Real time multiple object tracking using deep features and localization information

Karunasekera, Hasith; Zhang, Handuo; Wang, Han

2019

Karunasekera, H., Zhang, H., & Wang, H. (2019). Real time multiple object tracking using deep features and localization information. Proceedings of 2019 IEEE 15th International Conference on Control and Automation (ICCA), 332-337. doi:10.1109/ICCA.2019.8899498

<https://hdl.handle.net/10356/141797>

<https://doi.org/10.1109/ICCA.2019.8899498>

© 2019 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works. The published version is available at:
<https://doi.org/10.1109/ICCA.2019.8899498>.

Downloaded on 26 Apr 2023 21:43:51 SGT

Real Time Multiple Object Tracking using Deep Features and Localization Information

Hasith Karunasekera, Handuo Zhang and Han Wang¹

Abstract—In this paper we propose a tracking by detection method using a dissimilarity measure calculated based on the location and the appearance information of the object. These dissimilarity values are used in Hungarian Algorithm [1] in the data association step for track identity assignment. We make use of YOLO [2] deep learning based object detector in the detection step from camera image feed. Location measure is calculated using the predicted object location and bounding box, while the appearance measure is from the last feature layer from the detection network. Main focus in this work is to propose a tracking framework that can be used in real time automated vehicle guiding applications, by striking a balance between computational complexity and tracking accuracy. Therefore, we make use of the deep features available from detection framework rather than calculating a new appearance measure during the tracking step. The method proposed is very efficient and enables to achieve speeds up to 500+ frames per second (fps) in KITTI [3] tracking benchmark while achieving state-of-the-art results.

I. INTRODUCTION

Multiple Object Tracking (MOT) is an important sub task for better understanding of the road scenarios for number of applications such as traffic analysis, surveillance, driver assistance systems as well as automated driving platforms. The proposed work tries to address the MOT problem for automated driving applications. MOT problem is solved using (a) offline or (b) online approach [4]. While the offline (also known as the global) approaches process the whole image sequence at once, the online approaches address the tracking problem at each local frame. Non-real time applications favour global approach as they provide the highest amount of information as the whole image sequence is available as prior knowledge. On the contrary, online method is favoured by real time applications. Therefore, online approach is utilized in the proposed tracking framework, to enable it to be used in real time driving applications.

The most prominent approach for online MOT problem is the tracking by detection approach where the tracking framework is provided with multiple object detection results in each frame as a prior. Therefore, all tracking by detection approaches have two main stages, namely (a) detection stage and (b) data association stage. In our work, we already make use of object detection framework to detect vehicles and pedestrians. This motivated us to use the tracking by detection framework to minimize the computational time for tracking by reusing the detection results.

¹The authors are with the School of Electrical & Electronic Engineering, Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798. {karu0009,hzhang032}@e.ntu.edu.sg; hw@ntu.edu.sg

The contributions of this paper are (a) developing appearance measure based on deep features from detection network, (b) simple motion model to predict object localization information and using intersection over union for matching cost and (c) high speed tracking method suitable for real time applications. In the following section we briefly discuss the current research work in vision based multiple object tracking. The proposed method is discussed in the third section, followed by implementation and evaluation details presented in the fourth section.

II. RELATED WORK

Tracking research can broadly be divided in to MOT and single object tracking. While single object tracking [5], [6], [7], [8], [9] research focuses on detection and tracking a single object or image patch specified in the first frame, MOT focuses on tracking multiple objects belonging to single or multiple pre-defined classes. Unlike single object trackers, almost all the MOT methods follows tracking-by-detection framework where they rely on object detector to provide object candidates at each frame.

Byungjae Lee et al. [10] propose an offline (global) MOT using an ensemble detector based on deep learning and motion to detect objects and deep learning based data driven likelihood calculation with change point detection algorithm to assign detections to tracklets and then tracklets to tracks. In [11], Roberto Henschel et al. present a global optimization on detections based on two detectors on human body and head for pedestrian tracking. In [12] authors use logistic regression for feature vector generated using the information from spatio-temporal object localization, deep keypoint matching and deep person re-identification for track and object matching cost while using a graph based minimum cost lifted multicut approach for association.

