Online Multi-Object Tracking for Team Sports Analysis

Xianze Wu, Yuchen Fang, Mingcheng Chen, Jiayu Miao, Minghuan Liu
Shanghai Jiaotong University
{xzwu, arthur_fyc, cmc_iris, mg2015started, minghuanliu}@apex.sjtu.edu.cn

Figure 1: Tracking results of our implemented approach on NCAA basketball game video set. Each line shows a tracking flow, where every picture is sampled 0.2 second in average.

Abstract

Multi-Object Tracking (MOT), also known as visual tracking, has become an active research area of the computer vision community for decades. Recently, with the development of deep learning, an increasing amount of works are able to deal with more and more complex video sequences, which support complicate applications. An intriguing one is to apply MOT to help team sports analysis, such as soccer, basketball, football, hockey and so on. In this paper, we focus on tracking each player in a basketball game with the videos. To that end, we implement a real-time online MOT method under track-by-detect architecture, where YOLOv3 and SORT are served as the detector and the tracker respectively with clustering to separate each team. In our experiments, we apply our tracking algorithm on National Collegiate Athletic Association (NCAA) basketball game video dataset and present good performance on both detection and tracking. For illustration, we show a short demo video[1] *in anonymous on one of the 2011 NCAA basketball game.*

[1]https://www.dropbox.com/s/imp93f405k7u0bl/output.mp4?dl=0

1. Introduction

Multi-Object Tracking (MOT), which requires detecting and tracking multiple objects across a scene, has been a long standing challenge for Computer Vision and Machine Learning researchers with a wide range of applications. In this paper, we pay attention on sport analysis, a specific application area of MOT, which has high demands for tracking all the players on the ground to allow for several downstream tasks, e.g., drawing players with their coordinates on a 2D panel by projection each player in the video instead of by hands to help coaches to make useful tactics.

MOT is required to solve within a video, which consist lots of continuous pictures. Thus, typical MOT procedure contains the sub-task of detection, which aims to identify each object in each picture, then utilize the information provided by the detector to track each object in the video. Such a tracking framework is called track-by-detect architecture. MOT methods can be divided into online and offline tracking based on whether the algorithms use future information. Offline tracking can use the future position of each object, which has more accuracy, but can hardly be applied to tasks with highly real-time demands. On the contrary, online tracking methods concentrate on using previous in-

formation to track each object, which are convenient to be deployed to real-time tasks, like live stream.

Compared with MOT in other scenes, multiple player tracking in sports video is much more difficult due to the following reasons: (1) players in the same team are always visually similar; (2) sports players often interact with others in complex ways and (3) the occlusions are much more frequent and severe. These problems cause more difficulty in the tracking system, which requires not only reliable observations but also a sophisticated tracking strategy to make the system robust.

In this paper, we implement a real-time online MOT method to solve multiple player tracking problem. Specifically, we choose a practical real-time MOT architecture of track-by-detect, which is equipped with a YOLOv3[23], a competitive approach of object detection, and SORT[6], an efficient approach of real-time tracking with detection results, with K-means clustering to separate each team. In our experiments, we apply our implemented method on National Collegiate Athletic Association (NCAA) basketball game video dataset to tracking all the players on the ground and present good performance on both detection tracking metrics. The tracking results on one of the 2011 NCAA basketball game are shown in Fig. 1 which illustrates that our method effectively tracking all the players on the ground, which contains great potential to downstream application tasks.

2. Related Work

2.1. Single Object Tracking

Single Object Tracking (SOT)[9, 8] is a special case of multi-object tracking when the number of tracking object with is exactly one. This simplification makes the challenge much easier to solve. A simple feed-forward regression network learning a generic relationship between object motion and appearance was proposed in [11]. Moreover, siamese trackers [15, 16, 27, 31] have recently achieved state-of-the-art performance. The original siamese tracker was proposed in [27], which is based on a correlation filter learner. It is able to extract deep features that are tightly coupled with the correlation filter. An extension of the siamese tracker using region proposal networks (RPN) was introduced in [16]. It employs the siamese subnetwork for feature extraction and the RPN to perform classification and regression. Most of recent SOT trackers can be trained end-to-end, but their performance significantly drops when applied directly to MOT.

