Simultaneous Detection and Tracking with Motion Modelling for Multiple Object Tracking

Shijie Sun[1], Naveed Akhtar[2], Xiangyu Song[3], Huansheng Song[1(✉)], Ajmal Mian[2], and Mubarak Shah[4]

[1] Chang'an University, Xi'an, Shaanxi, China
{shijiesun,hshsong}@chd.edu.cn
[2] University of Western Australia, 35 Stirling Highway, Crawley, WA, Australia
{naveed.akhtar,ajmal.mian}@uwa.edu.au
[3] Deakin University, RWaurn Ponds, Victoria 3216, Melbourne, Australia
xiangyu.song@deakin.edu.au
[4] University of Central Florida, Orlando, FL, USA
shah@crcv.ucf.edu

Abstract. Deep learning based Multiple Object Tracking (MOT) currently relies on off-the-shelf detectors for tracking-by-detection. This results in deep models that are detector biased and evaluations that are detector influenced. To resolve this issue, we introduce Deep Motion Modeling Network (DMM-Net) that can estimate multiple objects' motion parameters to perform joint detection and association in an end-to-end manner. DMM-Net models object features over multiple frames and simultaneously infers object classes, visibility and their motion parameters. These outputs are readily used to update the tracklets for efficient MOT. DMM-Net achieves PR-MOTA score of 12.80 @ 120+ fps for the popular UA-DETRAC challenge - which is better performance and orders of magnitude faster. We also contribute a synthetic large-scale public dataset Omni-MOT for vehicle tracking that provides precise ground-truth annotations to eliminate the detector influence in MOT evaluation. This 14M+ frames dataset is extendable with our public script (Code at Dataset, Dataset Recorder, Omni-MOT Source). We demonstrate the suitability of Omni-MOT for deep learning with DMM-Net, and also make the source code of our network public.

Keywords: Multiple object tracking · Tracking-by-detection · Deep learning · Simultaneous detection and tracking.

1 Introduction

Multiple Object Tracking (MOT) [10,25,42,44] is a longstanding problem in Computer Vision [27]. Contemporary deep learning based MOT has widely

Electronic supplementary material The online version of this chapter (https://doi.org/10.1007/978-3-030-58586-0_37) contains supplementary material, which is available to authorized users.

A. Vedaldi et al. (Eds.): ECCV 2020, LNCS 12369, pp. 626–643, 2020.
https://doi.org/10.1007/978-3-030-58586-0_37

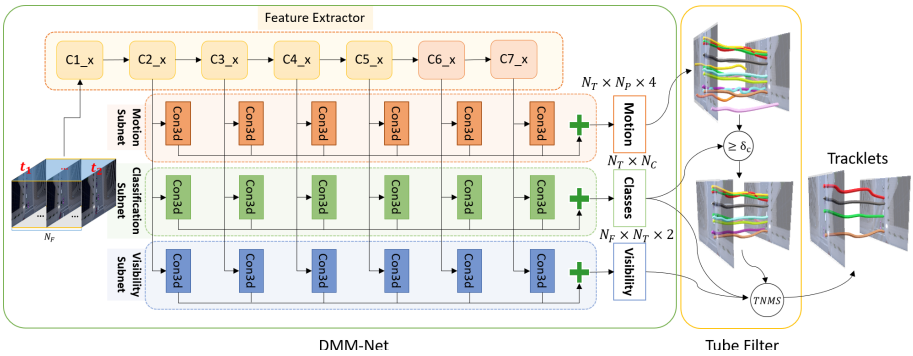

Fig. 1. Schematics of the proposed end-to-end trainable DMM-Net: N_F frames from time stamp $t_1 : t_2$ are input to the network. The frame sequence is first processed with a *Feature Extractor* comprising 3D ResNet-like [20] convolutional groups. Outputs of the last groups 2 to 7 are processed by Motion Subnet, Classifier Subnet, and Visibility Subnet. Each sub-network uses 3D convolutions to learn features that are concatenated to predict anchor tubes' motion parameters ($\boldsymbol{O}_M \in \mathbb{R}^{N_T \times N_P \times 4}$), object classes ($\boldsymbol{O}_C \in \mathbb{R}^{N_T \times N_C}$), and visibility ($\boldsymbol{O}_V \in \mathbb{R}^{N_F \times N_T \times 2}$), where N_T, N_P and N_C denote the number of anchor tubes, motion parameters and object classes (including background). We explain anchor tubes and motion parameters in Sect. 4. DMM-Net is trained with its specialized loss. For deployment, the anchor tubes predicted by the DMM-Net are filtered to compute tracklets defined over multiple frames. These tracklets are later combined to form a complete track.

adopted the tracking-by-detection paradigm [43], that capitalizes on the natural division of *detection* and *data association* tasks for the problem. In standard MOT evaluation protocol, the object detections are assumed to be known and public detections are provided on evaluation sequences, and MOT algorithms are expected to output object tracks by solving the data association problem.

Although attractive, using off-the-shelf detectors for MOT also has undesired ramifications. For instance, a deep model employed for the subsequent data association task (or a constituent sub-task) gets biased to the detector. The detector performance can also become a bottle-neck for the overall tracker. Additionally, composite techniques resulting from independent detectors fail to harness the true representation power of deep learning by sacrificing end-to-end training etc. It also seems unnatural that a MOT tracker must be evaluated on different detectors to interpret its tracking performance. All these problems are potentially solvable if trackers can implicitly detect the target objects, and detector bias can be removed from the ground-truth labeling of the tracks. This work makes strides towards these solutions.

We make two major contributions towards setting the tracking-by-detection paradigm free from off-the-shelf detectors in deep learning settings. Our first contribution comes as the first-of-its-kind deep network that performs object detections and data association by estimating multiple object motion parameters in an end-to-end manner. Our network, called Deep Motion Modeling Network

(DMM-Net), predicts object motion parameters, their classes and their visibility with the help of three sub-networks, see Fig. 1. These sub-networks exploit feature maps of frames in a video sequence that are learned with a Feature Extractor comprising seven 3D ResNet-like [20] convolutional groups. Instead of individual frames, DMM-Net simultaneously processes multiple (i.e. 16) frames. To handle multiple tracks in those frames, we introduce the notion of anchor tubes that extends the concept of anchor boxes in object detection [26] along the temporal dimension for MOT. Similar to [26], these anchor tubes are pre-defined to reduce the computation and improve the network speed. The predicted motion parameters can describe the shape offset and scale of each pre-defined anchor tube in the spatio-temporal space. We propose individual losses over the comprehensive representations of the sub-networks to predict object motion parameters, object classes and visibility. The DMM-Net output is readily usable to update the tracks. As our second major contribution, we propose a realistic large-scale dataset with accurate and extensive ground-truth annotations. The proposed dataset, termed Omni-MOT for its comprehensive coverage of the MOT scenarios, is generated with CARLA simulator [13] for vehicle tracking. The dataset comprises 14M+ frames, 250K tracks, 110 million bounding boxes, three weather conditions, three crowd levels and three camera views in five simulated towns. By eliminating the use of off-the-shelf detectors in ground-truth labeling, it removes any detector bias in evaluating the techniques.