While offline MOT methods mostly uses some form of global optimization to associate objects or tracklets to tracks using matching costs from all the frames, online MOT methods are more suitable for real time applications. Sarthak Sharma et al. in [13] present an online method where matching cost is calculated based on object shape, pose, 2D and 3D localization information and appearance based on keypoints detected by a deep network. Then Hungarian assignment [1] is used for association step. Two deep network based detectors [14], [15] are used to detect object proposals. [16] uses a Markov Decision Process for each track which update the state (*active*, *track*, *lost* and *inactive*) of the track at each frame using an appearance based template matching based on dense optical flow matching. Reinforcement learning based

pair wise similarity calculation with Hungarian assignment [1] is used when a track is in lost state and to decide a match is found or not. In [17], authors use predictions of object 3D co-ordinates, 3D motion and 2D object size from a Poisson multi-Bernoulli mixture filter to match with the detections from deep network train to detect object localization and depth information. Since it is filter based tracking it can achieve very high speeds. [18] also uses Hungarian assignment [1] for data association for which the dissimilarity measure is calculated using current frame and previous frame cosine distance of the location and Chi-Square similarity of the RGB histogram of the object.

Wongun Choi in [19] also uses an appearance model, motion dynamics and a novel 228 dimensional aggregated local flow descriptor (based on FAST features as interest point and their trajectories measured from optical flow) as an affinity measure, combined in an energy minimization framework to track objects. In both [20], [21] authors use relative structural motion cues between objects to mitigate the complications introduced by camera global motion on 2D object motion in MOT along with object size and appearance similarity. [20] additionally introduces an event aggregation to cater for false positives and false negatives. [22] introduces a novel Extended Kalman Filter that uses both image 2D and world 3D (using stereo matching and ego-motion calculation) information for object position and size to track the objects.

Some deep learning neural network (DNN) based tracking techniques, especially pedestrian tracking, have also been widely researched in recent years. The main task of DNN based MOT method is to optimize the design of distance measurements or similarities. According to the difference of learning features, DNN based MOT can be divided into appearance model [23], motion model and interaction model. For appearance model, some convolutional neural network (CNN) architectures are used by [24], [25] extracting high-level features. For motion model, Amir et al [26] presented an online method that encodes long-term temporal dependencies across multiple cues with LSTM (Long Short-Term) model over a temporal window. Another paper [27] proposed the Recurrent Autoregressive Network to characterize the appearance and motion model over time. Lastly, interaction models consider the interactions and forces between targets, especially in crowded scenes, like social force model [28] forecasting the long-term trajectories of all targets.

In our work we make use of deep feature information from detection DNN when calculating appearance dissimilarity and predicted object bounding box information to calculate localization dissimilarity. Then Hungarian assignment [1] is used in assigning objects to tracks for which the matching cost is calculated by equally weighting above two dissimilarity measures. We show that the proposed method can achieve state of the art performance with very high running speed due to light computational complexity.

III. METHODOLOGY

All tracking by detection frameworks consist of two main stages, (a) object detection and (b) data association. In the

proposed framework, first an object detector is used to detect the vehicles and the pedestrians present in a given frame. This detection will output the number of objects present along with its class, localization information (object bounding box coordinates in the image plane) as well as features describing each object. Then we use this information to calculate a dissimilarity score between objects detected in the current frame and the objects that are been tracked, which is then used in Hungarian algorithm [1] to decide the best tracks to associate the objects detected in the current frame or whether to initiate a new track if no suitable track is found. This process is presented as a flow chart in Fig. 1 and discussed in detail in this section.

A. Object Detection

In the recent past, object classification and detection performance has improved significantly due to the DNN based methods and the availability of large labelled datasets to train these DNNs. These DNN based object detection methods can be categorized as single stage detectors (or *one-stage detector*) and region proposal based detectors (or *two-stage detector* with *multi-stage detector*). In general region proposal based methods [29], [30] are slightly accurate than single stage detectors [2], [31]. However, the later has lower computational cost allowing for faster running time. Therefore, to favour real-time implementation, in this work we make use of a state-of-the-art single stage detector named YOLOv3 object detector [2], [32]. This object detector outputs localized objects (object bounding box coordinates) with their *objectness* score and classified object class (*car class* or *pedestrian class*). We then use the information in the data association stage.

