Object Hypotheses as Points for Efficient Multi-Object Tracking

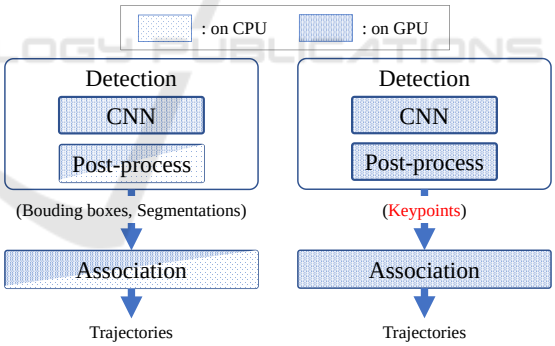
Shuhei Tarashima
Innovation Center, NTT Communications Corporation, Japan

Keywords: Multi-Object Tracking, CPU-GPU Data Transfer, Data Association.

Abstract: In most multi-object tracking (MOT) approaches under the tracking-by-detection framework, object detection and hypothesis association are addressed separately by setting bounding boxes as interfaces among them. This subdivision has greatly yielded advantages with respect to tracking accuracy, but it often lets researchers overlook the efficiency of whole MOT pipelines, since these interfaces can cause the time-consuming data communication between CPU and GPU. Alternatively, in this work we define an object hypothesis as a keypoint representing the object center, and propose simple data association algorithms based on the spatial proximity of keypoints. Different from standard data association methods like Hungarian algorithm, our approach can easily be run on GPU, which enables direct feed of detection results generated on GPU to our tracking module without the need of CPU-GPU data transfer. In this paper we conduct a series of experiments on MOT16, MOT17 and MOT-Soccer datasets in order to show that (1) our tracking module is much more efficient than existing methods while achieving competitive MOTA scores, (2) our tracking module run on GPU can improve the whole MOT efficiency via reducing the overhead of CPU-GPU data transfer between detection and tracking, and (3) our tracking module can be combined to a state-of-the-art unsupervised MOT method based on joint detection and embedding and successfully improve its efficiency.

1 INTRODUCTION

Multi-Object Tracking (MOT) aims to recover trajectories of objects from target categories in a given video, which has myriad of applications in surveillance (Alldieck et al., 2016), autonomous driving (Ošep et al., 2017), sports analysis (Zhang et al., 2020a) and biomedical image understanding (Liang et al., 2013; Meirovitch et al., 2019). Most recent approaches follow the *tracking-by-detection* paradigm, in which object detectors are first applied to each individual frame to find the object locations (*e.g.* bounding boxes (Kim et al., 2015; Tang et al., 2016), segmentations (Sun et al., 2018; Voigtlaender et al., 2019)), then these hypotheses are associated across frames to form trajectories of the same identity. Usually, under the tracking-by-detection framework object detection and data association are addressed separately: Bounding boxes or segmentations generated by object detectors are fed into a data association module, while no other knowledge is assumed to be transferred. This simple subdivision has led researchers to take significant advantage of existing object detectors based on convolutional neural networks (CNNs) (Ren et al., 2015; Choi et al., 2019; Redmon and Farhadi, 2018) running on GPU, and allow them to focus on the problem of linking hypotheses in video frames to estimate their tracks. However, at



(a) Existing Unsupervised MOT (b) CenterTracker (Proposed)

Figure 1: Different from existing unsupervised MOT approaches where object bounding boxes or segmentations are the interfaces between detection and association (as in (a)), our approach, named CenterTracker, feeds a set of keypoints representing object centers to a data association module (as in (b)). Current trajectories and detections are linked based on the spatial proximity between them. Since this data association can easily be run on GPU, this approach can reduce the communication overhead between CPU and GPU.

Tarashima, S.
Object Hypotheses as Points for Efficient Multi-Object Tracking.
DOI: 10.5220/0010343508280835
In *Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2021) - Volume 5: VISAPP*, pages 828-835
ISBN: 978-989-758-488-6

the same time, this often lets researchers overlook the efficiency of whole MOT algorithms: Basically, as shown in Figure 1 (a), modern object detectors feed an input image into CNNs to produce a set of tensors, which are transformed into final detection results in the following post-processing step. Notice that in many cases CNN feedforwarding is performed on GPU while post-processing is on CPU. In these cases tensors to be transformed are transferred from GPU to CPU, which may significantly slow down the overall throughput of MOT. While recently some deep learning frameworks provide GPU-based post-processing, CPU-GPU data communication is still required between detection and association when tracking is performed in an unsupervised manner (Bochinski et al., 2017; Bochinski et al., 2018; Bewley et al., 2016; Wojke et al., 2017; Fu et al., 2019). From these facts, we can say that the CPU-GPU communication overhead has not been fully addressed in the current tracking-by-detection framework.