The Omni-MOT dataset and DMM-Net source code are both publicly available for the broader research community. For the former, we also provide software tools to freely extend the dataset. We demonstrate the suitability of the Omni-MOT for deep models by training and evaluating DMM-Net on its videos. We also augment DMM-Net with Omni-MOT and evaluate our technique on the popular UA-DETRAC challenge [45]. The remote server computed results show that DMM-Net is able to achieve a very competitive 12.80 score for the comprehensive PR-MOTA metric with the overall speed of 123 fps. The orders of magnitude increase in the speed is directly attributed to the intrinsic detections in our tracking-by-detection technique.

2 Related Work

Multiple Object Tracking (MOT) is a fundamental problem in Computer Vision that has attracted significant interest of researchers in recent years. For a general review of the related literature, we refer to Luo et al. [27] and Emami et al. [14]. Here, we focus on the key contributions that relate to this work more closely. With the recent advances in object detectors, tracking-by-detection is fast becoming the common contemporary paradigm for MOT [38,39,43,46]. In this scheme, objects are first detected frame-wise and later associated with each other across multiple frames. Relying on off-the-shelf detectors, techniques following this paradigm mainly focus on object association, which can make them inherently detector biased. These methods can be broadly categorized as local [34,41] and global [10,37,48] approaches. The former use two frames

for data association, while the latter associate objects over a larger number of frames.

More recent global techniques cast data association into a network flow problem [4,7,33,40]. For instance, Berclaz et al. [4] solved a constrained flow optimization problem for MOT and used the k-shortest paths algorithm for associating the tracks. Chari et al. [8] added a pairwise cost to the min-cost network flow framework and proposed a convex relaxation of the problem. However, such methods rely on object detectors rather strongly, which makes them less attractive in the presence of occlusions and misdetections. To mitigate the problems resulting from occlusions, Milan et al. [29] employed a continuous energy minimization framework for MOT that incorporates explicit occlusion reasoning and appearance modeling. Wen et al. [47] also proposed a data association technique based on undirected hierarchical relation hyper-graph.

Sun et al. [42] proposed a deep affinity network to model features of pre-detected objects and compute object affinities by the same network. Bea et al. [3] modified the Siameses Network to learn deep representations for MOT with object association. There are few instances of deep learning techniques that aim at removing the reliance of tracking on independent detectors. For instance, Feichtenhofer et al. [15] proposed an R-FCN based network [9] that performs object detection and jointly builds an object model to be used for data association. However, their method is limited to frame-wise detections. Consequently, it only allows frame-by-frame association, requiring manual adjustment of temporal stride. Our technique is inherently different, as it directly computes tracklets over multiple frames by motion modeling, enabling realtime solutions while considering all the frames.

Besides the development of novel techniques, the role of datasets is central to the advancement of deep learning based MOT. Currently, a few large datasets for this task are available, e.g. PETS2009 [17], KITTI [18], DukeMTMC [36], PoseTrack [23], and MOT Challenge datasets [25,28]. These datasets are recorded in the real world with pedestrians and vehicles as the objects of interest. We refer to the original works for more details on these datasets. Below, we briefly discuss UA-DETRAC [45] for its high relevance to our contribution.

UA-DETRAC [45] is a large dataset for traffic scenes MOT. It provides object bounding boxes, their IDs and information on the overlapped ratio of the objects. However, the provided detections are individually generated by detectors and hence are prone to errors. This results in an undesired detector-bias in tracker evaluation. Besides, different pre-processing procedures of the detectors employed by the dataset also cause problems in fair evaluation. Although UA-DETRAC has served a great purpose in advancing the state-of-the-art in MOT for vehicles, the aforementioned issues call for a more transparent dataset that does not rely on off-the-shelf detectors for evaluating trackers. This work provides such a dataset with realistic settings and complete control over the environment conditions and camera viewpoints.

3 Omni-MOT Dataset

We term the proposed dataset as **O**mni-**MOT** (OMOT) dataset for its comprehensive coverage of the conditions and scenarios possible in MOT. The dataset is publicly available for download. Moreover, we also make the recording script for the dataset public that will enable the community to further extend the data The provided script has the ability to generate multi-camera videos. The dataset is recorded using virtual cameras in the CARLA simulator [13].

To the best of our knowledge, Omni-MOT is the first realistic large dataset for tracking vehicles that completely relinquishes off-the-shelf detectors in ground truth generation, and provides comprehensive annotations. Moreover, with the provided scripts, the dataset can easily be extended for future research. To put the scale of Omni-MOT into perspective, the provided number of frames is almost $1,200$ times larger than MOT17. The number of provided tracks and boxes respectively are 210 and 30 times larger than UA-DETRAC. Not to mention, all the boxes and tracks are simulator generated that avoids any labeling error. Considering that OMOT can also be used for data augmentation, we make the ground truth for the test videos public as well. Please see the supplementary material of the paper for complete details of the dataset.

4 Deep Motion Modeling Network

To absolve deep learning based tracking-by-detection from independently pre-trained off-the-shelf detectors, we propose **D**eep **M**otion **M**odeling **Net**work (DMM-Net) for online MOT (see Fig. 1). Our network enables MOT by jointly performing object detection, tracking, and classification across multiple video frames without requiring pre-detections and subsequent data association. For a given input video, it outputs objects' motion parameters, classes, and their visibility across the input frames. We provide a detailed discussion of our network below. However, we first introduce the used notations and conventions.

- N_F, N_C, N_P, N_T denote the number of input frames, object classes (0 for 'background'), time-related motion parameters, and anchor tubes.
- W, H are the frame width, and frame height.
- $\boldsymbol{I}_t$ denotes the video frame at time t. Subsequently, a 4-D tensor $\boldsymbol{I}_{t_1:t_2:N_F} \in \mathbb{R}^{3\times N_F\times W\times H}$ denotes N_F video frames from time t_1 to t_2-1. For simplicity, we often ignore the subscript ": N_F".
- $\boldsymbol{B}_{t_1:t_2:N_F}, \boldsymbol{C}_{t_1:t_2:N_F}, \boldsymbol{V}_{t_1:t_2:N_F}$ respectively denote the ground truth boxes, classes and visibilities in the selected N_F video frames from time t_1 to t_2-1. The text also ignores ": N_F" for these notations.
- $\boldsymbol{O}_{M,t_1:t_2:N_F}, \boldsymbol{O}_{C,t_1:t_2:N_F}, \boldsymbol{O}_{V,t_1:t_2:N_F}$ denote the estimated motion parameters, classes, and visiblities. With time stamps and frames clear from the context, we simplify these notations as O_M, O_C, O_V. In Fig. 2, we illustrate object visibility, classes and motion parameters.