Fig. 1. Flow chart of the proposed tracking framework

B. Data association

Equation (1) is proposed to calculate the dissimilarity cost for objects detected in the current frame between the existing objects being tracked.

$$dis(i, j) = \begin{cases} 1; & \text{if } i_{cls} \neq j_{cls} \\ 0.5 [app(i, j) + loc(i, j)]; & \text{otherwise} \end{cases} \quad (1)$$

where $dis(i, j)$ is the dissimilarity score between i^{th} object detected in the current frame with the j^{th} object in the tracked objects list, $app(i, j)$ is the appearance based dissimilarity score and $loc(i, j)$ is location based dissimilarity score. i_{cls} is the class of the i^{th} object, and if the object classes are not the same for two objects in comparison, dissimilarity score is set to the maximum value of 1.

In order to calculate the appearance dissimilarity, feature values are extracted from the final feature layer of YOLOv3 [32]. A trained DNN is a good feature extractor. Therefore, a well trained detection DNN results in more robust features. Furthermore, with this strategy same feature calculations are shared for tracking and detection enabling more efficient overall system. Some layers from the final feature layer for three adjacent frames are visualized in Fig. 2.

In this work, the feature descriptor for each object is extracted from the final feature layer of the DNN corresponding to the detected object bounding box. Hence, it results in a feature volume, where the depth is a constant for any object as it is the depth of the final feature layer which is defined during detection network design. The width and the height changes depending on the size of the object detected. This feature volume is used to calculate the value for $app(i, j)$ in (1) by finding the distance between the feature volumes of the i^{th} object in current detection with the j^{th} object in existing tracks. Since the feature volume sizes are different, it cannot be directly used to measure the difference. Therefore, the feature volume of each track is re-sized to the size of the feature volume of the object at hand. For re-sizing we make use of two different methods. If the area ($width \times height$) of the current objects feature volume is smaller than that of the track, the tracks feature volume is re-sized using inter area interpolation (which will shrink the volume by averaging over an area of the original values, depending on the size it needs to be shrunk when calculating new values). Otherwise, inter cubic interpolation is used to re-size the track's feature volume, which uses bicubic interpolation over 4x4 neighbourhood. After re-sizing operation, L2 norm is calculated to estimate the difference between feature volumes. L2 distance is calculated at each feature point in each feature layer and summed together to arrive at a single distance measure called feature dissimilarity $fea(i, j)$ for each i^{th} object in current detection with the j^{th} object in existing tracks, which is expressed in (2).

$$fea(i, j) = \sum_{n=1}^D \sqrt{\sum_{x=1}^W \sum_{y=1}^H [i_{f,n}(x, y) - j_{f,n}(x, y)]^2} \quad (2)$$

where D, W and H are the depth, width and height of feature volume of the i^{th} object in current detection. $i_{f,n}$ is the n^{th} feature layer of the i^{th} object in current detection and $j_{f,n}$ is the n^{th} re-sized feature layer of the j^{th} object in existing tracks.

Therefore, each object detected at the current frame, will get a vector indicating the dissimilarity between the feature volumes of the current object and all the tracks. Length of the feature vector F_i is the number of tracks T considered. Each of the feature dissimilarities $fea(i, j)$ are then normalized by dividing from the maximum of each feature vector F_i to arrive at the appearance dissimilarity $app(i, j)$ as below (3). This will ensure that appearance dissimilarity values $app(i, j)$ are always between 0 and 1.

$$app(i, j) = \frac{fea(i, j)}{F_i}; \quad F_i = \max_{1 \leq j \leq T} fea(i, j) \quad (3)$$

Appearance dissimilarity calculation detailed above is derived based on two assumptions, (a) object bounding box calculated by the detector is tightly fitting the object, thus the features inside the bounding box has negligible amount of features representing the background and a good majority of the features represent the object and (b) object size between adjacent frames do not differ significantly from each other, hence more similar feature volumes will give less feature distance. Based on these assumptions, we directly compare the feature volumes. However, if a detected object is highly occluded it may give rise to a new track as feature similarity is low.