This issue motivates us to propose an alternative to a bounding box or a segmentation as an interface between detection and association. Specifically, as shown in Figure 1 (b), we consider an object hypothesis as a keypoint representing its center location. This idea allow us to (i) design an efficient data association algorithm that can be easily run on GPU and (ii) directly feed detection outputs of recent object detectors (Redmon and Farhadi, 2018; Zhou et al., 2019a) into our data association module without the need of CPU-GPU data transfer. To this end, we propose a simple data association algorithm, named CenterTracker, which can efficiently be run on both CPU and GPU and enables whole MOT pipelines to be performed on GPU while minimizing CPU-GPU data transfer. Notice that since our data association is performed in an unsupervised manner, CenterTracker does not require any training for data association. Also, while in the literature points are used in feature tracking (Tomasi and Kanade, 1991; Shi and Tomasi, 1994), to the best of our knowledge this work is the first attempt to address the unsupervised data association problem in MOT by setting target objects as points.

We conduct a series of experiments on several MOT benchmarks including MOT16, MOT17 (Milan et al., 2016) and MOT-Soccer (Fu et al., 2019) datasets. Through the evaluations we will show that:

- CenterTracker is much more efficient than existing methods while achieving competitive MOTA scores. Specifically, on GPU, the tracking efficiency achieves 3500-4000 FPS which is 2-4 times faster than the state-of-the-art (§4.2).
- CenterTracker run on GPU can improve the whole MOT efficiency via reducing the overhead of CPU-GPU data transfer between detection and tracking (§4.3).
- CenterTracker can be combined to a state-of-the-art unsupervised MOT method (Zhang et al., 2020b) and can successfully improve the whole efficiency (§4.4).

2 RELATED WORKS

2.1 Unsupervised Data Association

Efficiency is crucial for MOT due to urgent needs from various applications such as surveillance, sports analysis and autonomous driving, all of which exploit object trajectories as input (Murray, 2017; Wang et al., 2019b). Existing works have addressed this issue based on the assumptions that detection is becoming more accurate and video frame rate is getting higher. For example, Bochinski *et al.* (Bochinski et al., 2017) built a MOT algorithm that simply evaluates spatial overlap (*i.e.* Intersection-over-Union, IoU) between object hypotheses from different frames to associate them. Bewley *et al.* (Bewley et al., 2016) combined motion modeling (*i.e.* the Kalman filter) and graph-based optimization (*i.e.* the Hungarian algorithm) with Bochinski's approach to improve tracking accuracy. Several following works (Bochinski et al., 2018; Chen et al., 2018; Wojke et al., 2017; Wojke and Bewley, 2018; Bergmann et al., 2019) further incorporated appearance cues into their MOT algorithms, in order to improve identity preservation across frames under cluttered environments. While these methods are highly efficient with respect to data association, they overlook the whole efficiency of MOT since the core components run on CPU while input detections are usually derived from GPU processing. Deploying these data association methods on GPU is one way to address the issue, but this approach includes potential challenges since usually there are difficult parts to be parallelized (*i.e.* difficult to be run on GPU efficiently).

In this work we explore another way to improve the whole efficiency of unsupervised MOT: In our CenterTracker we modify the form of detection input and introduce novel data association modules that can be easily deployed in GPU. Notice that our approach is different from supervised data association methods (Feichtenhofer et al., 2017; Sadeghian et al., 2017; Voigtlaender et al., 2019; Xu et al., 2019; Wang et al., 2019a; Liu et al., 2019; Zhou et al., 2020; Brasó and

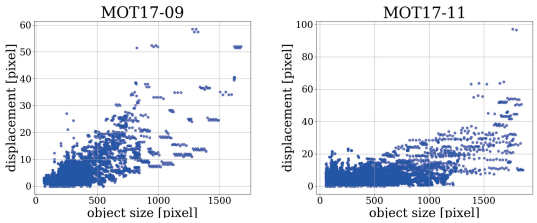


Figure 2: These scatter plots show the frame-by-frame displacements of objects with different sizes in two MOT17 sequences (Milan et al., 2016). We define an object size as the diagonal length of the object shown in the first frame. We can clearly see bigger objects can move larger whether a camera moves (*i.e.* MOT17-09) or not (*i.e.* MOT17-11).

