siamese Instance Search for Tracking," *arXiv e-prints*, p. arXiv:1605.05863, May 2016.

[7] D. Held, S. Thrun, and S. Savarese, "Learning to Track at 100 FPS with Deep Regression Networks," *arXiv e-prints*, p. arXiv:1604.01802, Apr 2016.

[8] A. Sauer, E. Aljalbout, and S. Haddadin, "Tracking holistic object representations," 2019.

[9] L. Zhang, A. Gonzalez-Garcia, J. van de Weijer, M. Danelljan, and F. S. Khan, "Learning the model update for siamese trackers," 2019.

[10] A. Milan, L. Leal-Taixe, I. Reid, S. Roth, and K. Schindler, "MOT16: A Benchmark for Multi-Object Tracking," *arXiv e-prints*, p. arXiv:1603.00831, Mar 2016.

[11] L. Zheng, L. Shen, L. Tian, S. Wang, J. Wang, and Q. Tian, "Scalable person re-identification: A benchmark," *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1116–1124, 2015.

[12] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," *Neural Information Processing Systems*, vol. 25, 01 2012.

[13] B. Benfold and I. Reid, "Coarse gaze estimation," https://www.robots.ox.ac.uk/ActiveVision/Research/Projects/2009bbenfold_headpose/project.html#datasets, 2009.