Tracking Multiple Deformable Objects in Egocentric Videos

Mingzhen Huang[1,2†], Xiaoxing Li[2], Jun Hu[2], Honghong Peng[2], Siwei Lyu[1]
[1]State University of New York at Buffalo, [2]Meta Reality Labs

Project Page: https://mingzhenhuang.com/projects/detracker.html

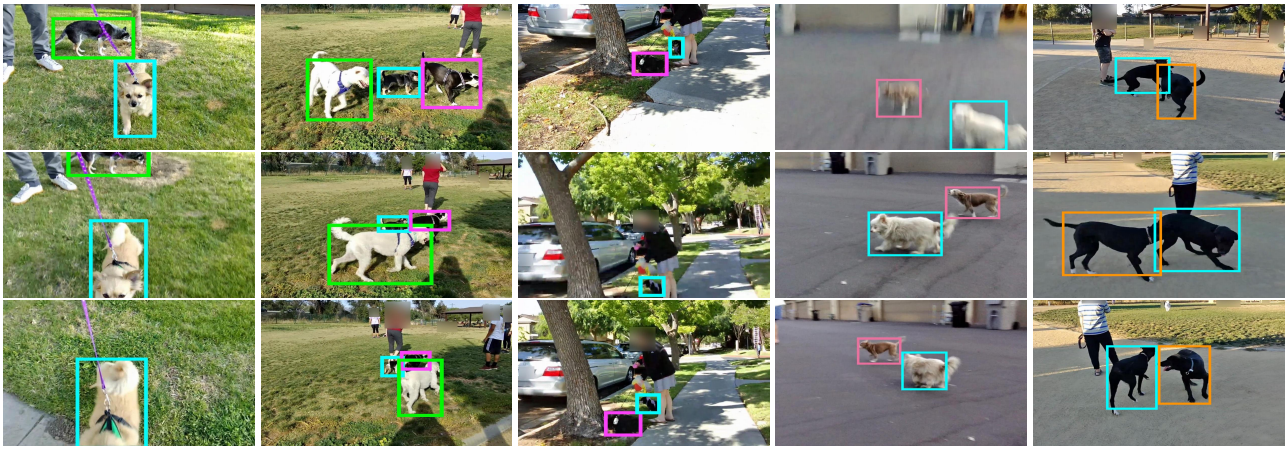


Figure 1. Example frames from the DogThruGlasses dataset. The videos were captured using the smart glasses.

Abstract

Most existing multiple object tracking (MOT) methods that solely rely on appearance features struggle in tracking highly deformable objects. Other MOT methods that use motion clues to associate identities across frames have difficulty handling egocentric videos effectively or efficiently. In this work, we present DogThruGlasses, a large-scale deformable multi-object tracking dataset, with 150 videos and 73K annotated frames, which is collected exclusively by smart glasses. We also propose DETracker, a new MOT method that jointly detects and tracks deformable objects in egocentric videos. DETracker uses three novel modules, namely the motion disentanglement network (MDN), the patch association network (PAN) and the patch memory network (PMN), to explicitly tackle severe ego motion and track fast morphing target objects. DETracker is end-to-end trainable and achieves near real-time speed, which outperforms existing state-of-the-art method on DogThruGlasses and YouTube-Hand.

1. Introduction

Wearable cameras have become an emerging trend, promoted by the rapidly growing collection of consumer products such as smart glasses. As the wearable cameras became more powerful with increased battery capacity, sensor size, on-board memory volume, and sophisticated in-device processors, there is an increasing demand for real-time scene understanding to run reliably and yet efficiently on-device. The underlying computer vision algorithms, on the other hand, frequently starts from the detection and tracking of objects within the scene. For the egocentric videos captured from wearable cameras, besides being challenged by occlusion, morphing shapes, and multiple visually resembling objects, the multiple object tracking (MOT) algorithms are stressed by the constantly changing egocentric viewpoint.

Unique to wearable cameras, the large ego motion caused by the head movements of the wearer is often drastic, unpredictable, and largely uncorrelated to the object motions. The reduced predictability of motion patterns forces traditional MOT algorithms [45] to search in larger regions to maintain same performance level, which in turn compromises the running speed and makes them less suitable for on-device execution. In the meanwhile, the ego motion may also exacerbate the deformation and occlusion of objects, by imposing lens distortion, blurriness and rolling shutter, especially those in the near field. The ego motion also aggravates object occlusion due to the constantly changing of points of view. These side effects further downgrade the performance of appearance-based MOT

[†]Work done during Mingzhen's internship with Meta.

approaches [16, 42, 53].

In this work, we propose an efficient end-to-end trainable method, **DETracker** (**D**eformable **E**gocentric **Tracker**), for tracking multiple deformable objects in egocentric videos. DETracker has three major components, the motion disentanglement network (MDN), the patch association network (PAN), and the patch memory network (PMN). MDN is to estimate the motion flow between two consecutive frames efficiently. It explicitly separates a global camera motion before estimating the local object motion and thus is robust under severe ego motion. PAN tackles deformable object tracking by dividing objects into patches and localizing individual patches by finding their best matched patches in upcoming frames. PMN retains and updates feature embeddings of tracked objects within a prolonged time window by leveraging a transformer network [40] and thus is able to use historical patch features for long-term association.

Although there exist several large-scale MOT datasets [7, 9, 14, 22], they are limited to either fixed camera views [7] or simple ego motion, e.g., from car-mount cameras [9, 52]. To build a large-scale, egocentric MOT dataset, we collected **DogThruGlasses**, a video set of dogs captured with the smart glasses. This dataset represents the complexity of real-life object tracking scenarios from wearable devices (see examples in Fig. 1). In DogThruGlasses, we release 150 videos with 73K annotated frames, 157K annotated bounding boxes, and 474 unique identities/trajectories. To our knowledge, this is the first large-scale dataset for tracking multiple objects in egocentric videos. It would serve as a challenging benchmark for existing and future MOT methods. The dataset and the code will be released for research purposes.

We summarize our major contributions as follows:

- We present DogThruGlasses, the first large-scale egocentric MOT dataset collected with smart glasses, offering extensive coverage of object deformation, ego motion, and diverse scenes.
- We propose DETracker, a new MOT algorithm that is designed to be robust under severe ego motion and fast object deformation.
- In DETracker, we use MDN to disentangle camera motion from object motion and predict a high-accuracy object trajectory very efficiently. In addition, we design PAN and PMN to help with the detection and long-term tracking of objects under large deformations and heavy occlusions.
- Experimental results in Table 2 show that our proposed method outperforms the state-of-the-art method by 8.1% on DogThruGlasses. It also achieves competitive results on YouTube-Hand [14] for hand tracking.