Leal-Taixé, 2019). In our approach we perform data association in an unsupervised manner, which does not require any training. We describe the detail of our CenterTracker in §3 and compare it to existing unsupervised data association methods (Bewley et al., 2016; Bochinski et al., 2017) in §4.3.

There are other lines of research that also aim for higher MOT efficiency via jointly performed detection and embedding (for data association) in one CNN on GPU (Wang et al., 2019b; Zhang et al., 2020b; Karthik et al., 2020). In this work we assume that these joint detection and embedding (JDE) methods are orthogonal to our data association: We propose to combine them so as to achieve better tradeoffs between accuracy and efficiency. In §3.2 we will explain how to extend our data association module to accommodate appearance features, and evaluate its performance in §4.4.

2.2 Object Detector

Convolutional neural networks (CNNs) have been a dominant approach for object detection in recent years. CNN-based detectors can be categorized into one-stage detectors (Redmon and Farhadi, 2018; Choi et al., 2019) and two-stage detectors (Ren et al., 2015; Yang et al., 2016), where one-stage methods achieve higher efficiency while slightly sacrificing their accuracy. Typical one-stage detectors like YOLO (Redmon and Farhadi, 2018) place anchor boxes over an image by dividing it into sparse grid units, and generate final box predictions by scoring anchor boxes and refining their coordinates through regression. Alternatively, in recently proposed anchor-free one-stage detectors (Duan et al., 2019; Zhou et al., 2019a; Zhou et al., 2019b; Law and Deng, 2018; Tian et al., 2019), an object is represented by a (set of) keypoint(s) and its location is retrieved as a peak in a heatmap predicted by CNNs. In this anchor-free approach corresponding object locations (*e.g.* bounding boxes, segmentation) can be recovered using auxiliary CNN outputs (*e.g.* object size).

In most one-stage detectors mentioned above, detection results are generated as its center location and auxiliary information, which is favorable to our proposed CenterTracker that assumes the same format of object hypothesis as input. Based on this observation, in our experiments we combine our CenterTracker with a popular anchor-free object detector, CenterNet (Zhou et al., 2020), so as to build whole MOT pipelines. While CenterTracker can potentially be combined with other detectors, we leave this validation as our future research.

3 APPROACH

The task of multi-object tracking (MOT) is to extract trajectories (*i.e.* spatial and temporal positions) of objects from a target category in a given frame sequence. In most cases the number of objects is not known a priori. We define a trajectory as a list of ordered object locations $T_k = \{o_k^{t_1}, o_k^{t_2}, ...\}$, where $o_k^t.\mathbf{c} \in \mathbb{R}^2$ is the center of object k at time t, $o_k^t.\mathbf{wh} \in \mathbb{R}^2$ is its size (*i.e.* width and height) and $o_k^t.s \in \mathbb{R}$ is the confidence score. From each object location o, the corresponding bounding box can be easily recovered as follows:

$$(x-w/2, y-h/2, x+w/2, y+h/2), \quad (1)$$

where $(x,y) = o.\mathbf{c}$ and $(w,h) = o.\mathbf{wh}$.

To fairly validate our key idea, we first build the vanilla version of our approach, named vanilla CenterTracker, by referring to the IoU Tracker (Bochinski et al., 2017), which is to our knowledge one of the most efficient MOT algorithms. Specifically, we directly follow the trajectory deactivation scheme in IoU Tracker[1]. Notice that this vanilla CenterTracker does not consider motion and appearance of objects. In §3.2 we discuss a simple extension of this algorithm with respect to the appearance perspective.

3.1 Vanilla CenterTracker

The algorithm of our CenterTracker is outlined in Algorithm 1. At every timestep t, our algorithm associates a list of object hypothesis D^t with current trajectories $\mathcal{T}_{\text{active}}$ based on spatial proximity of centers and object size (line 3-6). Trajectories are updated, finished or killed based on the association result (line 7-13), and the remaining hypotheses are used to initialize new tracks (line 14-16). For simplicity, in Algorithm 1 we do not show the pipeline of recovering

[1]If a track has shorter length than t_{min} and has least one detection with higher confident score than σ_h, it is finished.

Algorithm 1: Vanilla CenterTracker.