2.2. Multi-Object Tracking

Multi-object tracking often follows the track-by-detect paradigm. Unlike single object tracking, the goal of a MOT tracker is to solve the data association problem. Standard benchmarks are proposed in [14] for pedestrians tracking.

For offline MOT tasks, [25] formulates the multi-person tracking by a multi-cut problem and use a pair-wise feature which is robust to occlusions. Person re-identification methods have been combined into a tracking framework in [26]. Quadruplet convolutional neural networks are used in [24]. They perform metric learning for target appearances together with the temporal adjacencies which is then used for data association. In addition, [13] proposes a bilinear LSTM to learn the long-term appearance models, where the memory and the input have a linear relationship.

For online MOT tasks, [2, 4] formulate the problem in a probabilistic framework, then use a variational expectation maximization algorithm to find the track solution. Moreover, [7] proposes an aggregated local flow descriptor which encodes the relative motion pattern and then performs tracking. Another solution is proposed in [28], which uses Markov decision processes and reinforcement learning for the best data association. Alternatively, different information are fused together with a convolutional network. Besides, [30] proposes to use dual matching attention networks with both spatial and temporal attention mechanisms. The estimation of the detection confidence is realized in [3], where the detection with different scores are then clustered in to different classes and they are processed separately. [17] proposes a recurrent neural network (RNN) based method for both motion dynamics and detection-to-track data association and further explored it to NP-hard problems [18].

2.3. Multi-Player Tracking

There are also particular works that has been proposed to deal with the multi-player detection and tracking problem in sport games. For example, [19] uses mixture particle filters and Adaboost to detect and track hockey players using video sequences. Similarly, [29] proposes a track-by-detect framework for broadcast sports video that uses support vector machine (SVM) and particle filters[1]. These approaches are always limited in hand-crafted features, which most of the time, which makes them hard to scale to other scenes, even similar sports. With the development of deep learning, most of those common MOT approaches shown above can be also applied in this specific task, only if there is enough data and labels.

3. Object Detection

The architecture of our detection algorithm is based on YOLOv3 [23]. YOLOv3 is an improved version of the YOLO (You Only Look Once) and YOLOv2/YOLO9000 [21, 22]. Based on previous versions of models, YOLOv3 adjust the network structure, introduce multi-scale features for object detection and use logistic instead of softmax for object classification. Therefore, on the premise of main-

taining the speed advantage, the prediction accuracy is improved, especially the recognition ability of small objects is strengthened. This model has been proved to produce state-of-the-art results in terms of both accuracy and speed.

The main idea behind YOLO is that the model only looks once to the image. It frames object detection as a regression problem to spatially separated bounding boxes and associated class probabilities firstly. Then a neural network predicts bounding boxes and class probabilities directly from full images in one evaluation. Since the whole detection pipeline is a single network, it can be optimized end-to-end directly on performance. More specifically, the process of objective detection is listed into the following parts.

3.1. Bounding Box Prediction

Object detection, i.e. object recognition and location, can be regarded as two tasks: finding the region of an object in an image, and then identifying the specific object in the region.

To deal with the task of finding the region of interest (ROI), our model using a convolutional neural network to map ROI to feature. An input image is divided into coarse 13×13, medium 26×26 and fine 52×52 grids respectively, which enable the network predict three different sizes of bounding boxes. If the center of an object falls into a grid cell, that grid cell is responsible for detecting that object. The network predicts 4 encoded coordinates for each bounding box, t_x, t_y, t_w, t_h. Once given the cell offset (c_x, c_y) from the top left corner of the image and the bounding box prior with width p_w and height p_h, we can decode the output 4 dimension coordinates vector t_x, t_y, t_w, t_h into final prediction, which containing center point (b_x, b_y), width b_w and height b_h of prediction bonding box by following equations,

$$b_x = \sigma(t_x) + c_x \tag{1}$$
$$b_y = \sigma(t_y) + c_y \tag{2}$$
$$b_w = p_w e^{t_w} \tag{3}$$
$$b_h = p_h e^{t_h} \tag{4}$$

where $\sigma(t_x), \sigma(t_y)$ are offset from the top left corner of the grid to the center point of the rectangular bounding box, and σ is the activation function. In our model implementation, we take $Sigmoid$ function as our activation function here.