2. Related Works

Traditional MOT approaches mostly follow a tracking-by-detection scheme [5, 28, 31, 37, 42, 43, 48, 54]: an object detector is employed to detect objects, and then a specific association method is used to connect individual detections into continuous trajectories. Hungarian algorithm [23] is a popular method for associating, where the affinity cost is defined based on Intersection over Union (IoU) of two bounding boxes. Bewley et al. [4] proposed to use the Hungarian algorithm for associating detected bounding boxes with predicted tracklet movement generated by Kalman Filter [15]. ByteTrack [54] further improves the tracking quality by recovering low confident detection results with a two-stage association. However, the rapid camera motion leads to large offsets, which fails the traditional spatial location-based methods. Appearance-based Re-Identification (ReID) module is another widely adopted association method in MOT algorithms [11, 39, 42]. Those methods are robust to objects and ego motion since the objects' spatial location change is not used as a key for the association. Wojke and Bewley [42] proposed to extract ResNet [12] features for detected objects and then associate them based on their feature cosine similarities. A common drawback for those methods is that the detection and association modules are separately optimized, even though they are trying to describe the same object. As a result, the decoupled outcomes from two modules cannot benefit each other, and often yield sub-optimal final results.

Recently, the methods that jointly detect and track objects [20, 27, 34, 41, 45, 53, 57] became mainstream in MOT. The detection and association networks are usually trained end-to-end to avoid falling into local optimums. Zhan et al. [53] proposes to leverage a Re-ID branch for association jointly train it along with the detection network. FairMOT [53] achieves competitive results on MOT benchmarks [7, 22]. However, it is hard to handle highly deformable objects with frequently changing appearance, similar to appearance-based tracking-by-detection methods. Many other joint detection and tracking MOT approaches [10, 14, 31, 32, 35, 36, 45, 50, 57] achieve competitive performance by leveraging a motion estimation module for data association. However, such methods usually limit the object motion search within a small spatial window. This setting can easily fail under dramatic camera motion. Zhou et al. [57] followed [56] to detect objects' center points as a heatmap and proposed to compute the object center motion from concatenated features from two consecutive frames. Similarly, such methods estimate the motion offset within a local window, and cannot tolerate rapid camera motion. Inspired by RAFT [38], Wu et al. [45] extends [57] by estimating the object's movement offset from the global cost volume for each pair of pixels. This operation results in an affinity matrix $\mathcal{M} \in \mathbb{R}^{hw \times hw}$, where h and w

are the height and width of the input feature map. Different from [57], the estimated motion offset is not constrained by a small local neighborhood, but computing such an affinity matrix is extremely expensive and is not achievable on mobile devices.

Memory Network has been widely explored in video analysis tasks [19, 26, 44], and tracking objects with memories is also widely adopted in recent MOT methods [6, 8, 49]. By maintaining and optimizing an appearance bank, objects can be tracked in long term effects, with only a minimal footprint. Alone this line, Cai et al. [6] use their proposed Memory Aggregator Network to keep the appearance feature embedding of tracked objects in memory and use these memory embeddings to query current objects for detecting and tracking.

3. DogThruGlasses Dataset

It is notable that, dramatic camera motion and fast changing object appearance caused by occlusion and deformation are commonly seen from media captured with wearable devices. Yet, their combined impact on object tracking methods is largely unexplored in past literature. Targeting this absence, we present DogThruGlasses, a large-scale MOT dataset collected with wearable devices.

In this dataset, we carefully choose dogs as targeting objects for multiple reasons. First of all, as companion animals, dogs are among the most frequently imaged targets in consumer videos. Dog videos are also shared through social media in vast amounts on a daily basis. Also, dogs of different breeds vary dramatically in size, color, shape, and habit. They are also frequently in motion and demonstrate rapid deformation. All these factors post-add-on challenge the ego motion due to the wearers' head movements.

Data source. Multiple individuals participate the collection of videos in DogThruGlasses via smart glasses, refer to Fig. 1 for sample frames. The videos are captured in 30 frames per second (FPS) using the device domestic camera application, and are then resized to 1000×1000 pixel resolution. DogThruGlasses covers scene diversity by including beaches, dog parks, roadsides, backyards, parking lots, restaurants, etc. The videos are captured at different time point of the day, as well as under different weather condition. The dataset also aggressively covers object diversity by including up to 33 dog breeds. To name a few, Labrador Retriever, Golden Retriever, German Shepherd, Poodle, American Bulldog, Rottweile, Australian Shepherd, Beagle, etc.

The annotation task is carried out by two individual annotators, who are unfamiliar with any tracking algorithms. They are asked to annotate tight boxes around visible part of individual dogs, without hallucination about the occluded parts. During the annotation process, annotators noticed

Dataset	Unpred. EgoMotion	Deform. Objects	Total #Videos	Total #Frames	Anno. Boxes	Total #Trajs	Trajs Length
MOT17 [22]	◇	-	14	11K	215K	1342	110
MOT20 [7]	-	-	8	13K	1.6M	3457	453
KITTI [9]	-	-	50	13K	47K	917	51.5
YouTube-Hand [14]	◇	✓	240	20K	60K	864	70.5
VIVA [29]	×	✓	20	6K	13K	45	64.5
DogThruGlasses	✓	✓	150	73K	156K	474	325

Table 1. **Comparison among MOT datasets.** The ◇ denotes part of the dataset contain unpredictable ego motion. The length of trajectories we reported here is the median number of all trajectories length.

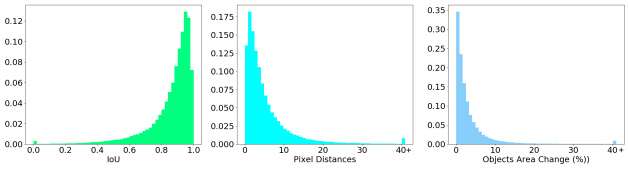


Figure 2. **Analysis of DogThruGlasses.**

that some of the dog breeds are in particular challenging to annotate due to very small size, all white/all black appearance, or highly similar patterns. These factors lead to inconsistent annotations. So we enforce an additional round of cross verification and correction between the two annotators to guarantee the quality.

Data statistics. We show the statistics of DogThruGlasses compared to other MOT datasets in Table 1. DogThruGlasses provides a complex combination of ego motion (compared to fixed or in-vehicle-mounted cameras) and objects deformation. Meanwhile, it has a larger volume in terms of both the number of videos and the number of annotated images. Content-wise, DogThruGlasses offers a large number of objects and longer trajectories. All these factors render DogThruGlasses a unique and challenging dataset for MOT benchmarking.