Location dissimilarity is calculated based on the object localization information (object bounding box) of each detection from DNN framework compared to the predicted localization information from object track history. Specifically, intersection over union (IoU) between the detection bounding box $bb_{i,det}$ of the i^{th} object in current detection and the predicted bounding box $bb_{j,pred}$ of the j^{th} object in existing tracks is considered. IoU is inversely proportional to the dissimilarity between two bounding boxes and the IoU will directly provide the similarity value normalized to 0 – 1. Therefore, location dissimilarity is defined below as in (4).

Fig. 2. Final feature layer responses for channels 1, 4 and 12

$$loc(i, j) = 1 - \frac{bb_{i,det} \cap bb_{j,pred}}{bb_{i,det} \cup bb_{j,pred}} \quad (4)$$

IoU is measured considering both the locations and sizes of the bounding boxes in comparison, thus is a good measure of the location similarity of bounding boxes.

Localization history of the each tack is maintained across frames in order to predict the bounding box for the next incoming frame. The predictions for the bounding box center, width and height is generated using a weighted average of it's previous five frames information. We use five frames because it is large enough to get good average for prediction and small enough not to drift because of old information. We use weights to give a higher weighting to the information from the most recent track assignments in the history. Predicted bounding box $bb_{j,pred}$ of the j^{th} object in existing tracks is calculated based on (5) below, where $t + 1$ is the incoming frame and $centerX, centerY, width$ and $height$ are bounding box parameters. m is the consecutive number of frames that track is undetected.

$$bb_{j,pred}^{t+1} = \begin{cases} centerX_{bb_{j,det}^t} + \Delta_{centerX}(1 + m); \\ centerY_{bb_{j,det}^t} + \Delta_{centerY}(1 + m); \\ width_{bb_{j,det}^t} + \Delta_{width}(1 + m); \\ height_{bb_{j,det}^t} + \Delta_{height}(1 + m); \end{cases} \quad (5)$$

where $\Delta_k : k \in [centerX, centerY, width, height]$ is defined as in (6).

$$\Delta_k = \frac{\sum_{a=2}^p [(a-1)(k_{h-p+a} - k_{h-p+a-1})]}{\sum_{a=2}^p (a-1)} \quad (6)$$

$$p = \max(5, h) \quad (7)$$

where h is the number of frames the j^{th} track is matched with a detection.

Once both location $loc(i, j)$ and appearance $app(i, j)$ dissimilarity measurements are calculated, they are averaged to arrive at the overall dissimilarity values $dis(i, j)$ as in (1). Hence, a dissimilarity score $dis(i, j)$ for each i^{th} object detected in the current frame against each j^{th} object in the list of tracked objects is calculated. Therefore, this dissimilarity matrix can be used in the Hungarian algorithm [1] to assign each of the object detection in the current frame to the best matching track. However, this association is done only if the dissimilarity between the matched detection and track is below a certain threshold, otherwise the detected object is initiated as a new track.

In the detection and track association stage object information in the list of tracks is updated with the current detection localization co-ordinates and objectness score. Then the list of tracks is updated to invalidate certain tracks which have not being tracked (i.e. matching object not found) for a predefined consecutive number of frames. In the next section we will discuss implementation details and evaluation of the proposed method.

IV. IMPLEMENTATION AND EVALUATION

The proposed tracking framework is implemented using C++ and all the test cases presented in this section is done with a laptop with Intel Core i7-4712HQ CPU with 2.30GHz 8 processors and Nvidia GeForce GTX 850M/PCIe/SSE2 graphics card. Even though there are 8 processors, code is not optimized for parallel processing. Graphics card is used to run the detection DNN while the tracking module on CPU. Laptop has been used to test the performance as it simulate the embedded computing environment more closely.