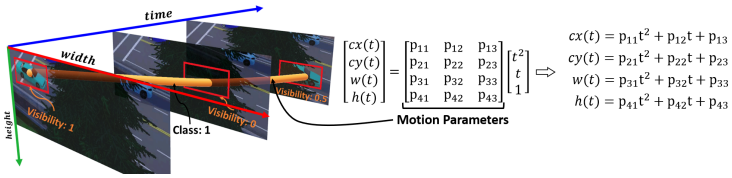

Fig. 2. (Best viewed in color) Illustration of motion visibility, class, and motion parameters: The visibility of the vehicle goes from 1 (fully visible) to 0 (invisible) and back to 0.5 (partially visible). Classes are predefined as 1 for vehicle and 0 for everything else. Motion parameters $\{p_{11}, \cdots, p_{43}\}$ are used to locate the center x (cx), center y (cy), width (w) and height (h) of the tube/tracklet at anytime. We employ a quadratic motion model leveraging 4×3 matrices. (Color figure online)

Table 1. Inputs for training and testing DMM-Net. The tensor dimensions are given as Channels×Duration×Width×Height. $n_{t_1:t_2}$ is the number of tracks in $I_{t_1:t_2}$.

Input	Dimensions/Size	Train	Test
$\boldsymbol{I}_{t_1:t_2}$	$3 \times N_F \times W \times H$	✓	✓
$\boldsymbol{t}_{t_1:t_2}$	N_F	✓	✓
$\boldsymbol{B}_{t_1:t_2}$	$N_F \times n_{t_1:t_2} \times 4$	✓	×
$\boldsymbol{C}_{t_1:t_2}$	$n_{t_1:t_2}$	✓	×
$\boldsymbol{V}_{t_1:t_2}$	$N_F \times n_{t_1:t_2}$	✓	×

Conventions: The shape of a network output tensor is considered to be Batch × Channels × Duration × Width × Height, where Duration accounts for the number of frames. For brevity, we often ignore the Batch dimension.

4.1 Data Preparation

Appropriate data preparation is important for training (and later testing) our network. In this process, we also avail the opportunity of augmenting the data to induce robustness in our model against camera photometric distortions, background scene variations, and other practical factors. The inputs expected by our network are summarized in Table 1. For training a robust network, we perform the following data pre-processing:

1. *Stochastic frame sequence* is introduced by randomly choosing N_F frames from $2N_F$ frames.
2. *Static scene emulation* is done by duplicating selected frames N_F times.
3. *Photometric camera distortions* are introduced in the frames by scaling each pixel by a random value in the range [0.7, 1.5]. The resulting frames are converted to HSV format, and their saturation channel is again scaled by a random value in [0.7, 1.5]. A frame is then converted back to RGB format and re-scaled by a random value in the same range. This process of photometric distortion is inspired by [42].

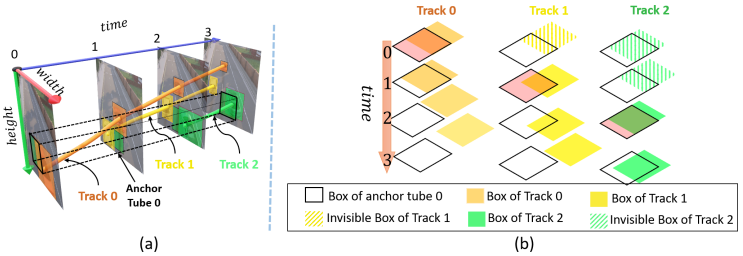

Fig. 3. (a) One anchor tube and three tracks are shown to illustrate the search for the best track of the anchor tube. (b) is the simplified demonstration of (a). Based on the largest overlap between the first tube box and the first *visible* track box, Track 2 is selected as the best match. (Color figure online)

4. *Frame expansion* is performed by enlarging video frames by a random factor in the range [1, 1.2]. To that end, we pad the original frames with extra pixels whose value is set to the mean pixel value of the training data.

In our pre-processing, the above-mentioned steps 2–4 are applied to the frames with probability 0.01, 0.5, 0.5, respectively. The resulting frame are then resized to the fixed dimension $H \times W \times 3$, and horizontally flipped with a probability of 0.5. We simultaneously process the selected N_F frames by applying the above mentioned transformations. These RGB video frames are then arranged as 4D-tensors $\boldsymbol{I}_{t_1:t_2} \in \mathbb{R}^{3\times N_F\times W\times H}$. We fill the ground truth data matrices $\boldsymbol{B}_{t_1:t_2}$, $\boldsymbol{C}_{t_1:t_2}$, $\boldsymbol{V}_{t_1:t_2}$ with the detected boxes, and visibilities in the video frames from t_1 to $t_2 - 1$. We set the labeled boxes, classes, and visibilities of fully occluded boxes to 0. We also remove the tracks whose ratio of fully occluded boxes are greater than δ_v. We let $N_F = 16$ in our experiments.

4.2 Anchor Tubes

Inspired by the effective Single Shot Detector (SSD) [26] for object detection, we extend the core concept of this technique for MOT. Analogous to the anchor boxes of SSD, we introduce *anchor tubes* for DMM-Net. Here, an anchor tube is a set of anchor boxes (and associated object class and visibility), that share the same location in multiple frames along the temporal dimension (see supplementary material for further visualization). The information of anchor tubes is encoded with the tensors $\boldsymbol{B}_{t_1:t_2}$, $\boldsymbol{C}_{t_1:t_2}$, and $\boldsymbol{V}_{t_1:t_2}$. The DMM-Net is designed to predict the tube shape offset parameters along the temporal dimension, the confidence for each object class, and the visibility of each box in the tube.

Computing anchor tubes for network training can also be interpreted as further data preparation for DMM-Net. We first employ a search strategy to find the best-matched track for each anchor tube, and subsequently encode the tubes by their best-matched tracks. For the former, we specify the overlap between an anchor tube and a track as the overlap ratio between the first visible box of the track and the corresponding box in the anchor tube. A simplified illustration of

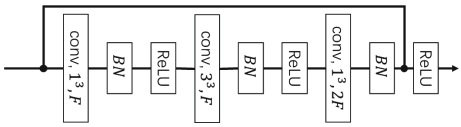

Fig. 4. Used ResNeXt block: (conv, x^3, F) denote 'F' convolutional kernels of size $x \times x \times x$. BN is Batch Normalization [22] and shortcut connection is summation.

this concept is presented in Fig. 3 with one anchor tube and three tracks. The anchor tube 0 iterates over all the tracks to find the largest overlap ratio. The first box of Track 1 and the first two boxes of Track 2 are occluded. Therefore, the overlap ratio of the anchor box filled red is used for selecting the best-matched track, i.e. Track 2 for the largest overlap.