Data: A video sequence $V = \{I^1, I^2, ..., I^L\}$ and a detection sequence $D = \{D^1, D^2, ..., D^L\}$ with $D^t = \{o_1^t, o_2^t, ...\}$ is a list of object hypothesis o_i^t, where $o_i^t.\mathbf{c} \in \mathbb{R}^2$ is its center, $o_i^t.\mathbf{wh} \in \mathbb{R}^2$ is the size (i.e. width and height), and $o_i^t.s \in \mathbb{R}$ is a detection confidence

Result: A set of trajectories $\mathcal{T}_{\text{finish}} = \{T_1, T_2, ...\}$ with $T_k = \{o_k^{t_1}, ..., o_k^{t_N} \mid 0 \leq t_1, ..., t_N \leq L\}$ as an ordered list of object locations o_k^t

1 $\mathcal{T}_{\text{active}} \longleftarrow \emptyset$, $\mathcal{T}_{\text{finish}} \longleftarrow \emptyset$
2 **for** $t \longleftarrow 1$ **to** L **do**
3 $C \longleftarrow \emptyset$
4 **for** $T_k \in \mathcal{T}_{\text{active}}$ **do**
5 $C \longleftarrow T_k[-1]$
  end
  /* $\mathcal{M}$:tuples of matched track index and hypothesis, */
  /* U:unmatched track indices, D^*:remaining hypotheses */
6 $\mathcal{M}, \mathcal{U}, D^* \longleftarrow$ `associate`(C, D_t)
7 **for** $(k, o) \in \mathcal{M}$ **do**
8 $\mathcal{T}_{\text{active}}[k]$.`update`$(o)$
  end
9 **for** $k \in U$ **do**
10 $T \longleftarrow \mathcal{T}_{\text{active}}[k]$
11 **if** $\max_{o \in T}\{o.s\} \geq \sigma_h$ **and** `len`$(T) \geq t_{\text{min}}$ **then**
12 $\mathcal{T}_{\text{finish}} \longleftarrow \mathcal{T}_{\text{finish}} + T$
13 $\mathcal{T}_{\text{active}} \longleftarrow \mathcal{T}_{\text{active}} \setminus T$
  end
  end
14 **for** $o \in D^*$ **do**
15 $T \longleftarrow$ `init_track`(o)
16 $\mathcal{T}_{\text{active}} \longleftarrow \mathcal{T}_{\text{active}} + T$
  end
17 **if** $t = L$ **then**
18 **for** $T \in \mathcal{T}_{\text{active}}$ **do**
19 **if** $\max_{o \in T}\{o.s\} \geq \sigma_h$ **and** `len`$(T) \geq t_{\text{min}}$ **then**
20 $\mathcal{T}_{\text{finish}} \longleftarrow \mathcal{T}_{\text{finish}} + T$
  end
  end
  end
end

a bounding box from the attributes of an object o. As is mentioned before, this recovery is only required just before the final output. In the following, we detail the components of CenterTracker focusing on data association (*i.e.* `associate()` in line 6).

Data Association: As with recent MOT approaches, we assume video frame rates are sufficiently high and therefore targets move only slightly between consecutive frames (Wojke et al., 2017; Bochinski et al., 2017; Bochinski et al., 2018; Bergmann et al., 2019). With this assumption the spatial proximity of centers between a track and a hypothesis can be seen as a strong cue to link them (Xu et al., 2020). Our approach follows the above notion in a straightforward way: We simply evaluate $L2$-norms of all the pairs of tracks and hypotheses, then choose the nearest hypothesis for each track as its new center location if the displacement is smaller than a pre-defined threshold σ_{disp}. When the same hypothesis are selected by multiple tracks, the nearest track is linked to the hypothesis and the remaining tracks fail to find new locations.

To further improve the matching performance, in our approach we exploit the object scale of each track. Intuitively, the displacement of larger objects can be bigger in a pixel coordinate system. As shown in Figure 2, we can easily validate this intuition using the public MOT benchmarks (Leal-Taixé et al., 2015; Milan et al., 2016). Therefore, in our approach we adaptively set displacement thresholds (*i.e.* σ_{disp}) for object sizes of tracks. Specifically, we set the threshold of track k as $\sigma_{\text{disp}}^k = \sigma_{size}\sqrt{(w_{-1}^k)^2 + (h_{-1}^k)^2}$, where (w_{-1}^k, h_{-1}^k) is the last object size in track k and σ_{size} is another threshold. We use a simple grid search on MOT17 training sequences to determine the best σ_{size}. Unless mentioned otherwise we set $\sigma_{\text{size}} = 0.08$, which can produce good results on other datasets (*e.g.* MOT-Soccer).