We use sum of squared error loss to train this bounding box prediction network, and minimize the error between output prediction t and ground truth encoded coordinate t'. The ground truth encoded coordinate can be easily calculated with b'_x, b'_y, b'_w, b'_h known by inverting the equations above.

After deriving prediction result of bounding box, the proposed model further calculates confidence score c for each bounding box by Eq. (5).

$$C = \text{Pr(Object)} * \text{IOU}^{truth}_{pred} \tag{5}$$

where Pr(Object) is the probability of objects in bounding box, $\text{IOU}^{truth}_{pred}$ is IOU (Intersection over Union) between prediction bounding box and ground truth bounding box. If $C = 1$, the bounding box prior overlaps a ground truth object by more than any other bounding box prior. If the bounding box prior is not the best but does overlap a ground truth object by more than threshold $e = 0.5$, we ignore the prediction.

3.2. Class Prediction

To determine which class the object in bounding box is, the task of identifying the specific object in the region is raised after bounding box prediction.

We formulate the problem as a multilabel classification problem, and use independent logistic classfiers between each pair of classes. With the object category probability $p(c)$, we calculate cross entropy loss as the loss of class prediction task, and simple combining the loss to the sum of squared error loss in bounding box prediction task.

Let λ_{coord} denotes a hyperparameter that increases the loss from bounding box coordinate predictions, λ_{noobj} denotes a hyperparameter that decreases the loss from confidence predicitons for boxes that don't contain objects, $\mathbf{1}_i^{obj}$ denotes if an object appears in cell i and $\mathbf{1}_{ij}^{obj}$ denotes that the jth bounding box predictor in cell i is responsible for that prediction. The overall loss function of detection model can be written as Eq. (6)

$$\begin{aligned}
\mathcal{L}oss =& \lambda_{coord} \sum_{i=0}^{N\times N} \sum_{j=0}^{K} \mathbf{1}_{ij}^{obj} \left[(t_x - t'_x)^2 + (t_y - t'_y)^2\right] \\
&+ \lambda_{coord} \sum_{i=0}^{N\times N} \sum_{j=0}^{K} \mathbf{1}_{ij}^{obj} \left[(t_w - t'_w)^2 + (t_h - t'_h)^2\right] \\
&- \sum_{i=0}^{N\times N} \sum_{j=0}^{K} \mathbf{1}_{ij}^{obj} \left[c'_i \log(c_i) + (1 - c'_i) \log(1 - c_i)\right] \\
&- \lambda_{noobj} \sum_{i=0}^{N\times N} \sum_{j=0}^{K} \mathbf{1}_{ij}^{noobj} [c'_i \log(c_i) + \\
&(1 - c'_i) \log(1 - c_i)] \\
&- \sum_{i=0}^{N\times N} \mathbf{1}_{i}^{obj} \sum_{c\in classes} [p'_i(c) \log(p_i(c)) + \\
&(1 - p'_i(c)) \log(1 - p_i(c))]
\end{aligned} \tag{6}$$

According to the Eq. (6) loss function, we simultaneously penalizes incorrect object detection as well as considers what the best possible classification would be. In

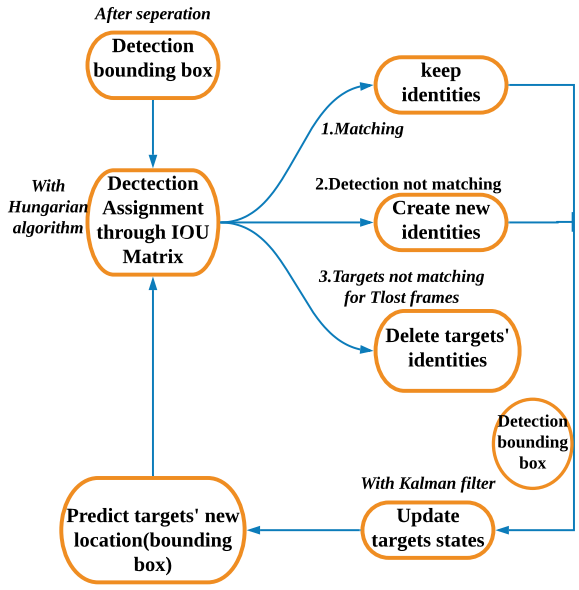

Figure 2: The tracking diagram

conclusion, our end-to-end object detection model simultaneously solve the object detection and object classification tasks.