To encode the anchor tubes, we employ their best-matched tracks ($\boldsymbol{B}_{t_1:t_2}$) along with the classes ($\boldsymbol{C}_{t_1:t_2}$) and visibility ($\boldsymbol{V}_{t_1:t_2}$). We denote a box of the i^{th} anchor tube at the t^{th} frame as $a_{i,t} = (a^{cx}_{i,t}, a^{cy}_{i,t}, a^{w}_{i,t}, a^{h}_{i,t})$, where the superscripts cx and cy indicate the (x, y) location of the center, and w, h denote the width and height of the box. We use the same notation in the following text as well. Each anchor box is encoded by the box of its best-matched track as follows:

$$\begin{cases} g^{w}_{i,t} = log(b^{w}_{i,t}/a^{w}_{i,t}) \\ g^{h}_{i,t} = log(b^{h}_{i,t}/a^{h}_{i,t}) \\ g^{cx}_{i,t} = (b^{cx}_{i,t} - a^{cx}_{i,t})/a^{w}_{i,t} \\ g^{cy}_{i,t} = (b^{cy}_{i,t} - a^{cy}_{i,t})/a^{h}_{i,t}, \end{cases} \tag{1}$$

where $(b^{cx}_{i,t}, b^{cy}_{i,t}, b^{w}_{i,t}, b^{h}_{i,t})$ describe the box of the best-matched track at the t^{th} frame, and $(g^{cx}_{i,t}, g^{cy}_{i,t}, g^{w}_{i,t}, g^{h}_{i,t})$ represent the resulting encoded box for the best-matched track. Following this encoding, we replace each original box, class, and visibility by its newly encoded counterpart. In the above-mentioned encoding, an anchor tube that has its best-matched track overlap ratio less than δ_o, is identified as the 'background' class.

4.3 Motion Model

For motion modeling, we leverage a quadratic model in time. One of the outputs of DMM-Net is a tensor of motion parameters $\boldsymbol{O}_M \in \mathbb{R}^{N_T \times 4 \times N_P}$, where $N_P = 3$ in our experiments and N_T indicates the number of anchor tubes. We estimate an encoded anchor tube for a track as:

$$\begin{cases} \hat{g}^{w}_{i,t} = p_{11}t^2 + p_{12}t + p_{13} \\ \hat{g}^{h}_{i,t} = p_{21}t^2 + p_{22}t + p_{23} \\ \hat{g}^{cx}_{i,t} = p_{31}t^2 + p_{32}t + p_{33} \\ \hat{g}^{cy}_{i,t} = p_{41}t^2 + p_{42}t + p_{43}, \end{cases} \tag{2}$$

where $\hat{g}_{i,t}$ indicates an estimated box descriptor of the i^{th} encoded anchor tube at the t^{th} frame, the superscripts cx, cy, w and h indicate the center x, center y,

Table 2. Convolutional groups of the Feature Extractor. Conv is abbreviated as 'C', 'F' is the number of feature maps, as in Fig. 4, and 'N' is the number of bottleneck blocks in each layer.

C1_x	C2_x		C3_x		C4_x		C5_x		C6_x		C7_x	
conv, 7^3, 64	F	N	F	N	F	N	F	N	F	N	F	N
	64	3	128	4	256	23	512	3	32	3	32	3

height, width of the box respectively, and $\{p_{11}, \cdots, p_{43}\}$ are the motion parameters . Each encoded ground-truth anchor tube can be decoded into a ground truth track by Eq. (1). We further elaborate on the relationship between the motion parameters and the ground truth tracks in the supplementary material. Note that the used motion function is replaceable by any differential function in our technique. We choose quadratic motion modeling considering vehicle tracking as our primary objective.

4.4 Architecture

As shown in Fig. 1, our network comprises a Feature Extractor and three sub-network. The network simultaneously processes N_F video frames for which features are first extracted. The feature maps of six intermediate layers of the Feature Extractor are fed to the sub-networks. These networks predict tensors containing motion parameters ($\boldsymbol{O}_M \in \mathbb{R}^{N_T \times N_P \times 4}$), object classes ($\boldsymbol{O}_C \in \mathbb{R}^{N_T \times N_C}$), and visibility information ($\boldsymbol{O}_V \in \mathbb{R}^{N_F \times N_T \times 2}$).

Feature Extractor: We construct our Feature Extractor based on the ResNeXt blocks of the 3D ResNet [20][1]. The architectural details of the blocks are provided in Fig. 4. The blocks accept 'F' channel input that allows us to simultaneously model spatio-temporal features in N_F frames. We enhance the 3D ResNet architecture by removing the *fully-connected* and *softmax* layers of the network and appending two extra convolutional groups denoted as Conv6_x and Conv7_x. We perform this enhancement because the additional convolutional layers are still able to encode further higher level spatio-temporal features of the frames. Details on the convolutional groups used by the Feature Extractor are provided in Table 2. Here, Conv1_x (abbreviated as C1_x) contains 64 convolutional kernels of size $7 \times 7 \times 7$, stride $1 \times 2 \times 2$. A $3 \times 3 \times 3$ max-pooling layer with stride 2 is inserted before Conv2_x for down sampling. Each convolutional layer is followed by batch normalization [21] and ReLU [31]. Spatio-temporal down-sampling is performed by Conv3_1, Conv4_1 and Conv5_1 with a stride of 2.

Sub-networks: The DMM-Net has three sub-networks to compute motion parameters, object classes, and their visibility. These networks use the outputs of Conv2_x to Conv7_x groups of the Feature Extractor. Each sub-network processes the input with six convolutional layers. The architectural details of

[1] Notation are adopted from the original work.

Table 3. Details of the Motion Subnet (M.S.), Classifier Subnet (C.S.), and Visibility Subnet (V.S.): K.S., P.S., S.S., and N.K. respectively denote the kernel shape, padding size, stride size, and the number of kernels. The row 'Output' reports output tensor shape. $\boldsymbol{\mathcal{K}} = \{10, 8, 8, 5, 4, 4\}$ is the number of anchor tubes for each of the six input feature maps for each sub-network.

	M.S	C.S	V.S
K.S.	$\{8, 4, 2, 1, 1, 1\} \times 3 \times 3$		
P.S.	$0 \times 1 \times 1$		
S.S.	$1 \times 1 \times 1$		
N.K.	$4\boldsymbol{\mathcal{K}}N_P$	$\boldsymbol{\mathcal{K}}N_C$	$2\boldsymbol{\mathcal{K}}$
Output	$N_T \times N_P \times 4$	$N_T \times N_C$	$N_F \times N_T \times 2$

all three sub-networks are summarized in Table 3. The kernel shape, padding size and stride are the same for each network, whereas the employed numbers of kernels are different. We fix the temporal kernel size for each network to $\{8, 4, 2, 1, 1, 1\}$. Denoting the numbers of anchor tubes defined for the six input feature maps of a sub-network by $\boldsymbol{\mathcal{K}}$, we let $\boldsymbol{\mathcal{K}} = \{10, 8, 8, 5, 4, 4\}$ in our experiments. For each sub-network, we concatenate the output feature maps of each convolutional layer. We reshape the concatenated feature according to the dimensions mentioned in the last row of the table. These outputs are subsequently used to compute the network loss.