We further split the dataset into training and testing set. Visual inspection was conducted to guarantee that enough scene and object diversity is covered by both set. As a result, we included 120 videos/376 trajectories in the training set, with about 60K frames and 125K annotated boxes. The testing set contains 30 videos/98 trajectories, with about 13K frames and 31K annotated boxes.

Data analysis. Compared to existing MOT datasets, DogThruGlasses aggressively covers ego motion and object deformation. To qualitatively show data coverage, we compare pairs of bounding boxes form adjacent frames in the ground truth annotation. Refer to Fig. 2 for a detailed break down of the statistics, where we see cases that, per frame translation(distance between box centers) goes up to 300 pixels, and IoU drops to as low as 0. These numbers reflect the general challenge of object tracking on egocentric videos.

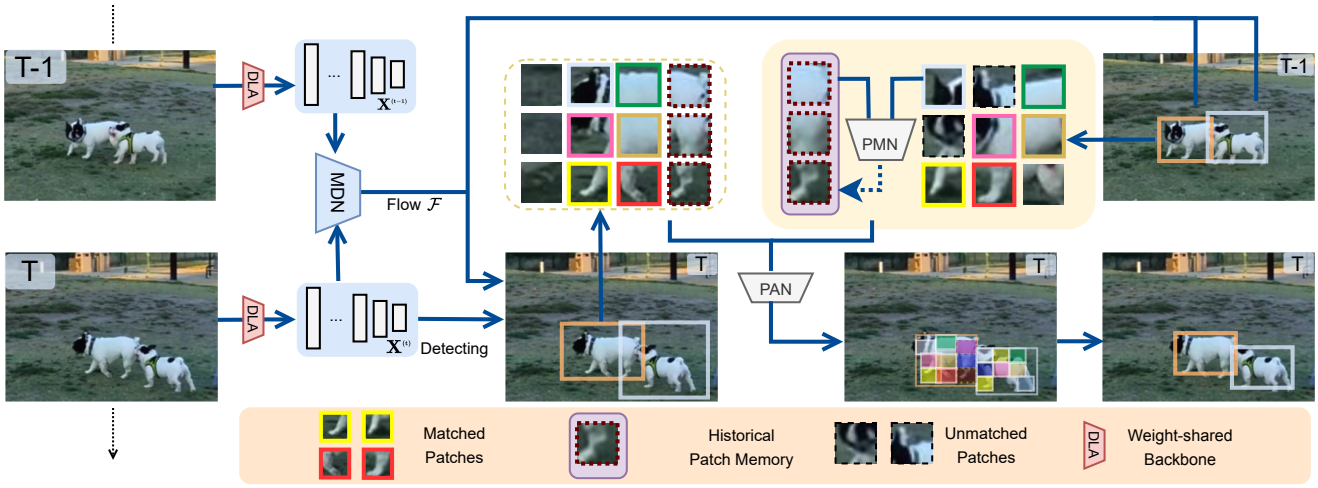


Figure 3. **The overall processing pipeline of DETracker.** This figure is best viewed in color.

4. Method

The overall architecture of DETracker is illustrated in Fig. 3. Let $\mathbf{I}^{(t)}$ be the frame at time t, we first obtain its feature map pyramid $\{\mathbf{X}_l^{(t)}\}(l \in [0, L)$ is the pyramid index) from a DLA-34 backbone [51]. Along with $\{\mathbf{X}_l^{(t-1)}\}$ from frame `t-1`, we estimate the pixel-wise motion flow $\mathcal{F}^{(t)}$ with motion disentanglement network, MDN. Next, with a collection of tracked objects on frame `t-1` and the motion flows $\mathcal{F}^{(t)}$, we propagate the bounding boxes into frame `t`. Then the patch association network, PAN, as described in Sec. 4.2, detects and associates objects by matching sub-divided patches between the two frames. Different from most prior MOT methods, the detection and association are done by a single network to ensure both localization and identity association accuracy. Finally, the patch memory network, PMN, retains unmatched patches from PAN output in a fixed-length ring patch memory buffer $\mathbf{P}^{mem}$ to help track deformable and occluded objects. Buffered and novel patches from newly seen frames jointly participate in patch-matching in the next time step.

4.1. Motion disentanglement network

The inputs to MDN are two pyramid feature maps $\{\mathbf{X}_l^{(t)}\}$ and $\{\mathbf{X}_l^{(t-1)}\}$. We follow [33] to estimate the optical flow $\mathcal{F}^{(t)}$ between frames according to simple and well-established principles: pyramid processing, warping, and the use of a cost volume. To handle large camera motion, which is common for egocentric videos, we expand the cost volume search range to the whole frame pixels like RAFT [38].

Let $\mathcal{F}_l^{(t)}$ represents the estimated optical flow at the pyramid level l. For level $l = 0$, we use the features $\mathbf{X}_0^{(t-1)}$ and $\mathbf{X}_0^{(t)}$ to construct a cost volume that costs for associating each pixel with **all** pixels at the next frame as follow,

$$\mathbf{CV}_0(\mathbf{x_1}, \mathbf{x_2}) = (\mathbf{X}_0^{(t-1)}(\mathbf{x_1}))^\top \mathbf{X}_0^{(t)}(\mathbf{x_2}) \quad (1)$$

where $\top$ is the transpose operator and $\mathbf{X}_0^{(t-1)}(\mathbf{x_1})$ is the 1D feature vector extracted from $\mathbf{X}_0^{(t-1)}$ at pixel $\mathbf{x_1}$. The cost volume $\mathbf{CV}_0$ feeds to the optical flow estimator and the context network [33] and produce $\mathcal{F}_0^{(t)}$ which is mainly responsible for the large camera motion and it is self-supervised as described in Sec. 4.4.

For the lth level($l \geq 0$), its cost volume $\mathbf{CV}_l$ construction is different since global search over the whole image is impractical, so we construct a cost volume that costs for associating each pixel with only the neighboring pixels at the next frame as follow,

$$\mathbf{CV}_l(\mathbf{x_1}, \mathbf{x_2}) = (\tilde{\mathbf{X}}_l^{(t-1)}(\mathbf{x_1}))^\top \mathbf{X}_l^{(t)}(\mathbf{x_2}) \quad (2)$$
$$\text{where } |\mathbf{x_1} - \mathbf{x_2}|_\infty \leq r$$

$\tilde{\mathbf{X}}_l^{(t-1)}$ is the warped version of $\mathbf{X}_l^{(t-1)}$ under the upsampled flow $\mathcal{F}_{l-1}^{(t)}$. Finally, similar to the level 0, $\mathbf{CV}_l$ feeds to the optical flow estimator and the context network, and produce $\mathcal{F}_l^{(t)}$. This is a recursive process, that goes all the way down, till the layer $L-1$ and produces full resolution flow $\mathcal{F}^{(t)} = \mathcal{F}_{L-1}^{(t)}$.