KITTI [3] vision benchmark suite is a famous publicly available benchmark for autonomous driving based applications. KITTI [3] detection dataset has been used to train YOLOv3 [32] network for two classes. Training is initiated with original weight file provided and trained for 13,300 iterations. Training dataset for KITTI [3] detection consist of a total of 7481 images, which is used for both training and validation following the same division used in [33]. KITTI [3] tracking dataset consists of 21 training sequences with 8,008 images and 29 test sequences with 11,095 images. Since the proposed tracking method does not require any training on tracking data, both KITTI [3] training and test datasets have been used as a test bed in our experiments. The evaluation code by the KITTI [3] benchmark was used, which is based on CLEARMOT [34] and Mostly Tracked (MT) Mostly Lost (ML) [35] metrics. CLEARMOT [34] includes Multiple Object Tracking Accuracy (MOTA), Multiple Object Tracking Precision (MOTP), Multiple Object Detection Accuracy (MODA), Multiple Object Detection Precision (MODP), total Identity Switches (IDS) and average run-time per frame excluding detection time (Time).

Results with the KITTI [3] training and test data are presented in Table 1 and Table 2 respectively while results are visualized in Fig. 3. First, we compare the effectiveness of the proposed location based dissimilarity measure loc , with the standard Kalman filter kf based location dissimilarity, where a Kalman filter is used to predict the bounding box for each track. Results indicate that the proposed method perform slightly better than the Kalman filter based method in terms of accuracy as well as execution time. Particularly the ID switches are lower in the proposed method.

TABLE I
RESULTS IN KITTI[3] TRACKING TRAINING SET

	MOTA	IDS	Time
CAR CLASS			
loc	84.60 %	83	0.08 ms
kf	83.76 %	285	0.33 ms
$appl_{ast}$	82.38 %	617	2.07 ms
app_{16}	81.92 %	730	15.64 ms
TDFL	84.80 %	35	2.11 ms
PEDESTRIAN CLASS			
loc	42.56 %	96	0.08 ms
kf	40.86 %	285	0.33 ms
$appl_{ast}$	40.99 %	270	2.07 ms
app_{16}	40.28 %	349	15.64 ms
TDFL	42.81 %	68	2.11 ms

TABLE II
RESULTS IN KITTI[3] TRACKING TEST SET AS DISPLAYED IN THE BENCHMARK ONLINE

	MOTA	MOTP	MT	ML	IDS	MODA	MODP	Time
CAR CLASS								
TDFL* (ours)	84.30 %	85.63 %	71.69 %	2.92 %	328	85.26 %	88.78 %	0.01 s / 1 core
MOTBeyondPixels [13]	84.24 %	85.73 %	73.23 %	2.77 %	468	85.60 %	88.62 %	0.3 s / 1 core
IMMDP [16]	83.04 %	82.74 %	60.62 %	11.38 %	172	83.54 %	86.58 %	0.19 s / 4 cores
3D-CNN/PMBM [17]	80.39 %	81.26 %	62.77 %	6.15 %	121	80.74 %	85.47 %	0.01 s / 1 core
extraCK [18]	79.99 %	82.46 %	62.15 %	5.54 %	343	80.99 %	86.02 %	0.03 s / 1 core
TDFL (ours)	78.41 %	82.33 %	57.08 %	7.54 %	67	78.61 %	86.05 %	0.01 s / 1 core
NOMT* [19]	78.15 %	79.46 %	57.23 %	13.23 %	31	78.24 %	83.87 %	0.09 s / 16 cores
PEDESTRIAN CLASS								
MDP [16]	47.22 %	70.36 %	24.05 %	27.84 %	87	47.59 %	90.96 %	0.9 s / 8 cores
SCEA* [20]	43.91 %	71.86 %	16.15 %	43.30 %	56	44.15 %	92.71 %	0.06 s / 1 core
RMOT* [21]	43.77 %	71.02 %	19.59 %	41.24 %	153	44.43 %	92.38 %	0.02 s / 1 core
CIWT* [22]	43.37 %	71.44 %	13.75 %	34.71 %	112	43.86 %	92.43 %	0.28 s / 1 core
NOMT-HM* [19]	39.26 %	71.14 %	21.31 %	41.92 %	184	40.05 %	92.43 %	0.09 s / 8 cores
TDFL (ours)	30.43 %	72.73 %	11.34 %	43.30 %	100	30.86 %	93.08 %	0.01 s / 1 core