Network Loss: Based on the architecture, the overall loss of DMM-Net is defined as a weighted sum of three sub-losses, including the motion loss ($\mathcal{L}_M$), the classification loss ($\mathcal{L}_C$) and the visibility loss ($\mathcal{L}_V$), given as:

$$\mathcal{L} = \frac{1}{N}(\alpha\mathcal{L}_M + \beta\mathcal{L}_C + \mathcal{L}_V), \tag{3}$$

where N is the number of *positive* anchor tubes, and α, β are both hyper-parameters. The positive anchor tubes exclude the background tubes.

The motion loss $\mathcal{L}_M$ is defined as the sum of *Smooth*-L1 losses between the ground truth encoded tracks $g_{i,t}$ and their predicted encoded tracks $\hat{g}_{i,t}$, formally:

$$\mathcal{L}_M = \sum_{t=0}^{N_F-1} \sum_{i \in Pos} \sum_{m \in \{cx,cy,h,w\}} ||g_{i,t}^m - \hat{g}_{i,t}^m||_1, \tag{4}$$

where Pos is the set of positive encoded anchor tube indices. We compute $\hat{g}_{i,t}^m$ using Eq. (2) discussed in Sect. 4.3.

For the classification loss $\mathcal{L}_C$, we employ the hard negative mining strategy [26] to balance the positive and negative anchor tubes, where the negative tubes correspond to the background. We denote $x_{i,j,t}^p \in \{1, 0\}$ to be an indicator for matching the i^{th} anchor tube and the j^{th} ground-truth track of class 'p' at the t^{th} frame. Each anchor tube at least has one best-matched ground-truth

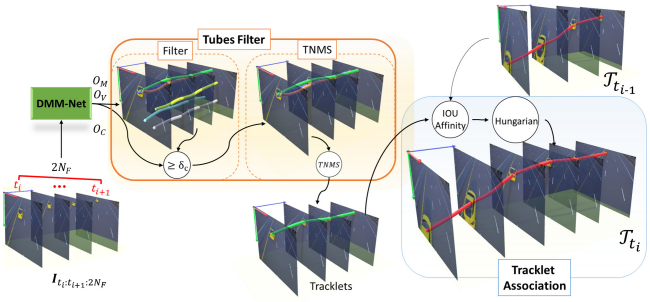

Fig. 5. Deployment of DMM-Net: For the $2N_F$ frames $\boldsymbol{I}_{t_i:t_{i+1}:2N_F}$ from t_i to t_{i+1}, the trained DMM-Net selects N_F frames as its input and outputs predicted tubes encoded by object motion parameter matrix ($\boldsymbol{O}_M$), classification matrix ($\boldsymbol{O}_C$) and visibility matrix ($\boldsymbol{O}_V$). These matrices are filtered and the track set $\boldsymbol{\mathcal{T}}_{t_i}$ is updated by associating the filtered tracklets by their IOU with the previous track set $\boldsymbol{\mathcal{T}}_{t_{i-1}}$.

track, implying $\sum_i x^p_{i,j,t} \geq 1$ for any t. The classification loss of the DMM-Net is defined as:

$$\mathcal{L}_C = \sum_{t=0}^{N_F} \left(- \sum_{i \in Pos} x^p_{i,j,t} \, log(\hat{c}^p_{i,t}) - \sum_{i \in Neg} log(\hat{c}^0_{i,t}) \right), \tag{5}$$

where $\hat{c}^p_{i,t} = \frac{exp\, c^p_{i,t}}{\sum_p exp\, c^p_{i,t}}$ such that $c^p_{i,t}$ is the predicted confidence of the object being for class $p : p \in \{1, \cdots, N_C\}$, Neg is the set of negative encoded anchor tube indices. We also consider the visibility estimation task fro classification viewpoint and classify each positive box into invisible '0' or visible '1' box. Based on that, the loss is computed as:

$$\mathcal{L}_V = \sum_{t=0}^{N_F} \left(- \sum_{i \in Pos} y^q_{i,j,t} \, log(\hat{v}^q_{i,t}) \right), \tag{6}$$

where $\hat{v}^q_{i,t} = \frac{exp\, v^q_{i,t}}{\sum_q exp\, v^q_{i,t}}$, and $y^q_{i,j,t}$ is an indicator for matching the i^{th} anchor tube and the j^{th} ground-truth track of visibility 'q' at the t^{th} frame, such that $q \in \{0, 1\}$.

4.5 Deployment

The trained DMM-Net is readily deployable for tracking (Fig. 5). The overall tracker processes $2N_F$ frames $\boldsymbol{I}_{t_i:t_{i+1}:2N_F}$, where the DMM-Net first selects N_F consecutive frames and outputs the predicted tubes encoded with object motion parameters, classes and visibility using Eq. (2). The tubes are decoded with Eq. (1). A filtration is then done to remove the undesired tubes and we get

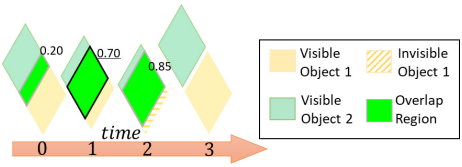

Fig. 6. Example of calculating Tube IOU. There are two tracklets (yellow, cyan). Their intersection is green. The tracklet IOU is the maximum IOU of the visible box pair. Although IOU at $t = 2$ is the largest, the lined yellow object is invisible, hence, we select the second largest IOU at $t = 1$ as the tracklet IOU.

tracklets (details to follow). We compute updated trajectory set $\boldsymbol{\mathcal{T}}_{t_i}$ by associating the tracklets with the previous trajectory set $\boldsymbol{\mathcal{T}}_{t_{i-1}}$ using their IOUs.

To filter, we first remove tubes with confidence lower than a threshold δ_c. We subsequently perform a Tube None Maximum Suppression (TNMS). To that end, we cluster the detected tubes of the same category into multiple groups by their IOUs with a threshold δ_{nms}. Then, we only keep one tube for each group that has the maximum confidence of being positive. The kept tubes, namely "tracklets", are employed to update trajectory set.

We initialize our track set $\boldsymbol{\mathcal{T}}_{t_i}$ with as many trajectories as the number of tracklets. The track set is updated from t_i^{th} to t_{i+1}^{th} stamp using the Hungarian algorithm [30] applied to an IOU matrix $\boldsymbol{\Psi} \in \mathbb{R}^{n'_{t_{i-1}} \times n_{t_i}}$, where $n'_{t_{i-1}}$ is the number of element in the previous track set, and n_{t_i} is the number of tracklets. Notice that we perform association of tracklets and not the individual objects. The object association remains implicit and is done inside the DMM-Net which lets us define the tubes. The association of tracklets, defined across multiple frames, leads to significant computational gain. This is one of the core reasons of the orders of magnitude gain in the tracking speed of our technique over existing methods, as will be clear in Sect. 5.