4.2. Deformable objects as patches

Given feature maps $\{\mathbf{X}_l^{(t-1)}\}$ and $\{\mathbf{X}_l^{(t)}\}$, an estimated flow $\mathcal{F}$, and the bounding box of the tracked object $\mathcal{B}^{(t-1)}$ in $\mathbf{I}^{(t-1)}$, we now extend the trajectory into $\mathbf{I}^{(t)}$ by simultaneously detecting and tracking.

To start with, we use the ROI Align method [13] to obtain features $\mathbf{K}^{(t-1)} \in \mathbb{R}^{n \times n \times d}$ for each tracked object in $\mathbf{I}^{(t-1)}$, where n is a hyper-parameter defining the feature resolution, and d is the number of channels of the ROI feature embedding. Note that, by doing so, the obtained features not only have translation invariant, but also has unified

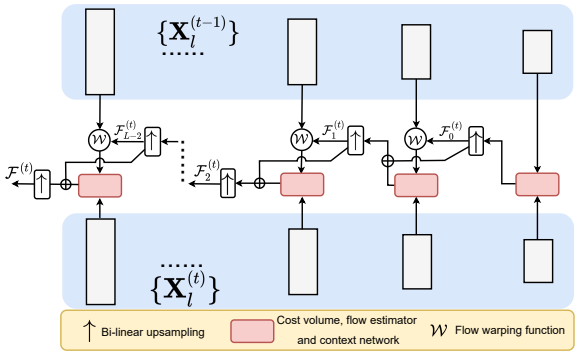


Figure 4. **Illustration of MDN.**

feature size. In other words, this operation can be viewed as dividing the ROI into $n \times n$ super-pixels, where each super-pixel is a small image patch $\mathbf{P}_i^{(t-1)}, i \in [0, n^2)$. Further, each patch is converted to a feature embedding of d-dimension, which is denoted as $\kappa_i \in \mathbb{R}^{1\times d}$ and i is the patch index.

The key steps for the proposed tracking algorithm are illustrated in Fig. 5. To locate the position of $\mathcal{B}^{(t-1)}$ on $\mathbf{I}^{(t)}$ and obtain $\mathcal{B}^{(t)}$, we establish a potential searching region by first propagating the center point of $\mathcal{B}^{(t-1)}$ with flow $\mathcal{F}$; then, we enlarge the region with a ratio $\alpha > 1.0$ to produce an expanded region that is large enough to capture the intended object, even in case of zoom-in or stretched shape. This expanded region is drawn as the dash-lined blue box in Fig. 5. At this point, we can follow the same routine to sub-divide the expanded region into $\alpha n \times \alpha n$ super pixels, which yield features $\mathbf{K}^{(t)} \in \mathbb{R}^{\alpha n\times \alpha n\times d}$. Now we are ready for both detection and association.

Patch association network. We detect and track an object by first associating each individual query patch in frame `t-1` and patch memory with candidate patches in frame `t`. An intuitive way for patch-wise association is to compute feature cosine similarity and solve patch-matching as a bipartite graph matching problem. However, such simple method treats each individual patch independently, and all cross-patch relationships are disregarded. In a prior work [58], the authors leverage a transformer encoder-decoder network [40], to reason about the relations of detected objects and trajectories for object association. In a similar spirit, we propose PAN (see Fig. 6) to reason the relations of all patches that belong to the same object and solve the patch-wise association problem. For each object, we compute a set of association scores $\mathcal{S}_i$ between each query patch and all candidate patches using standard transformer encoder-decoder network $g(\cdot,\cdot)$ [40], and obtain $\mathcal{S}_i = g(\kappa_i^{(t-1)}, \mathbf{K}^{(t)}) \in \mathbb{R}^{1\times \alpha^2 n^2}$. We further use softmax to normalize each set of association scores $\mathcal{S}_i$ between a query patch from frame `t-1` and all the candidate patches in frame `t`.

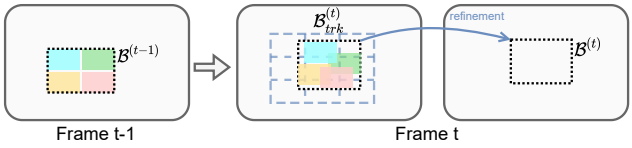


Figure 5. **Patch propagation and association for detection and tracking.**

Patch-based sub-grid localization. After obtaining the association scores for all patches, a naive way is to select the patch with the highest score. However, this will constrain the matching to the predefined sparse patch grids. Increasing the grid density allows a finer resolution; however, this will increase the computation cost and lead to smaller patch size, thus lowering quality association scores. To minimize the computation cost and maintain high-quality association, we propose using a simple yet effective sub-grid search. At this point, all query patches that have the largest association score smaller than threshold θ are declared as unmatched since they might be either heavily occluded or going out of view. Remaining patches are tagged as matched, and each matched patch has scores S_i represent the likelihood that patch $\mathbf{P}_i^{(t-1)}$ corresponds to each of the target patches in frame `t`. We then localize the matched patch in frame `t` by the weighted sum of all possible grid locations, where the normalized association scores are used as weights. Refer to Fig. 5 for the localization of color-matched individual patches.

Deformable object tracking with confidence. With all the successfully localized patches, we take their minimum bounding rectangle $\mathcal{B}_{trk}^{(t)}$ as the initialization of tracked object location. As a class-specific proposal, $\mathcal{B}_{trk}^{(t)}$ can be further refined based on the objectness, as in Faster-RCNN [30]. We use such a regression branch for refinement and get $\mathcal{B}^{(t)}$, where a regression confidence $\mathcal{P}(\mathcal{B}^{(t)}|\mathcal{B}_{trk}^{(t)})$ is obtained in addition. Unlike the typical MOT methods, whose confidence values solely indicate the objectness from the detected regions, we aim to obtain confidence scores that combine both detection and identity association confidences. Thus, they can be used to monitor the healthiness of active tracks and help to suppress false positives.