Fig. 3. Tracking results on few frames on KITTI[3] tracking sets

For the appearance based method we compare the performance from features from the last layer app_{last} as well as from a middle layer. As the middle layer 16^{th} layer app_{16} is considered with dimensions 128, 36, 128 for width, height and depth. Results show that last layer features are more robust and hence a higher tracking accuracy. When the proposed location based method and the appearance based method using features from the last layer are combined together, we get the proposed overall tracking method *TDFL* achieving 84.80% and 42.81% MOTA for car class and pedestrian class respectively. Average run time for the detection DNN is 69 ms per frame. Computation time for the proposed tracking method depends on the complexity of the scene, proportional to the number of objects detected in the frame. Therefore, the running time recorded is an average. In certain frames where no object is detected, tracking does not run and in certain frames with large amounts of objects, as shown in Fig. 3 time per frame can even be approximately 20 ms per frame, still achieving real time performance.

In Table 2, the proposed method is compared with the best tracking frameworks in the KITTI [3] benchmark that are published and are vision based online multiple object

tracking methods. MOTA is considered as the main parameter when ranking. *TDFL* is the proposed method using the trained YOLOv3 [32] network as the detection DNN and *TDFL** is the same with RRC [14] bounding box results. The proposed *TDFL* method has the fastest implementation time among all the methods, while achieving satisfactory MOTA for the car class. In KITTI [3] benchmark the lowest amount that can be entered for time is 0.01 s. In 3D-CNN/PMBM [17] authors record a running time of 73 fps for the test set while our method can achieve 500+ fps (average of 1.48 ms per frame). The DNN used in the proposed method for object classification runs very fast with an average speed of 69 ms per frame for test sequence. However, higher speeds compromise detection accuracy which will adversely affect the tracking accuracy, hence MODA is the second last rank for car class and last rank for pedestrian class. Therefore, in the results it is visible that the MODA and MOTA is proportionate to each other. The proposed *TDFL* method has the second lowest IDS for car class and third lowest for pedestrian class. Therefore, to check the effect from the detection method we utilize the RRC [14] detection results for the car class. RRC [14] is one of the two detection

frameworks used in MOTBeyondPixels [13] and authors have made RRC [14] bounding box information for KITTI [3] tracking set publicly available. Tracking results with this data is presented under *TDFL** and it has achieved the best MOTA, even it has slightly lower MODA than MOTBeyondPixels [13]. Even though the RRC [14] has higher accuracy, authors have recorded 3.6 s running time per frame in high-end GPU.

V. CONCLUSION

In this paper we have presented an efficient frame work for multiple object tracking, where the overall framework with classification DNN can achieve 14 fps while the tracking framework alone can achieve 500+ fps. Thus, the proposed method is highly suitable for real time applications and the results show that the proposed method achieve state-of-the-art performance as well.