To form the matrix $\boldsymbol{\Psi}$, we must compute IOU between the existing tracklets and the new tracklets. Figure 6 shows the procedure adopted for calculating the IOU between two tracklets with a simplified example. Overall, our tracker is an on-line technique in the sense that it does not use future frames to predict object trajectories. Concurrently, DMM-Net can perform tracking in real-time that makes it a highly desirable technique for practical applications.

5 Experiments

We evaluate our technique using the proposed Omni-MOT dataset and the popular UA-DETRAC challenge [45]. The former is to demonstrate both the effectiveness of our network for MOT and trainability of deep models for MOT with Omni-MOT. We mainly focus on 'vehicle' tracking in this work. The proposed dataset is also for the vehicle tracking problem. Whereas DMM-Net is trained here for vehicle tracking, it is possible to train it for e.g.. pedestrian tracking.

Table 4. Results on Omni-MOT: The symbol ↑ indicates higher values are better, and ↓ implies lower values are favored.

Type	Camera	IDF1↑	IDP↑	IDR↑	Rcll↑	Prcn↑	GT↑	MT↑	PT↑	ML↓	FP↓	FN↓	IDs↓	FM↓	MOTA↑	MOTP↑
Test	Camera_0	72.2%	69.3%	75.4%	90.6%	83.2%	41	38	3	0	1762	911	18	98	72.1%	79.3%
	Camera_1	59.3%	56.2%	62.7%	81.9%	73.4%	35	19	15	1	1199	730	12	61	52.0%	75.5%
	Camera_5	40.2%	61.3%	29.9%	34.7%	71.1%	44	9	20	15	2779	12855	38	80	20.4%	76.1%
	Camera_7	77.8%	79.6%	76.1%	88.3%	92.4%	37	29	8	0	1518	2457	20	99	80.9%	80.3%
	Camera_8	30.5%	47.3%	22.5%	33.5%	70.7%	38	0	28	10	1789	8548	36	110	19.3%	70.3%
Train	Camera_0	59.2%	58.6%	59.7%	87.7%	86.0%	48	40	8	0	2517	2174	43	172	73.2%	79.5%
	Camera_1	47.6%	44.2%	51.6%	80.0%	68.6%	46	29	16	1	3681	2005	50	137	42.9%	76.9%
	Camera_5	45.0%	52.7%	39.2%	57.0%	76.7%	49	6	37	6	4844	12000	122	234	39.2%	77.3%
	Camera_7	74.7%	71.9%	77.8%	90.1%	83.3%	43	31	12	0	1804	985	16	91	71.8%	79.7%
	Camera_8	41.7%	52.2%	34.7%	44.1%	66.3%	43	0	41	2	1884	4705	18	117	21.4%	68.3%
Average		55.5%	61.1%	50.9%	66.5%	79.8%	424	201	188	35	23777	47370	373	1199	49.4%	77.8%

Fig. 7. Tracking illustration of DMM-Net on Omni-MOT (first and second row) and UA-DETRAC (third row).The mentioned IDs are for reference in the text only.

Implementation Details: We implement the DMM-Net using Pytorch [32] and train it using NVIDIA GeForce GTX Titan GPU. The hyper-parameter values are selected with cross-validation to maximize the MOTA metric on the validation set of UA-DETRAC. The chosen values of the hyperparameters are as follows. Batch size $B = 8$, number of training epochs per model = 20, number of input frames = 16, input frame size = 168×168, We use Adam Optimizer [24] for training. Other hyper-parameter values are $\delta_c = 0.4, \delta_{nms} = 0.3$, and $\delta_o = 0.5$.

Omni-MOT Dataset: To demonstrated the efficacy of DMM-Net and trainability of deep models on our dataset, we select five scenes from the Omni-MOT and perform tracking. The scenes are chosen with Easy camera viewpoint with clear weather conditions. The selected scenes are from Town 02 (with 50 vehicles) that are indexed 0, 1, 5, 7 and 8 in the dataset. In this experiment, we train and test DMM-Net using the train and test partitions of the selected scene, as provided by Omni-MOT. Our training required 59.2 h for 22 epochs.

We use both CLEAR MOT [5], and MT/ML [35] metrics and summarize the results in Table 4. For the definitions of metrics we refer to the original

Table 5. Results on UA-DETRAC: 'T.S.' is tracker speed (fps), 'D.S.' is detector speed (fps), and 'A.S.' is the overall speed (fps). The DMM-Net does not need explicit detector. DMM-Net+ uses a subset of Omni-MOT for data augmentation.

Tracker	Detector	PR-MOTA↑	PR-MT↑	PR-ML↓	PR-IDS↓	PR-FRAG↓	PR-FP↓	PR-FN↓	T.S.↑	D.S.↑	A.S.↑
CEM [1]	ACF [12]	4.50%	2.90%	37.10%	265.4	366.0	15180.3	270643.2	3.74	0.67	0.57
CMOT [49]	ACF [12]	7.80%	14.30%	20.70%	418.3	2161.7	81401.4	183400.2	3.12	0.67	0.55
DCT [2]	ACF [12]	7.90%	4.80%	34.40%	108.1	101.4	13059.7	251166.4	1.29	0.67	0.44
GOG [33]	ACF [12]	10.80%	12.20%	22.30%	3950.8	3987.3	45201.5	197094.2	319.29	0.67	0.67
H2T [47]	ACF [12]	8.20%	13.10%	21.30%	1122.8	1445.8	71567.4	189649.1	1.08	0.67	0.41
IHTLS [11]	ACF [12]	6.60%	11.50%	22.40%	1243.1	4723.0	72757.5	198673.5	5.09	0.67	0.59
CEM [1]	DPM [16]	3.30%	1.30%	37.80%	265	317.1	13888.7	270718.5	4.49	0.17	0.16
DCT [2]	DPM [16]	2.70%	0.50%	42.70%	72.2	**68.8**	7785.8	280762.2	2.85	0.17	0.16
GOG [33]	DPM [16]	5.50%	4.10%	27.70%	1873.9	1988.5	38957.6	230126.6	476.52	0.17	0.17
IOUT [6]	DPM [16]	1.92%	0.84%	43.70%	**61.4**	106.0	**3111.5**	290412.2	100842	0.17	0.17
CEM [1]	R-CNN [19]	2.70%	2.30%	34.10%	778.9	1080.4	34768.9	269043.8	5.4	0.1	0.10
DCT [2]	R-CNN [19]	11.70%	10.10%	22.80%	758.7	742.9	336561.2	210855.6	0.71	0.1	0.09
CMOT [49]	R-CNN [19]	11.00%	**15.70%**	19.00%	506.2	22551.1	74253.6	**177532.6**	3.59	0.1	0.10
GOG [33]	R-CNN [19]	10.00%	13.50%	20.10%	7834.5	7401.0	58378.5	192302.7	352.8	0.1	0.10
H2T [47]	R-CNN [19]	11.10%	14.60%	19.80%	1481.9	1820.8	66137.2	184358.2	2.78	0.1	0.10
IHTLS [11]	R-CNN [19]	8.30%	12.00%	21.40%	1536.4	5954.9	68662.6	199268.8	11.96	0.1	0.10
DMM-Net	-	**11.80%**	10.30%	**15.20%**	230.3	658.0	36238.8	194886.4	-	-	**123.25**
DMM-Net+	-	**12.20%**	10.80%	**14.90%**	228.2	674.1	36355.8	192289.6	-	-	**123.25**

works. We refer to [25] for the details on the comprehensive metrics MOTA and MOTAP. The table provides metric values for both training and test partitions. The variation between these values is a clear indication that despite the Easy camera view, the dataset provides reasonable challenges for the MOT task. We provide results with additional view points in the supplementary material to further put the reported values into better perspective.