4. Multi-Object Tracking

Our tracking algorithm is a implementation of tracking-by-detection framework where objects are detected first in each frame and then tracked through the bounding boxes. Our work focus on online tracking where only detection from the previous and the current frame are presented to the tracker, which is simple but effective.

Following some ideas in SORT [6], Our tracking method is mainly composed of three parts, which includes propagating object states into future frames, associating current detection with existing objects, and managing the lifespan of tracked objects. The complete tracking diagram is shown in Fig. 2. Additionally, we use clustering method to divide the players into two teams first and then conduct tracking procedure separately for each team.

4.1. Object Estimation Model

The object estimation model, which can be seen as the representation and motion model, is used to propagate a target's identity into the next frame. With a linear constant velocity model which is independent of other objects and camera motion, we approximate the inter-frame displacements.The state of each target is modelled as:

$$\mathbf{x} = [u, v, s, r, \dot{u}, \dot{v}, \dot{s}]^T \tag{7}$$

where u and v represent the horizontal and vertical pixel location of the centre of the target, while the scale s and r represent the scale (area) and the aspect ratio of the target's bounding box respectively. The aspect ratio can be considered to be constant.

When a detection is associated to a target, the detected bounding box is used to update the target state where the velocity components are solved optimally via a Kalman filter framework [12]. If no detection is associated to the target, its state is simply predicted without correction using the linear velocity model.

4.2. Object Association between frames

To assign detection to existing targets we have constructed, we estimate each target's bounding box geometry by predicting its new location in the current frame. Then the assignment cost matrix is computed according to the intersection-over-union (IOU) distance between each detection and all predicted bounding boxes from the existing targets. If the detection to target overlap is less than IOU_{min}, a minimum IOU, we will not assign the detection to existing targets.

By using the IOU distance of the bounding boxes for assigning, it implicitly handles short term occlusion caused by passing targets. Specifically, when a target is covered by an occluding object, only the occluder is detected, since the IOU distance appropriately favours detection with similar scale. Therefore both the occluder and target is allowed to be corrected with the detection while the covered target is unaffected as no assignment is made.

4.3. Creation and Deletion of Track Identities

Unique identities need to be created when objects enter the image. When a detection is found to have an overlap less than IOU_{min}, it is regarded as an untracked object and an object identity is created for it. We initialise the tracker using the geometry of the bounding box with the velocity set to zero. Also the covariance of the velocity component is initialised with large values in that the velocity is unobserved at this point, which reflects this uncertainty. Additionally, In order to prevent tracking of false positives, the new tracker then undergoes a probationary period where the target needs to be associated with detection to accumulate enough evidence.

When objects leave the image, its origin identity is required to be deleted in time. If tracks are not detected for T_{Lost} frames, it will be terminated. This prevents an unbounded growth in the number of trackers and localisation errors caused by predictions over long duration without corrections from the detector.

4.4. Separation of Teams

When applied tracking method to our basketball scenario, separating the players into their belong teams and then conducting tracking is thought to be able to improve our tracking performance. Since this alleviates the noise when too many player overlap each other.

We first choose the center part in each frame, which is $1/3$ scale of the original detection bounding box. We hope that most of the selected areas are jersey, and jerseys' color can help us divide players into two teams. Then, we calculate the color distribution histogram for selected area. And then k-means clustering method is applied to cluster the distribution vectors into two different clusters(teams). Finally we conduct tracking methods for different teams separately.

5. Experiments

5.1. Dataset

We use the NCAA basketball dataset [2] [20]. The dataset is collected from historical NCAA basketball match live records. 523 video clips of those records are selected and annotated resulted in 9787 annotated frames. We split the dataset into 472 clips containing 8808 frames for training, and 51 clips containing 979 frames for testing. Two frames in NCAA dateset are shown in Fig. 3.

Figure 3: Two frames of the NCAA basketball dataset, the annotated bounding boxes are not showed here.