VI. REFERENCES

REFERENCES

- [1] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.
- [2] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.
- [3] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.
- [4] W. Luo, J. Xing, A. Milan, X. Zhang, W. Liu, X. Zhao, and T.-K. Kim, "Multiple object tracking: A literature review," *arXiv preprint arXiv:1409.7618*, 2014.
- [5] A. Adam, E. Rivlin, and I. Shimshoni, "Robust fragments-based tracking using the integral histogram," in *Computer vision and pattern recognition, 2006 IEEE Computer Society Conference on*, vol. 1. IEEE, 2006, pp. 798–805.
- [6] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 2544–2550.
- [7] Z. Kalal, K. Mikolajczyk, J. Matas, *et al.*, "Tracking-learning-detection," *IEEE transactions on pattern analysis and machine intelligence*, vol. 34, no. 7, p. 1409, 2012.
- [8] Y. Wu, J. Lim, and M.-H. Yang, "Online object tracking: A benchmark," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2013, pp. 2411–2418.
- [9] 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, 2015.
- [10] B. Lee, E. Erdenee, S. Jin, M. Y. Nam, Y. G. Jung, and P. K. Rhee, "Multi-class multi-object tracking using changing point detection," in *European Conference on Computer Vision*. Springer, 2016, pp. 68–83.
- [11] R. Henschel, L. Leal-Taixé, D. Cremers, and B. Rosenhahn, "Fusion of head and full-body detectors for multi-object tracking," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE, 2018, pp. 1509–150909.
- [12] S. Tang, M. Andriluka, B. Andres, and B. Schiele, "Multiple people tracking by lifted multicut and person reidentification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3539–3548.
- [13] S. Sharma, J. A. Ansari, J. Krishna Murthy, and K. Madhava Krishna, "Beyond pixels: Leveraging geometry and shape cues for online multi-object tracking," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2018.
- [14] J. QiongYan and Y. LiXu, "Accurate single stage detector using recurrent rolling convolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
- [15] Y. Xiang, W. Choi, Y. Lin, and S. Savarese, "Subcategory-aware convolutional neural networks for object proposals and detection," in *Applications of Computer Vision (WACV), 2017 IEEE Winter Conference on*. IEEE, 2017, pp. 924–933.
- [16] Y. Xiang, A. Alahi, and S. Savarese, "Learning to track: Online multi-object tracking by decision making," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 4705–4713.
- [17] S. Scheidegger, J. Benjaminsson, E. Rosenberg, A. Krishnan, and K. Granström, "Mono-camera 3d multi-object tracking using deep learning detections and PMBM filtering," in *2018 IEEE Intelligent Vehicles Symposium, IV 2018, Changshu, Suzhou, China, June 26-30, 2018*, 2018, pp. 433–440.
- [18] G. Gündüz and T. Acarman, "A lightweight online multiple object vehicle tracking method," in *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 427–432.
- [19] W. Choi, "Near-online multi-target tracking with aggregated local flow descriptor," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 3029–3037.
- [20] J. Hong Yoon, C.-R. Lee, M.-H. Yang, and K.-J. Yoon, "Online multi-object tracking via structural constraint event aggregation," in *Proceedings of the IEEE Conference on computer vision and pattern recognition*, 2016, pp. 1392–1400.
- [21] J. H. Yoon, M.-H. Yang, J. Lim, and K.-J. Yoon, "Bayesian multi-object tracking using motion context from multiple objects," in *Applications of Computer Vision (WACV), 2015 IEEE Winter Conference on*. IEEE, 2015, pp. 33–40.
- [22] A. Osep, W. Mehner, M. Mathias, and B. Leibe, "Combined image-and world-space tracking in traffic scenes," in *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE, 2017, pp. 1988–1995.
- [23] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool, "Robust tracking-by-detection using a detector confidence particle filter," in *Computer Vision, 2009 IEEE 12th International Conference on*. IEEE, 2009, pp. 1515–1522.
- [24] D. Held, S. Thrun, and S. Savarese, "Learning to track at 100 fps with deep regression networks," in *European Conference Computer Vision (ECCV)*, 2016.
- [25] L. Leal-Taixé, C. Canton-Ferrer, and K. Schindler, "Learning by tracking: Siamese cnn for robust target association," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2016, pp. 33–40.
- [26] A. Sadeghian, A. Alahi, and S. Savarese, "Tracking the untrackable: Learning to track multiple cues with long-term dependencies," in *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, 2017, pp. 300–311. [Online]. Available: <https://doi.org/10.1109/ICCV.2017.41>
- [27] K. Fang, Y. Xiang, X. Li, and S. Savarese, "Recurrent autoregressive networks for online multi-object tracking," in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, March 2018, pp. 466–475.
- [28] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social lstm: Human trajectory prediction in crowded spaces," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 961–971.
- [29] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.
- [30] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.
- [31] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*. Springer, 2016, pp. 21–37.
- [32] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.
- [33] X. Chen, K. Kundu, Y. Zhu, A. Berneshawi, H. Ma, S. Fidler, and R. Urtasun, "3d object proposals for accurate object class detection," in *NIPS*, 2015.
- [34] K. Bernardin and R. Stiefelhagen, "Evaluating multiple object tracking performance: the clear mot metrics," *Journal on Image and Video Processing*, vol. 2008, p. 1, 2008.
- [35] Y. Li, C. Huang, and R. Nevatia, "Learning to associate: Hybrid-boosted multi-target tracker for crowded scene," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 2953–2960.