Notice that graph-based optimization (*e.g.* the Hungarian algorithm, the Kuhn-Munkres algorithm) is another choice for data association based on the spatial proximity of center locations. However, these algorithms cannot be run in GPU efficiently since they are difficult to parallelize. Although sophisticated CPU implementations of these optimizations are easily available, this scheme unavoidably leads data transfer from GPU to CPU, which slows down the whole MOT algorithm. Contrary, our proposed method can be performed efficiently on GPU. In section 4.1 we will perform ablation studies to validate our data association approach.

3.2 CenterTracker+

We here extend the above vanilla CenterTracker by combining appearance modeling techniques, resulting in CenterTracker+.

Appearance Model: To avoid tracks to be fragmented, we exploit visual information to keep short but highly-confident tracks active. Specifically, if a track that cannot find a new hypothesis contains at least one detection with higher confidence score than σ_h, we store the track for a fixed number of frames while extracting a visual feature corresponding the the detection. We also extract visual features from unmatched hypotheses in the current frame, then compare the feature in the embedding space and re-identify via a threshold. Basically, in this scheme any feature extractor tailored to identify the same object can be applied. Considering the data transfer between CPU and GPU, using embedding features extracted from recently proposed joint detection and embedding (JDE) approaches is a good choice. (Wang et al., 2019b; Zhang et al., 2020b; Karthik et al., 2020) We call our approach with motion and appearance modeling as CenterTracker+. In §4.4 we evaluate the method in combination with a state-of-the-art JDE method (Zhang et al., 2020b).

4 EVALUATION

In this paper we validate our CenterTracker with respect to the efficiency of CenterTracker itself (§4.2), the efficiency of the whole tracking pipeline (§4.3) and the compatibility with a state-of-the-art unsupervised MOT approach (§4.4). We use the following three public datasets for our experiments:

MOT16 and MOT17 (Milan et al., 2016): These datasets from the MOTChallenge benchmarks[2] are the most standard datasets for MOT, which include several challenging pedestrian tracking sequences with frequent occlusions and crowded scenes.

MOT-Soccer[3] (Fu et al., 2019): This dataset is recently introduced by Fu *et al.*, which consists of 10 clips of amateur soccer videos captured by a static camera installed in a straight view of high position. Different from other tracking tasks, the objects in this dataset display smaller scale changes as well as relatively similar appearance features.

Unless mentioned otherwise, we report the standard MOT metrics including MOTA (Milan et al., 2016), IDF1 (Ristani et al., 2016), percentage of Mostly Tracked trajectories (MT), percentage of Mostly Lost trajectories (ML) and the number of IDentity switches (IDs) in addition to Frames Per Second of the whole pipeline including detection, tracking and data transfer (FPSW). Higher MOTA, IDF1, MT and FPSW are better while lower ML and IDs are better.

Table 1: Ablations for data association on MOT17.

	MOTA	IDF1	FPSW
Hungarian	40.1	42.2	28.1
Direct tuning of σ_{disp}	41.1	43.6	30.3
Proposed	41.4	43.8	30.2

4.1 Ablation Study

We first perform an ablation study with respect to our data association module. To do so, we replace our data association module with the Hungarian algorithm, then measure its tracking accuracy and efficiency. Also, to evaluate the effect of considering the object scales in our data association, we run another algorithm in which σ_{disp} (*cf.* §3.1) is directly tuned. The results are shown in Table 1. As expected, our proposed data association is faster than the Hungarian algorithm. Considering the scale information further improves both MOTA and IDF1 without sacrificing the efficiency. Interestingly, our methods outperform Hungarian algorithm with respect to tracking accuracy. One possible reason is that finding nearest neighbors of centers works very well and considering subsequent neighbors may decrease the accuracy in MOT17 training sequences.

4.2 Efficiency of Tracking Module

Here we first evaluate our CenterTracker focusing on the tracking module itself. To do so, we use public detections provided by MOT17 and MOT-Soccer datasets as input to the tracker and compare the MOTA-FPS balance (Leal-Taixé et al., 2015; Bewley et al., 2016; Fu et al., 2019) to that of existing methods. Figure 3 (a) shows the result for MOT17 and (b) shows the result for MOT-Soccer, respectively. In both datasets we show the results of our CenterTracker run on both CPU and GPU. For MOT17 we draw the results of the existing fastest trackers[4] from the MOTChallenge leaderboard[5]. For MOT-Soccer we show the results of IoUTracker (Bochinski et al., 2017) and SORT (Bewley et al., 2016) (fastest trackers in MOT17) which are obtained by running official codes[6,7], while we directly draw the results of MF-SORT (Fu et al., 2019) and DeepSORT (Wojke et al., 2017) from the dataset paper (Fu et al., 2019). From both results, our CenterTracker achieves the fastest