In particular, we define a tracklet T of length t as the sequence of detected boxes in the past frames: $T^t = \{B^{(0)}, B^{(1)}...B^{(t)}\}$. Using chain rule, it's confidence can be computed recursively: $\mathcal{P}(T^t) = \mathcal{P}(T^t|T^{t-1})\mathcal{P}(T^{t-1})$, where $\mathcal{P}(T^t|T^{t-1})$ has two components, one is the regression confidence obtained during box refinement, and the other is a ROI-wise association score $\overline{\mathcal{S}^t}$, obtained by averaging the normalized association scores S_i for all matched patches inside the object. Putting these together, we get:

$$\mathcal{P}(T^t) = \mathcal{P}(\mathcal{B}^{(t)}|\mathcal{B}_{trk}^{(t)})\mathcal{P}(T^{t-1})\overline{\mathcal{S}^t} \tag{3}$$

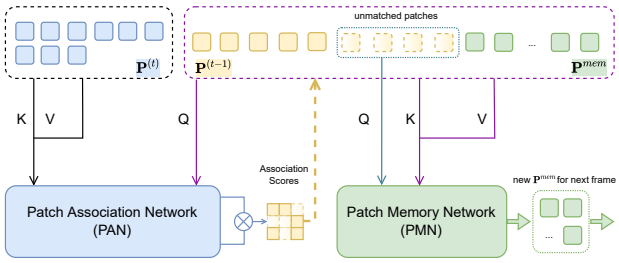


Figure 6. **Patch association and patch memory.** The Q, K and V are the query, key and value in the transformer networks [40] and $\otimes$ is the matrix multiplication.

Naturally, confidence defined in this way drops for long tracklets. Thus we concatenate old tracklets with new born ones when necessary, to obtain complete tracks.

4.3. Patch Association and Memory

Temporal memory has been previously explored in MOT algorithms [6] to deal with occlusion, periodical pose change, etc. To adopt this scheme for deformable object tracking, we proposed PMN. This network gathers a collection of representative yet unique patches $\mathbf{P}^{mem}$ to describe the long-term appearance of each active tracklet. $\mathbf{P}^{mem}$ participate during patch association, in addition to the concurrent patches on frame `t-1`. Refer to Fig. 6.

Recall that post PAN, we obtain a bucket of matched patches (colored with solid yellow in Fig. 6) and a bucket of unmatched ones (colored with lighter yellow in Fig. 6). Potentially, the unmatched patches may carry a novel appearance introduced by deformation or occlusion. Thus, we use these unmatched patches as queries in PMN, to explore the similarity between unmatched patches and all other patches, including the $\mathbf{P}^{mem}$ inherited from the previous frame. We obtain the updated unmatched patch features from PMN outputs and roll those patches into the collection of $\mathbf{P}^{mem}$. In the meanwhile, all the matched patches in $\mathbf{P}^{mem}$ get removed. The $\mathbf{P}^{mem}$ is managed in a FIFO manner, where old patches will be kicked out automatically subject to a fixed time window β.

4.4. Training and Inference

Online inference. For a new frame, DETracker tracks all active objects $\{\mathcal{B}^{(t-1)}\}$ from the previous frame. After the tracking step, all objects $\{\mathcal{B}^{(t)}\}$ with a confidence score larger than ξ are marked as tracked, and all remaining ones are marked as mis-tracked. The mis-tracked objects are not discarded immediately, instead, they persist in memory for another β frames, for potential resumable tracking after brief occlusion. In addition, we still rely on an object detector [56] to promptly initialize newborn objects once appeared. We define an object as a "newborn" and initialize a trajectory for it, as long as its detection confidence score is larger than ϕ and the IoUs are smaller than γ with any existing object.

Training MDN. During training, the inputs to the network are either two frames randomly sampled from same video or with randomly shifted and rotated static images. Camera flow estimator can be trained in a supervised fashion with static images, where we convert the random transforms into dense flow vector fields. Thus the camera loss $\mathcal{L}_{cf}$ is formulated as a standard endpoint error between the predicted camera flow and the generated dense camera flow. We set $\mathcal{L}_{cf} = 0$ when the inputs are video image sequence as the ground truth camera flow is not available in video sequence.

Unfortunately, the ground truth dense motion flow annotation is extremely difficult to obtain for real-world video captures, which is thus not provided alone with DogThruGlasses. As an alternative, we propose to generate a sparse pseudo-flow to enable the training. Given an annotated object that appears on two distinct frames, we compute the displacement of the bounding box central points $p^{(t)}$ and $p^{(t-1)}$ and spread it into a small neighborhood of radius r:

$$\mathcal{F}_q^{gt} = p^{(t)} - p^{(t-1)}, q \in N_r(p) \tag{4}$$

The overall motion flow loss is then set to the endpoint error, accumulated across layers l in the feature pyramid. Note that $\mathcal{L}_f$ is only calculated at the pixels that have a generated ground truth flow.

$$\mathcal{L}_f = \sum_l \|\mathcal{F}_l - \mathcal{F}^{gt}\|_2 \tag{5}$$

Training PAN. PAN is supervised by a patch association loss L_{pm}:

$$\mathcal{L}_{pm} = -\sum_i^{n^2} \sum_j^{\alpha^2 n^2} \sigma(\mathbf{P}_j^{(t)}) log(\mathcal{S}_i(j)) \tag{6}$$

where i is the index of query patches, and $\mathcal{S}_i(j)$ represents the matching score between the query patch and the jth candidate patch. $\sigma(\mathbf{P}_j^{(t)}) = 1$ if $\mathbf{P}_j^{(t)}$ has an overlapping with the ground truth bounding box. Otherwise σ is set to zero. α is the scale adaptive ratio described in Sec. 4.2.

Loss function. To train all module in our network, we use the following combined loss function: $\mathcal{L} = \mathcal{L}_f + \mathcal{L}_{cf} + \mathcal{L}_{pm} + \mathcal{L}_{det}$, where $\mathcal{L}_{det}$ is the standard detection loss introduced in [56].

5. Experimental Results

In this section, we first describe all the related implementation details. Then we compare our method with existing state-of-the-art multi-object tracking methods in Table 2 and report qualitative results in Fig. 7. At last, we demonstrate the effectiveness and efficiency of each proposed module with ablation studies.

Methods	IDF1↑	DetA ↑	AssA↑	FP(%) ↓	FN(%)↓	IDs(%)↓	MOTA↑	HOTA↑
SORT [43]	61.43	67.87	49.43	12.97	7.90	1.60	77.53	57.74
Tracktor [2]	60.47	63.61	46.01	2.40	21.83	1.45	74.32	53.97
FairMOT [53]	60.33	62.90	45.09	2.48	21.83	3.04	72.65	53.16
CenterTrack [57]	60.60	60.49	43.51	3.61	20.55	1.14	74.70	51.14
GTR [58]	66.88	69.59	52.63	6.73	10.19	0.86	82.22	60.38
ByteTrack w/ Re-ID [54]	71.93	64.33	57.75	4.95	16.41	0.61	78.02	60.75
BoT-SORT [1]	72.71	67.95	57.71	5.30	14.87	0.76	79.06	62.53
DETracker (Ours)	73.12	74.06	58.72	4.60	8.39	0.63	86.38	65.25

Table 2. **Main Results.** The best performance is highlight in blue and the second best performance in green.