An illustration of tracking for Camera_0 is also provided in Fig. 7 (top). Our technique is able to easily track vehicles that are stationary, (e.g. 1), moving along a straight path (e.g. 2), and moving along a curved path (e.g. 3), which justifies the selection of our motion model. Additionally, our tracker has the power to deal with even full occlusions between the frames. We show such a case for a hard camera view point in the middle row of Fig. 7. In this case, vehicle-1 is totally occluded at the 86^{th} frame which can be detected and tracked by our tracker. At the 99^{th} frame, vehicle-1 is partially occluded, but the tracklets association performs correctly. Note that, our tracker does not need to be evaluated with (detector-based) UA-DETRAC metrics [45] as the Omni-MOT dataset provides ground truth detections without using off-the-shelf detectors.

UA-DETRAC: The UA-DETRAC challenge [45] is arguably the most widely used large-scale benchmark for MOT of vehicles. It comprises 100 videos @ 25 fps, recorded in 24 locations with frame size 540×960. For this challenge, results are computed by a remote server using CLEAR MOT and MT/ML metrics with Precision-Recall curve of the detection. We refer to [45] for further details on the metrics. In Table 5, we show our results on the UA-DETRAC challenge. The pre-fix PR for the metrics indicates the use of Precision-Recall curve. It can be seen that DMM-Net achieves excellent MOTA score, which is widely

considered as the most comprehensive metric for MOT. We also provide results for DMM-Net+ which augments the UA-DETRAC training set with a subset of Omni-MOT. This subset contained 8 random videos from Omni-MOT. We can see an overall performance gain with this augmentation, exemplifying the utility of Omni-MOT in data augmentation.

Notice that our technique does not require an external 'detector' and achieves highly promising PR-MOTA score for UA-DETRAC @ 120+ fps - orders of magnitude faster than the existing methods. In the third row of Fig. 7, we show an example of tracking result. The box color indicates the predicted object identity. Our tracker leverages information from previous frames to detect partially occluded objects, e.g.. vehicle-3 in frame 518. The figure also shows the generalization power of our in-built detection mechanism. For instance, vehicle-1 is a very rare vehicle that is consistently assigned the correct identity. In the supplementary material, we provide further example tracking videos that clearly demonstrate successful tracking by DMM-Net for UA-DETRAC.

6 Conclusion

In the context of tracking-by-detection, we proposed a deep network DMM-Net that removes the need for an explicit external detector and performs tracking at 120+ fps to achieve 12.80% PR-MOTA value on the UA-DETRAC challenge. Our network meticulously models object motions, classes and their visibility that are subsequently used for efficiently associating the object tracks. The proposed network provides an end-to-end solution for detection and tracklet generation across multiple video frames. We also propose a realistic CARLA simulator based large-scale dataset with over 14M frames for vehicle tracking. The dataset provides precise and comprehensive ground truth with full control over data parameters, which allows for the much needed transparency in evaluation. We also provide scripts to generate more data under our framework and we make the code of DMM-Net public.

References

1. Andriyenko, A., Schindler, K.: Multi-target tracking by continuous energy minimization. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 1265–1272 (2011). https://doi.org/10.1109/CVPR.2011.5995311
2. Andriyenko, A., Schindler, K., Roth, S.: Discrete-continuous optimization for multi-target tracking. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 1926–1933 (2012). https://doi.org/10.1109/CVPR.2012.6247893
3. Bae, S.H., Yoon, K.J.: Confidence-based data association and discriminative deep appearance learning for robust online multi-object tracking. IEEE Trans. Pattern Anal. Mach. Intell. **40**(3), 595–610 (2018). https://doi.org/10.1109/TPAMI.2017.2691769

4. Berclaz, J., Fleuret, F., Türetken, E., Fua, P.: Multiple object tracking using k-shortest paths optimization. IEEE TPAMI **33**(9), 1806–1819 (2011). https://doi.org/10.1109/TPAMI.2011.21
5. Bernardin, K., Stiefelhagen, R.: Evaluating multiple object tracking performance: the CLEAR MOT metrics. Eurasip J. Image Video Process. **2008** (2008). https://doi.org/10.1155/2008/246309
6. Bochinski, E., Eiselein, V., Sikora, T.: High-Speed tracking-by-detection without using image information. In: 2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance, AVSS 2017 (2017). https://doi.org/10.1109/AVSS.2017.8078516
7. Butt, A.A., Collins, R.T.: Multi-target tracking by Lagrangian relaxation to min-cost network flow. In: Proceedings of CVPR, pp. 1846–1853 (2013)
8. Chari, V., Lacoste-Julien, S., Laptev, I., Sivic, J.: On pairwise costs for network flow multi-object tracking. In: Proceedings of CVPR, 07–12 June, pp. 5537–5545 (2015). https://doi.org/10.1109/CVPR.2015.7299193
9. Dai, J., Li, Y., He, K., Sun, J.: R-FCN: object detection via region-based fully convolutional networks. In: NIPS 2016 Proceedings of the 30th International Conference on Neural Information Processing Systems, pp. 379–387 (2016). https://academic.microsoft.com/paper/2407521645
10. Dehghan, A., Modiri Assari, S., Shah, M.: GMMCP tracker: globally optimal generalized maximum multi clique problem for multiple object tracking. In: Proceedings of CVPR, pp. 4091–4099 (2015)
11. Dicle, C., Camps, O.I., Sznaier, M.: The way they move: Tracking multiple targets with similar appearance. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2304–2311 (2013). https://doi.org/10.1109/ICCV.2013.286
12. Dollar, P., Appel, R., Belongie, S., Perona, P.: Fast feature pyramids for object detection. IEEE Trans. Pattern Anal. Mach. Intell. (2014). https://doi.org/10.1109/TPAMI.2014.2300479
13. Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., Koltun, V.: CARLA: an Open Urban Driving Simulator. In: Proceedings of the 1st Annual Conference on Robot Learning, pp. 1–16 (2017). http://arxiv.org/abs/1711.03938
14. Emami, P., Pardalos, P.M., Elefteriadou, L., Ranka, S.: Machine learning methods for solving assignment problems in multi-target tracking. arXiv:1802.06897 **1**(1), 1–35 (2018)
15. Feichtenhofer, C., Pinz, A., Zisserman, A.: Detect to track and track to detect. In: Proceedings of the IEEE International Conference on Computer Vision 2017-October, pp. 3057–3065 (2017). https://doi.org/10.1109/ICCV.2017.330
16. Felzenszwalb, P.F., Girshick, R.B., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part-based models. IEEE TPAMI **32**(9), 1627–1645 (2010)
17. Ferryman, J., Shahrokni, A.: PETS2009: Dataset and challenge. In: Proceedings of the 12th IEEE International Workshop on Performance Evaluation of Tracking and Surveillance, PETS-Winter 2009 (2009). https://doi.org/10.1109/PETS-WINTER.2009.5399556
18. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the KITTI vision benchmark suite. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 3354–3361 (2012). https://doi.org/10.1109/CVPR.2012.6248074

19. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (2014). https://doi.org/10.1109/CVPR.2014.81
20. Hara, K., Kataoka, H., Satoh, Y.: Can Spatiotemporal 3D CNNs Retrace the History of 2D CNNs and ImageNet? In: CVPR, pp. 6546–6555 (2018). https://doi.org/10.1109/CVPR.2018.00685, http://arxiv.org/abs/1711.09577
21. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (2016). https://doi.org/10.1109/CVPR.2016.90
22. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. arXiv (2015)
23. Iqbal, U., Milan, A., Gall, J.: PoseTrack: joint multi-person pose estimation and tracking. In: Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017 (2017). https://doi.org/10.1109/CVPR.2017.495
24. Kingma, D.P., Ba, J.L.: Adam: a Method for Stochastic Optimization. In: ICLR 2015 : International Conference on Learning Representations 2015 (2015). https://academic.microsoft.com/paper/2964121744
25. Leal-Taixé, L., Milan, A., Reid, I., Roth, S., Schindler, K.: MOTChallenge 2015: towards a benchmark for multi-target tracking. arXiv:1504.01942 [cs] pp. 1–15 (2015). http://arxiv.org/abs/1504.01942
26. Liu, W., et al.: SSD: single shot multibox detector. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9905, pp. 21–37. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46448-0_2
27. Luo, W., et al.: Multiple object tracking: a literature review. arXiv:1409.7618v4, pp. 1–18 (2017). https://doi.org/10.1145/0000000.0000000
28. Milan, A., Leal-Taixe, L., Reid, I., Roth, S., Schindler, K.: MOT16: a benchmark for multi-object tracking. CoRR abs/1603.0 (2016). http://arxiv.org/abs/1603.00831
29. Milan, A., Roth, S., Schindler, K.: Continuous energy minimization for multitarget tracking. IEEE Trans. Pattern Anal. Mach. Intell. **36**(1), 58–72 (2014). https://doi.org/10.1109/TPAMI.2013.103
30. Munkres, J.: Algorithms for the assignment and transportation problems. J. Soc. Ind. Appl. Math. **5**(1), 32–38 (1957). https://doi.org/10.1137/0105003
31. Nair, V., Hinton, G.: Rectified linear units improve restricted Boltzmann machines. In: Proceedings of the 27th International Conference on Machine Learning (2010)
32. Paszke, A., et al.: Automatic differentiation in PyTorch. Adv. Neural Inf. Process. Syst. **30**(Nips), 1–4 (2017)
33. Pirsiavash, H., Ramanan, D., Fowlkes, C.C.: Globally-optimal greedy algorithms for tracking a variable number of objects. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 1201–1208 (2011). https://doi.org/10.1109/CVPR.2011.5995604
34. Reid, D., et al.: An algorithm for tracking multiple targets. IEEE Trans. Autom. Control **24**(6), 843–854 (1979)
35. Ristani, E., Solera, F., Zou, R., Cucchiara, R., Tomasi, C.: Performance measures and a data set for multi-target, multi-camera tracking. In: Hua, G., Jégou, H. (eds.) ECCV 2016. LNCS, vol. 9914, pp. 17–35. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-48881-3_2
36. Ristani, E., Tomasi, C.: Tracking multiple people online and in real time. In: Cremers, D., Reid, I., Saito, H., Yang, M.-H. (eds.) ACCV 2014. LNCS, vol. 9007, pp. 444–459. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-16814-2_29

37. Roshan Zamir, A., Dehghan, A., Shah, M.: GMCP-tracker: global multi-object tracking using generalized minimum clique graphs. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012. LNCS, vol. 7573, pp. 343–356. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33709-3_25
38. Shafique, K., Shah, M.: A noniterative greedy algorithm for multiframe point correspondence. IEEE Trans. Pattern Anal. Mach. Intell. **27**(1), 51–65 (2005)
39. Sheng, H., Zhang, Y., Chen, J., Xiong, Z., Zhang, J.: Heterogeneous association graph fusion for target association in multiple object tracking. IEEE Trans. Circ. Syst. Video Technol. **29**, 3269–3280 (2018)
40. Shitrit, H.B., Berclaz, J., Fleuret, F., Fua, P.: Multi-commodity network flow for tracking multiple people. IEEE TPAMI **36**(8), 1614–1627 (2014)
41. Shu, G., Dehghan, A., Oreifej, O., Hand, E., Shah, M.: Part-based multiple-person tracking with partial occlusion handling. In: Proceedings of CVPR, pp. 1815–1821. IEEE (2012)
42. Sun, S., Akhtar, N., Song, H., Mian, A., Shah, M.: Deep affinity network for multiple object tracking **13**(9), 1–15 (2018). http://arxiv.org/abs/1810.11780
43. Tian, Y., Dehghan, A., Shah, M.: On detection, data association and segmentation for multi-target tracking. IEEE Trans. Pattern Anal. Mach. Intell. **41**, 2146–2160 (2018)
44. Voigtlaender, P., et al.: Mots: multi-object tracking and segmentation. In: Proceedings of CVPR, pp. 7942–7951 (2019)
45. Wen, L., et al.: UA-DETRAC: a new benchmark and protocol for multi-object detection and tracking (2015). http://arxiv.org/abs/1511.04136
46. Wen, L., Du, D., Li, S., Bian, X., Lyu, S.: Learning non-uniform hypergraph for multi-object tracking. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 8981–8988 (2019)
47. Wen, L., Li, W., Yan, J., Lei, Z., Yi, D., Li, S.Z.: Multiple target tracking based on undirected hierarchical relation hypergraph. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, pp. 1282–1289 (2014). https://doi.org/10.1109/CVPR.2014.167
48. Wu, B., Nevatia, R.: Detection and tracking of multiple, partially occluded humans by Bayesian combination of edgelet based part detectors. Int. J. Comput. Vis. **75**(2), 247–266 (2007)
49. Zhu, J., Yang, H., Liu, N., Kim, M.: Online Multi-Object Tracking with Dual Matching Attention Networks, pp. 1–17 (2019)