[2]https://motchallenge.net/

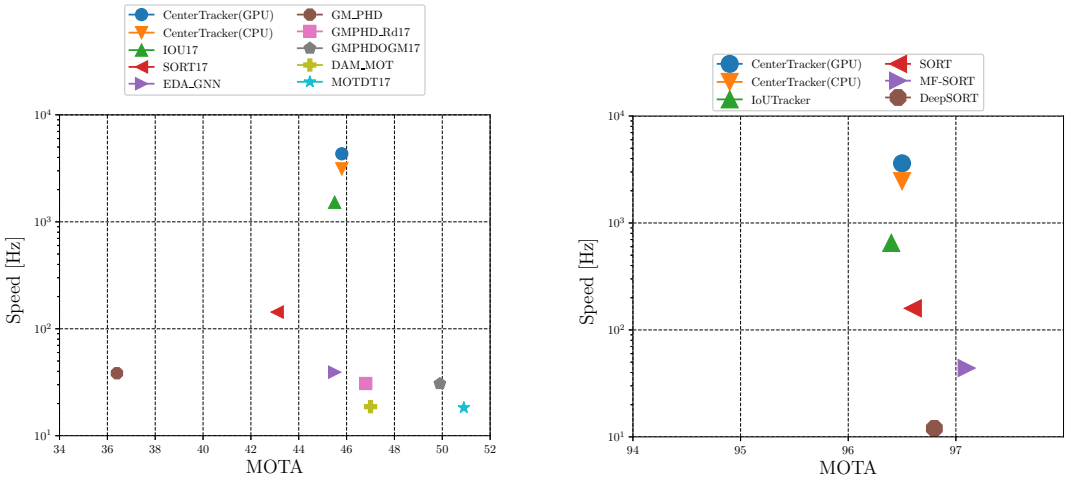
[3]https://github.com/jozeeandfish/motsoccer:

[4]We do not include anonymous submissions.

[5]https://motchallenge.net/results/MOT17/

[6]https://github.com/bochinski/iou-tracker

[7]https://github.com/abewley/sort



(a) MOT17 (b) MOT-Soccer

Figure 3: MOTA-FPS balances for MOT17 (a) and MOT-Soccer (b) datasets. CenterTracker (CPU) represents our proposed tracker implemented on CPU, while CenterTracker (GPU) is implemented on GPU. For MOT17 results of existing trackers are drawn from the MOTChallenge leaderboard. For MOT-Soccer we run IoUTracker and SORT by ourselves while drawing results of other trackers from the dataset paper (Fu et al., 2019).

among the competitors while maintaining reasonable MOTA scores. While CenterTracker on CPU can already be run very fast ($>$ 3000 FPS on MOT17 and $>$ 2500 FPS on MOT-Soccer), it is further accelerated by GPU ($>$ 4000 FPS on MOT17 and $>$ 3500 FPS on MOT-Soccer), which are about 2-4 times faster than the faster existing tracker (*i.e.* IoUTracker). These results indicate the efficiency of our CenterTracker itself.

4.3 Efficiency of Whole Tracking Pipeline

When our CenterTracker is combined with an efficient detector and they are both run on GPU, we can directly feed the detection results to CenterTracker without any CPU-GPU data transfer, which should also improve the whole efficiency of MOT. We experimentally validate the effect using the MOT-Soccer dataset. Specifically, we build a whole MOT pipeline using CenterNet (Zhou et al., 2019a) as a detector and our CenterTracker as a tracker, then evaluate the whole running times of MOT including detection, tracking and data transfer by switching the hardware on which the tracker is run. For CenterNet, we use the ResNet-18 backbone finetuned with images in MOT-Soccer training sequences. Following the recent MOT protocols (Wang et al., 2019b; Zhou et al., 2020; Zhang et al., 2020b; Karthik et al., 2020), we resize every frame to 1088×608 and feed it to the detector. To perform comparison, we also build other MOT pipelines by replacing CenterTracker with IoUTracker (Bochinski et al., 2017) and SORT (Bewley et al., 2016), both of which are run on CPU. The results are shown in Figure 4 and Table 2. Notice that in all the cases detection is performed on GPU. From Table 2 our CenterTracker achieves 88.8 MOTA score and 89.0 IDF1 score in 36.4 FPS, which is more efficient than alternatives while keeping almost the same MOT scores with them. From Figure 4, we can see CenterTracker (orange bars in Figure 4) is accelerated by GPU, and more importantly, the elapsed time for data transfer (green bars) is almost halved by directly feeding detections to CenterTracker among the GPU memory. This result indicates that our CenterTracker can contribute to reducing the overhead of MOT pipelines and improve efficiency as a whole.