5.1. Implementation details and evaluation metrics

Architecture Details. Our framework is implemented by PyTorch and Detectron2 [46]. Specifically, we build upon CenterNet [56] with a DLA-34 [51] for detection, and a Cascaded-ROI head for bounding box refinement.

Training and testing scheme. We train DETracker for 30K iterations using a SGD optimizer, with a learning rate of $2e{-}5$ and a batch size of 64. The network can be trained end-to-end and the inference speed is 24 FPS. Experiments are performed on NVIDIA A100 GPUs. In all our experiments, we set the hyper-parameters $\theta = 0.6, \xi = 0.4, \beta = 5, \gamma = 0.3$ and $\phi = 0.65$.

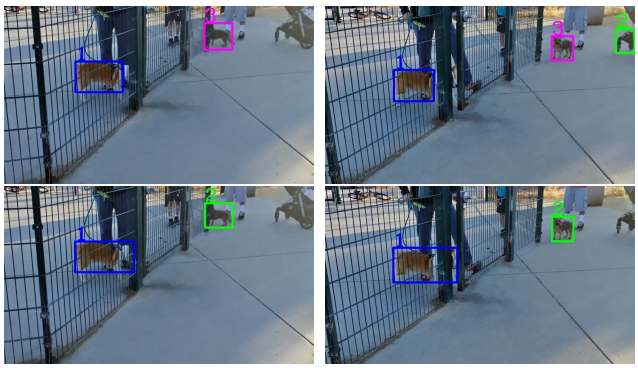
Evaluation metrics. We adopt the standard multiple-object-tracking evaluation metrics [3, 21, 22] to evaluate the performance, including: identification F1 score (IDF1), detection accuracy (DetA), association accuracy (AssA), false positives (FP), false negatives (FN), identity switches (IDs), multiple object tracking accuracy (**MOTA**) and higher order tracking accuracy (**HOTA**). Among these evaluation metrics, MOTA and HOTA evaluate overall detecting and tracking performance and are considered the most important metrics. For detection, we use standard mAP as the metric.

5.2. Main Results

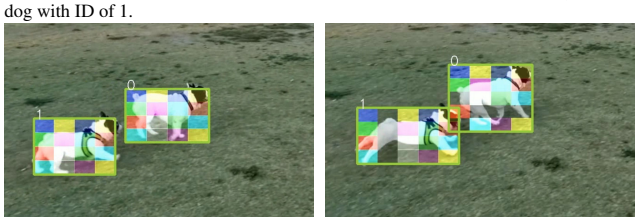
We compare our proposed DETracker with other state-of-the-art MOT approaches, the results are reported on DogThruGlasses test set (see Table 2), DETracker outperforms existing state-of-the-art algorithms on most metrics with a significant margin. For an apple-to-apple comparison, we adopted CenterNet [56] as detection backbone for all methods reported in Table 2. We pretrained CenterNet on COCO [18] dataset for object detection. To minimize the computation cost and memory footprint, and move toward mobile platform deployment, we set the input frame window length to at most 4 consecutive frames. Compared with the previous state-of-the-art, GTR [58], our DETracker outperforms it by 8.1% in terms of HOTA.

5.3. Ablation Studies

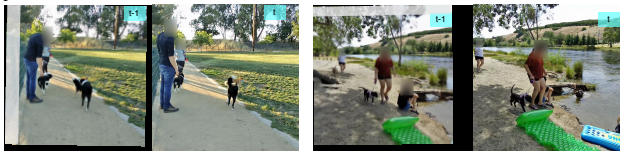
Effectiveness of proposed modules. We now demonstrate the effectiveness of each of our proposed modules in Table 3. We first show the baseline that removes all of our proposed three components: MDN, PAN and PMN. Secondly, we remove the flow estimation network and replace it with a direct propagation of object location into the next frame. Next, we remove PAN along with PMN and instead directly propagate prior tracked objects to current frame with estimated flow then refined with the regression branch. Finally, we switch on-and-off the PMN to show the benefit of storing long-term memory.



(a) Tracking results by DETracker (bottom) and GTR (top). Each row visualizes tracking results across two frames. Dogs that belong to the same trajectory are visualized with the same color and same ID. Benefit from our proposed PMN and PAN, DETracker has less FP tracklet and more accurate tracked bounding box for dog with ID of 1.



(b) Patch association results. The matched patches are visualized with the same color and the bounding boxes in green are the minimum bounding rectangle of the patches.



(c) DETracker produced ego motion compensation. The space in gray and black is the compensated ego motion.

Figure 7. **Qualitative results.**

Motion estimation. As we illustrated before, the global matching flow estimation methods [38, 47] are robust to ego motion but are not affordable in mobile device. We

Methods	IDF1 ↑	IDs(%) ↓	MOTA ↑	HOTA ↑
Baseline	60.47	1.45	74.32	53.97
w/o PAN&PMN	71.97	0.89	81.91	61.74
w/o MDN	70.14	1.21	81.76	62.99
w/o PMN	71.41	1.10	84.00	63.58
DETracker (ours)	**73.12**	**0.63**	**86.38**	**65.25**

Table 3. **Effectiveness of each proposed components.**

Methods	IDF1	IDs(%)	MOTA	HOTA	Time ↓
Global Matching Flow	**73.38**	0.68	86.28	65.13	151
Disentangled Flow	73.12	**0.63**	**86.38**	**65.25**	**14**

Table 4. **Comparison of different motion estimation network.** Note that time indicts the running time in ms for flow estimation network, experiments are performed on a NVIDIA V100 GPU.

Methods	IDF1 ↑	IDs(%) ↑	MOTA ↑	HOTA ↑
IoU	68.74	0.86	83.88	61.58
Cosine Similarity	68.81	1.18	83.31	62.34
PAN	**73.12**	**0.63**	**86.38**	**65.25**

Table 5. **Comparison of different association methods on DogThruGlasses dataset testing set.**

take sate-of-the-art global matching flow estimation method GMFlow [47] as an alternative for MDN. In GMFlow, the affinity of every pair of pixels has been computed to produce a global matching result for flow estimation, which costs much more computation resources than our MDN. Based on the results in Table 4, we conclude that with an outcome of comparable quality, MDN is far more efficient.

Patch association method. An intuitive way to associate patches is to compute the cosine similarity Re-ID [55] between each pair and select the one with the largest score. However, cosine similarities are computed on each pair independently, regardless of their temporal and spatial relationships. PAN, in turn, is able to capture these relationships, considering the candidate patches are subcomponents coming from one common object. Experimental results in Table 5 shows that the relation learned by PAN would greatly benefit our proposed DETracker.