5.2. Training

We initialize the model with pretrained weight [3] expect for the last three layers, whose size are related to the number of categories. Since we only focus on one category, the weights of the last three layers can't not be initialized by pretrained weight due to the size mismatch.

We train the model for two stages. In the first stage, we only train the weights of the last three layers when remaining parts of the model are fixed. For the second stage, we fine-tune the whole model.

We scale all images in the train set into (608×608). To avoid image distortion caused by stretching, the aspect ratio is preserved after scaling and the empty part is filled with a fixed pixel. We set the batch size as 6. The first stage of training lasts for 20 epochs and the second stage lasts for 10 epochs. We use Adam optimizer with $\beta 1$ = 0.9 and $\beta 2$ = 0.999. The learning rate is increased from 0 to 1e-4 over the first 2 epoch. Subsequently, we decrease the learning rate linearly until 1e-6 over the remaining training process.

[2] http://basketballattention.appspot.com/

[3] https://pjreddie.com/darknet/yolo/

5.3. Experimental Results

Detection For evaluation, we compute the precision-recall curves and the mAP (mean average precision). In addition to the standard PASCAL criteria[10] which requires IOU > 0.5, we also apply some other criteria. Note that since we only focus on a category (i.e. person), the mAP value is equal to the AP (average precision) value of the 'person' category, which are shown in Tab. 1. The Precision-Recall curves are shown in Fig. 4. The results show that as the threshold of IOU increase, the performance is getting down, and the threshold of 0.3 provides a better choice.

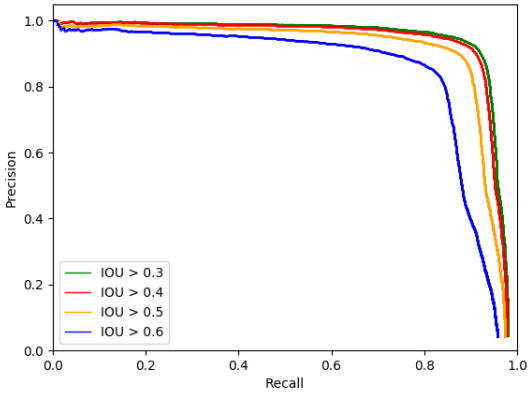

Figure 4: Precision-recall curve when using different criteria

Table 1: mAP when applying different criteria

Criteria	IOU>0.3	IOU>0.4	IOU>0.5	IOU>0.6
mAP	94.16	93.33	90.54	84.06

Table 2: Multi Object Tracking Results

Model	MOTA	MOTP	IDP
Cluster	55.3%	0.422	94.4%
w.o. Cluster	70.1%	0.441	92.8%

Tracking We follow [5] and compute Multi Object Tracking Accuracy (MOTA) and Precision (MOTP) and ID Pre-