4.4 Compatibility with State-of-the-Art

As mentioned in §3.2, our CenterTracker can be combined with the state-of-the-art MOT methods using unsupervised data association. In this paper we adopt FairMOT (Zhang et al., 2020b) as a testbed and replace its data association module to our CenterTracker+. We choose FairMOT due to its excellent balance between tracking performance and efficiency. We use MOT16, MOT17 and MOT-Soccer datasets for comparison and results are shown in Table 3. For MOT16 and MOT17 we use the DLA-34 backbone with provided weight parameters for detection and embedding. For MOT-Soccer we also use the DLA-34 backbone while its parameters are finetuned with MOT-Soccer training sequences. From Table 3

Table 2: Tracking performance on the MOT-Soccer (Fu et al., 2019) test dataset. In all the settings we use CenterNet (Zhou et al., 2019a) with ResNet-18 backbone finetuned by the MOT-Soccer train sequences as the person detector.

Tracker	MOTA↑	IDF1↑	MT↑	ML↓	IDs↓	FPSW↑
IoUTracker (Bewley et al., 2016)	88.7	89.3	93.5	1.0	160	34.5
SORT (Bewley et al., 2016)	88.4	89.5	94.5	1.0	156	31.2
CenterTracker (GPU)	88.8	89.0	93.5	1.0	157	36.4

Table 3: We replace the data association module of FairMOT (Zhang et al., 2020b) to our CenterTracker and compare both tracking performance and efficiency on MOT16, MOT17 and MOT-Soccer datasets.

Dataset	Tracker	MOTA↑	IDF1↑	MT↑	ML↓	IDs↓	FPSW↑
MOT16	FairMOT	68.7	70.4	39.5	19.0	953	23.4
	CenterTracker+	68.1	68.6	34.8	19.0	1021	27.8
MOT17	FairMOT	67.5	69.8	37.7	20.8	2868	23.4
	CenterTracker+	66.7	68.5	35.0	21.2	2934	27.8
MOT-Soccer	FairMOT	89.6	90.5	95.4	1.0	147	22.1
	CenterTracker+	89.2	89.5	93.5	1.0	154	27.1

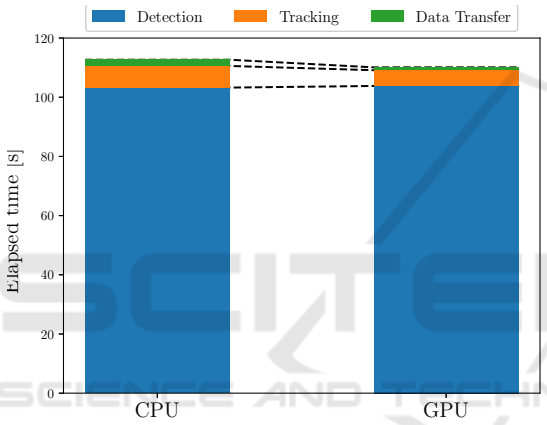


Figure 4: Total elapsed times [s] for tracking all the test sequences in MOT-Soccer (Fu et al., 2019) dataset when CenterTracker is implemented on CPU (left) and GPU (right). In both cases detection results are generated by CenterNet (Zhou et al., 2019a) run on GPU and finetuned with MOT-Soccer training images. Notice that in the GPU case detections are passed to CenterTracker without GPU-CPU data transfer.

we can see the efficiency (*i.e.* FPSW) is improved in all the datasets with minimum sacrifice of tracking performance.

5 CONCLUSION

In this work we proposed an efficient data association algorithms named CenterTracker that are friendly for GPU processing and help minimize the overhead of MOT pipelines due to unnecessary data transfer between CPU and GPU. Our experiments on MOT16, MOT17 and MOT-Soccer datasets showed that our CenterTracker is much more efficient than existing trackers, and can successfully improve the whole MOT efficiency by directly feeding detection results to our tracking module without GPU-CPU data transfer. Also, we showed that CenterTracker can be combined with a state-of-the-art unsupervised MOT algorithm (Zhang et al., 2020b) and improve its efficiency with minimum sacrifice of MOT scores.