Detection performance. Our patch-based detection and tracking algorithm not only boosts the tracking quality but also helps improve the bounding box detection performance by jointly utilizing temporal consistency. We evaluate our method on DogThruGlasses testing test with common detection metrics [17] and compare with other state-of-the-art joint detection and tracking algorithms in Table 6. In addition, we provide results after switching the key components on and off for a detailed insight into component-wise effectiveness.

5.4. Performance on Other Datasets

Hand Tracking. To test the robustness of our proposed method, we conducted a similar experiment on YouTube-Hand [14] dataset, which is a multiple-hand tracking benchmark. We choose this dataset because the appearance and shape of hands change drastically and frequently, which makes hands a highly deformable object. Even though YouTube-Hands is not designed for egocentric tracking which most videos are captured by fix cameras, some videos from the YouTube-Hand [14] are captured by body-worn or hand-held cameras; therefore, ego motion is partially included in this dataset. Following [14], we pretrain DETracker on TV-Hand [24] and COCO-Hand [24] datasets for hand detection and camera flow estimation. Then we fine-tune the model on YouTube-Hand [14] dataset for hand tracking. For a fair comparison, we consider hands as individual objects but not part of humans, so for all the methods reported here, we do not use the pose-based post-processing method proposed in [14]. Experimental results show that our proposed DETracker outperforms all other general MOT approaches and achieves competitive performance compared with HandLer [14]. However, HandLer [14] is specifically designed for hand tracking. DETracker, on the other hand, can be adapted to track any arbitrary objects.

Methods	w/ Flow	w/ PAN	w/ PMN	mAP
CenterTrack [57]	-	-	-	44.62
Tracktor [2]	-	-	-	51.69
FairMOT [53]	-	-	-	51.30
GTR [58]	-	-	-	58.04
DETracker	✓	×	×	60.99
DETracker	✓	✓	×	62.12
DETracker	✓	✓	✓	**62.93**

Table 6. **Comparison of detection performance in term of mAP on DogThruGlasses dataset testing set.**

	IDF1↑	FP↓	FN↓	IDs↓	MOTA↑	HOTA↑
LightTrack [25]	53.4	6240	12816	1955	30.8	48.5
FairMot [53]	41.4	2065	12753	3448	39.9	39.0
MPNTrack [5]	49.0	5918	11263	1039	40.0	40.7
CenterTrack[57]	37.2	2279	12379	3362	40.7	39.0
SORT [4]	48.3	2295	12960	1475	44.9	46.1
TraDeS [45]	53.6	3271	9102	1982	52.7	46.4
HandLer [14]	65.7	2875	6169	1256	66.1	57.2
DETracker (ours)	67.4	2781	6743	919	65.6	55.6

Table 7. **Comparing different methods on YouTube-Hand.**

6. Conclusion

In this work, we present DogThruGlasses, the first large-scale multi-object tracking dataset captured with wearable devices. The dataset is rich with deformation, occlusion, and ego motion, representing a broad spectrum of challenges commonly seen in real-world scenarios. We also proposed DETracker to explicitly tackle the two major problems in MOT: deformation and ego motion. The proposed method demonstrated proposing tracking quality on the challenging dataset.

References

[1] Nir Aharon, Roy Orfaig, and Ben-Zion Bobrovsky. Bot-sort: Robust associations multi-pedestrian tracking. *arXiv preprint arXiv:2206.14651*, 2022. 7

[2] Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixe. Tracking without bells and whistles. In *Proceedings of the International Conference on Computer Vision*, 2019. 7, 8

[3] Keni Bernardin and Rainer Stiefelhagen. Evaluating multiple object tracking performance: the clear mot metrics. *EURASIP Journal on Image and Video Processing*, 2008:1–10, 2008. 7

[4] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *Proceedings of the IEEE International Conference on Image Processing*. IEEE, 2016. 2, 8

[5] Guillem Brasó and Laura Leal-Taixé. Learning a neural solver for multiple object tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 2, 8

[6] Jiarui Cai, Mingze Xu, Wei Li, Yuanjun Xiong, Wei Xia, Zhuowen Tu, and Stefano Soatto. Memot: Multi-object tracking with memory. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8090–8100, 2022. 3, 6

[7] Patrick Dendorfer, Hamid Rezatofighi, Anton Milan, Javen Shi, Daniel Cremers, Ian Reid, Stefan Roth, Konrad Schindler, and Laura Leal-Taixé. Mot20: A benchmark for multi object tracking in crowded scenes. 2020. 2, 3

[8] Zhihong Fu, Qingjie Liu, Zehua Fu, and Yunhong Wang. Stmtrack: Template-free visual tracking with space-time memory networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13774–13783, 2021. 3

[9] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. 2012. 2, 3

[10] Shoudong Han, Piao Huang, Hongwei Wang, En Yu, Donghaisheng Liu, and Xiaofeng Pan. Mat: Motion-aware multi-object tracking. volume 476, pages 75–86. Elsevier, 2022. 2

[11] Jiawei He, Zehao Huang, Naiyan Wang, and Zhaoxiang Zhang. Learnable graph matching: Incorporating graph partitioning with deep feature learning for multiple object tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5299–5309, 2021. 2

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 2

[13] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *Proceedings of the International Conference on Computer Vision*, 2017. 4

[14] Mingzhen Huang, Supreeth Narasimhaswamy, Saif Vazir, Haibin Ling, and Minh Hoai. Forward propagation, backward regression, and pose association for hand tracking in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 2, 3, 8

[15] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. 1960. 2

[16] Chao Liang, Zhipeng Zhang, Xue Zhou, Bing Li, Shuyuan Zhu, and Weiming Hu. Rethinking the competition between detection and reid in multiobject tracking. *IEEE Transactions on Image Processing*, 31:3182–3196, 2022. 2

[17] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft COCO: Common objects in context. In *Proceedings of the European Conference on Computer Vision*, 2014. 8

[18] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 7

[19] Xiankai Lu, Wenguan Wang, Martin Danelljan, Tianfei Zhou, Jianbing Shen, and Luc Van Gool. Video object segmentation with episodic graph memory networks. In *European Conference on Computer Vision*, pages 661–679. Springer, 2020. 3

[20] Zhichao Lu, Vivek Rathod, Ronny Votel, and Jonathan Huang. Retinatrack: Online single stage joint detection and tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14668–14678, 2020. 2

[21] Jonathon Luiten, Aljosa Osep, Patrick Dendorfer, Philip Torr, Andreas Geiger, Laura Leal-Taixé, and Bastian Leibe. Hota: A higher order metric for evaluating multi-object tracking. *International Journal of Computer Vision*, 129(2): 548–578, 2021. 7