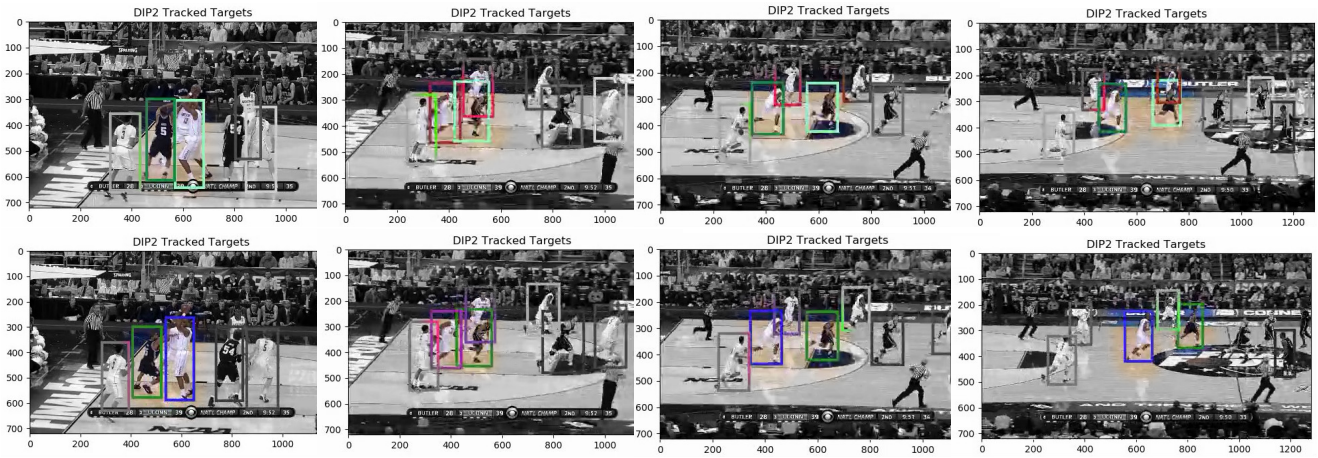

Figure 5: Tracking results when separation of teams are not used (in the upper row) or used (in the lower row). Each color corresponds to a tracking identity. Once the mapping relation is built, it will not be changed during the tracking process. In each row, frames are ranked in time order from left to right.

cision (IDP) as our metrics for tracking. The results are shown in Tab. 2. It is surprisingly that for MOT metrics, the results after clustering deteriorate. We think that this mainly comes from the accumulated error of detection and clustering. Since the clustering principle is simple and naive, those pixels that we choose to compute the color histogram are unrecognizable if there is serious coincidence. Section 5.4 further discuss relevant details of such problems.

5.4. Case study

As we have mentioned in the subsection 4.4, we hope the separation of teams help us relieve the issue that tracking identities of two players are swapped when the two players exchange their relative position. So we study some cases to verify whether our expections are meeted, as shown in Fig. 5.

Consider the two players (marked in color) who belongs to different teams. When separation of teams are not conducted, the tracking identities of the two players are exchanged after the player wearing black jersey run by the player wearing white jersey (see the first and the third image in the first row). However, this case doesn't appear after applying separation of teams (see the first and the third image in the second row), which show the effectiveness of the proposed separation of teams method.

It is worth mentioning that although to us the tracking result after applying separation of teams is better, it would cause one more mismatch case comparing to the one not using separation of teams. This might explain why MOT metrics of the later case is worse.

6. Conclusion and Future Work

In this paper, we implement a real-time online MOT method under track-by-detect architecture, particularly, we use YOLOv3 as the detector and SORT as the tracker to improve our performance. Experiments are mainly conducted on NCAA basketball game video dataset, and we show good results on both detection and tracking.

The weakness comes from that our online tracking algorithm suffers from severe player coincidence, since the tracking performance are highly influenced by the detection results. This can be solved through offline methods which utilize the information both before and after the particular tracking frame.

In the future, we plan to apply more efficient methods on both detection and tracking component, and utilize the MOT architecture to achieve practical downstream applications like projecting the players' position into 2D coordinates on a panel with geometric prior.

References

[1] M Sanjeev Arulampalam, Simon Maskell, Neil Gordon, and Tim Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on signal processing*, 50(2):174–188, 2002. 4322

[2] Sileye Ba, Xavier Alameda-Pineda, Alessio Xompero, and Radu Horaud. An on-line variational bayesian model for multi-person tracking from cluttered scenes. *Computer Vision and Image Understanding*, 153:64–76, 2016. 4322

[3] Seung-Hwan Bae and Kuk-Jin Yoon. Confidence-based data association and discriminative deep appearance learning for robust online multi-object tracking. *IEEE transactions on pattern analysis and machine intelligence*, 40(3):595–610, 2017. 4322

[4] Yutong Ban, Sileye Ba, Xavier Alameda-Pineda, and Radu Horaud. Tracking multiple persons based on a variational bayesian model. In *European Conference on Computer Vision*, pages 52–67. Springer, 2016. 4322

[5] Keni Bernardin, Alexander Elbs, and Rainer Stiefelhagen. Multiple object tracking performance metrics and evaluation in a smart room environment. In *Sixth IEEE International Workshop on Visual Surveillance, in conjunction with ECCV*, volume 90, page 91. Citeseer, 2006. 4325

[6] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 3464–3468. IEEE, 2016. 4322, 4324

[7] Wongun Choi. Near-online multi-target tracking with aggregated local flow descriptor. In *Proceedings of the IEEE international conference on computer vision*, pages 3029–3037, 2015. 4322

[8] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. Eco: Efficient convolution operators for tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6638–6646, 2017. 4322