[22] Anton Milan, Laura Leal-Taixe, Ian Reid, Stefan Roth, and Konrad Schindler. Mot16: A benchmark for multi-object tracking. 2016. 2, 3, 7

[23] J. Munkres. Algorithms for the assignment and transportation problems. *Journal of the Society of Industrial and Applied Mathematics*, 5(1):32–38, March 1957. 2

[24] Supreeth Narasimhaswamy, Zhengwei Wei, Yang Wang, Justin Zhang, and Minh Hoai. Contextual attention for hand detection in the wild. In *Proceedings of the International Conference on Computer Vision*, 2019. 8

[25] Guanghan Ning and Heng Huang. Lighttrack: A generic framework for online top-down human pose tracking. *Proceedings of CVPR Workshop on Towards Human-Centric Image/Video Synthesis and the 4th Look Into Person Challenge*, 2020. 8

[26] Seoung Wug Oh, Joon-Young Lee, Ning Xu, and Seon Joo Kim. Video object segmentation using space-time memory networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9226–9235, 2019. 3

[27] Jinlong Peng, Changan Wang, Fangbin Wan, Yang Wu, Yabiao Wang, Ying Tai, Chengjie Wang, Jilin Li, Feiyue Huang, and Yanwei Fu. Chained-tracker: Chaining paired attentive regression results for end-to-end joint multiple-object detection and tracking. In *European conference on computer vision*, pages 145–161. Springer, 2020. 2

[28] Lorenzo Porzi, Markus Hofinger, Idoia Ruiz, Joan Serrat,

Samuel Rota Bulo, and Peter Kontschieder. Learning multi-object tracking and segmentation from automatic annotations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6846–6855, 2020. 2

[29] Akshay Rangesh, Eshed Ohn-Bar, Mohan M Trivedi, et al. Driver hand localization and grasp analysis: A vision-based real-time approach. In *International Conference on Intelligent Transportation Systems*, 2016. 3

[30] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. volume 28, 2015. 5

[31] Fatemeh Saleh, Sadegh Aliakbarian, Hamid Rezatofighi, Mathieu Salzmann, and Stephen Gould. Probabilistic tracklet scoring and inpainting for multiple object tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 2

[32] Bing Shuai, Andrew Berneshawi, Xinyu Li, Davide Modolo, and Joseph Tighe. Siammot: Siamese multi-object tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12372–12382, 2021. 2

[33] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 4

[34] Peize Sun, Jinkun Cao, Yi Jiang, Rufeng Zhang, Enze Xie, Zehuan Yuan, Changhu Wang, and Ping Luo. Transtrack: Multiple object tracking with transformer. *arXiv preprint arXiv:2012.15460*, 2020. 2

[35] ShiJie Sun, Naveed Akhtar, XiangYu Song, HuanSheng Song, Ajmal Mian, and Mubarak Shah. Simultaneous detection and tracking with motion modelling for multiple object tracking. In *European Conference on Computer Vision*, pages 626–643. Springer, 2020. 2

[36] ShiJie Sun, Naveed Akhtar, XiangYu Song, HuanSheng Song, Ajmal Mian, and Mubarak Shah. Simultaneous detection and tracking with motion modelling for multiple object tracking. In *Proceedings of the European Conference on Computer Vision*, 2020. 2

[37] 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*, 2017. 2

[38] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *European conference on computer vision*, pages 402–419. Springer, 2020. 2, 4, 7

[39] Pavel Tokmakov, Jie Li, Wolfram Burgard, and Adrien Gaidon. Learning to track with object permanence. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10860–10869, 2021. 2

[40] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, **L**ukasz Kaiser, and Illia Polosukhin. Attention is all you need. volume 30, 2017. 2, 5, 6

[41] Zhongdao Wang, Liang Zheng, Yixuan Liu, Yali Li, and Shengjin Wang. Towards real-time multi-object tracking. In *European Conference on Computer Vision*, pages 107–122. Springer, 2020. 2

[42] Nicolai Wojke and Alex Bewley. Deep cosine metric learning for person re-identification. In *Proceedings of the IEEE Workshop on Applications of Computer Vision*, 2018. 2

[43] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *Proceedings of the IEEE International Conference on Image Processing*. IEEE, 2017. 2, 7

[44] Chao-Yuan Wu, Christoph Feichtenhofer, Haoqi Fan, Kaiming He, Philipp Krahenbuhl, and Ross Girshick. Long-term feature banks for detailed video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 284–293, 2019. 3

[45] Jialian Wu, Jiale Cao, Liangchen Song, Yu Wang, Ming Yang, and Junsong Yuan. Track to detect and segment: An online multi-object tracker. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 1, 2, 8

[46] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019. 7

[47] Haofei Xu, Jing Zhang, Jianfei Cai, Hamid Rezatofighi, and Dacheng Tao. Gmflow: Learning optical flow via global matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8121–8130, 2022. 7, 8

[48] Jiarui Xu, Yue Cao, Zheng Zhang, and Han Hu. Spatial-temporal relation networks for multi-object tracking. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3988–3998, 2019. 2

[49] Tianyu Yang and Antoni B Chan. Learning dynamic memory networks for object tracking. In *Proceedings of the European conference on computer vision (ECCV)*, pages 152–167, 2018. 3

[50] Junbo Yin, Wenguan Wang, Qinghao Meng, Ruigang Yang, and Jianbing Shen. A unified object motion and affinity model for online multi-object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6768–6777, 2020. 2

[51] Fisher Yu, Dequan Wang, Evan Shelhamer, and Trevor Darrell. Deep layer aggregation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 4, 7

[52] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2

[53] Yifu Zhan, Chunyu Wang, Xinggang Wang, Wenjun Zeng, and Wenyu Liu. A simple baseline for multi-object tracking. *arXiv preprint arXiv:2004.01888*, 2020. 2, 7, 8

[54] Yifu Zhang, Peize Sun, Yi Jiang, Dongdong Yu, Fucheng Weng, Zehuan Yuan, Ping Luo, Wenyu Liu, and Xinggang Wang. Bytetrack: Multi-object tracking by associating every detection box. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXII*, pages 1–21. Springer, 2022. 2, 7

[55] Liang Zheng, Yi Yang, and Alexander G Hauptmann. Person re-identification: Past, present and future. 2016. 8

[56] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019. 2, 6, 7

[57] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Tracking objects as points. *arXiv preprint arXiv:2004.01177*, 2020. 2, 3, 7, 8

[58] Xingyi Zhou, Tianwei Yin, Vladlen Koltun, and Philipp Krähenbühl. Global tracking transformers. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 5, 7, 8