[9] Martin Danelljan, Andreas Robinson, Fahad Shahbaz Khan, and Michael Felsberg. Beyond correlation filters: Learning continuous convolution operators for visual tracking. In *European Conference on Computer Vision*, pages 472–488. Springer, 2016. 4322

[10] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International journal of computer vision*, 111(1):98–136, 2015. 4325

[11] David Held, Sebastian Thrun, and Silvio Savarese. Learning to track at 100 fps with deep regression networks. In *European Conference on Computer Vision*, pages 749–765. Springer, 2016. 4322

[12] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960. 4324

[13] Chanho Kim, Fuxin Li, and James M Rehg. Multi-object tracking with neural gating using bilinear lstm. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 200–215, 2018. 4322

[14] Laura Leal-Taixé, Anton Milan, Ian Reid, Stefan Roth, and Konrad Schindler. Motchallenge 2015: Towards a benchmark for multi-target tracking. *arXiv preprint arXiv:1504.01942*, 2015. 4322

[15] Bo Li, Wei Wu, Qiang Wang, Fangyi Zhang, Junliang Xing, and Junjie Yan. Siamrpn++: Evolution of siamese visual tracking with very deep networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4282–4291, 2019. 4322

[16] Bo Li, Junjie Yan, Wei Wu, Zheng Zhu, and Xiaolin Hu. High performance visual tracking with siamese region proposal network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8971–8980, 2018. 4322

[17] Anton Milan, S Hamid Rezatofighi, Anthony Dick, Ian Reid, and Konrad Schindler. Online multi-target tracking using recurrent neural networks. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017. 4322

[18] Anton Milan, S Hamid Rezatofighi, Ravi Garg, Anthony Dick, and Ian Reid. Data-driven approximations to np-hard problems. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017. 4322

[19] Kenji Okuma, Ali Taleghani, Nando De Freitas, James J Little, and David G Lowe. A boosted particle filter: Multitarget detection and tracking. In *European conference on computer vision*, pages 28–39. Springer, 2004. 4322

[20] Vignesh Ramanathan, Jonathan Huang, Sami Abu-El-Haija, Alexander Gorban, Kevin Murphy, and Li Fei-Fei. Detecting events and key actors in multi-person videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3043–3053, 2016. 4325

[21] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 4322

[22] Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 4322

[23] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. 4322

[24] Jeany Son, Mooyeol Baek, Minsu Cho, and Bohyung Han. Multi-object tracking with quadruplet convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5620–5629, 2017. 4322

[25] Siyu Tang, Bjoern Andres, Mykhaylo Andriluka, and Bernt Schiele. Multi-person tracking by multicut and deep matching. In *European Conference on Computer Vision*, pages 100–111. Springer, 2016. 4322

[26] Siyu Tang, Mykhaylo Andriluka, Bjoern Andres, and Bernt Schiele. Multiple people tracking by lifted multicut and person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3539–3548, 2017. 4322

[27] Jack Valmadre, Luca Bertinetto, João Henriques, Andrea Vedaldi, and Philip HS Torr. End-to-end representation learning for correlation filter based tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2805–2813, 2017. 4322

[28] Yu Xiang, Alexandre Alahi, and Silvio Savarese. Learning to track: Online multi-object tracking by decision making. In *Proceedings of the IEEE international conference on computer vision*, pages 4705–4713, 2015. 4322

[29] Guangyu Zhu, Changsheng Xu, Qingming Huang, and Wen Gao. Automatic multi-player detection and tracking in broadcast sports video using support vector machine and particle filter. In *2006 IEEE International Conference on Multimedia and Expo*, pages 1629–1632. IEEE, 2006. 4322

[30] Ji Zhu, Hua Yang, Nian Liu, Minyoung Kim, Wenjun Zhang, and Ming-Hsuan Yang. Online multi-object tracking with dual matching attention networks. In *Proceedings of the Eu-*

ropean Conference on Computer Vision (ECCV), pages 366–382, 2018. 4322

[31] Zheng Zhu, Qiang Wang, Bo Li, Wei Wu, Junjie Yan, and Weiming Hu. Distractor-aware siamese networks for visual object tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 101–117, 2018. 4322