In the future we will further evaluate our approach on different datasets, different settings (*e.g.* combined with other detectors than CenterNet(Zhou et al., 2019a)) and different objects (Zhu et al., 2018; Dendorfer et al., 2020). We will also explore the ways to further improve our data association algorithms that can improve both MOT metrics and efficiency.

REFERENCES

Alldieck, T., Bahnsen, C. H., and Moeslund, T. B. (2016). Context-aware fusion of rgb and thermal imagery for traffic monitoring. *Sensors*.

Bergmann, P., Meinhardt, T., and Leal-Taixé, L. (2019). Tracking without bells and whistles. In *ICCV*.

Bewley, A., Ge, Z., Ott, L., Ramos, F., and Upcroft, B. (2016). Simple online and realtime tracking. In *ICIP*.

Bochinski, E., Eiselein, V., and Sikora, T. (2017). High-speed tracking-by-detection without using image information. In *AVSS Workshop*.

Bochinski, E., Senst, T., and Sikora, T. (2018). Extending iou based multi-object tracking by visual information. In *AVSS*.

Brasó, G. and Leal-Taixé, L. (2019). Learning a neural solver for multiple object tracking. *arXiv preprint arXiv:1912.07515*.

Chen, L., Ai, H., Zhuang, Z., and Shang, C. (2018). Real-time multiple people tracking with deeply learned candidate selection and person re-identification. In *ICME*.

Choi, J., Chun, D., Kim, H., and Lee, H.-J. (2019). Gaussian yolov3: An accurate and fast object detector us-

ing localization uncertainty for autonomous driving. In *ICCV*.

Dendorfer, P., Rezatofighi, H., Milan, A., Shi, J., Cremers, D., Reid, I., Roth, S., Schindler, K., and Leal-Taixé, L. (2020). Mot20: A benchmark for multi object tracking in crowded scenes. *arXiv preprint arXiv:2003.09003*.

Duan, K., Bai, S., Xie, L., Qi, H., Huang, Q., and Tian, Q. (2019). Centernet: Keypoint triplets for object detection. In *CVPR*.

Feichtenhofer, C., Pinz, A., and Zisserman, A. (2017). Detect to track and track to detect. In *ICCV*.

Fu, H., Wu, L., Jian, M., Yang, Y., and Wang, X. (2019). Mf-sort: Simple online and realtime tracking with motion features.

Karthik, S., Prabhu, A., and Gandhi, V. (2020). Simple unsupervised multi-object tracking. *arXiv preprint arXiv:2006.02609*.

Kim, C., Li, F., Ciptadi, A., and Rehg, J. M. (2015). Multiple hypothesis tracking revisited. In *ICCV*.

Law, H. and Deng, J. (2018). Cornernet: Detecting objects as paired keypoints. In *ECCV*.

Leal-Taixé, L., Milan, A., Reid, I., Roth, S., and Schindler, K. (2015). Motchallenge 2015: Towards a benchmark for multi-target tracking. *arXiv preprint arXiv:1504.01942*.

Liang, L., Shen, H., Rompolas, P., Greco, V., Camilli, P. D., and Duncan, J. S. (2013). A multiple hypothesis based method for particle tracking and its extension for cell segmentation. In *Inf Process Med Imaging*.

Liu, Q., Liu, B., Wu, Y., Li, W., and Yu, N. (2019). Real-time online multi-object tracking in compressed domain. *IEEE Access*.

Meirovitch, Y., Mi, L., Saribekyan, H., Matveev, A., Rolnick, D., and Shavit, N. (2019). Cross-classification clustering: An efficient multi-object tracking technique for 3-d instance segmentation in connectomics. In *CVPR*.

Milan, A., Leal-Taixé, L., Reid, I., Roth, S., and Schindler, K. (2016). Mot16: A benchmark for multi-object tracking. *arXiv preprint arXiv:1603.00831*.

Murray, S. (2017). Real-time multiple object tracking - a study on the importance of speed. *arXiv preprint arXiv:1709.03572*.

Ošep, A., Mehner, W., Mathias, M., and Leibe, B. (2017). Combined image- and world-space tracking in traffic scenes. In *ICRA*.

Redmon, J. and Farhadi, A. (2018). Yolov3: An incremental improvement. In *arXiv preprint arXiv:1804.02767*.

Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*.

Ristani, E., Solera, F., Zou, R. S., Cucchiara, R., and Tomasi, C. (2016). Performance measures and a data set for multi-target, multi-camera tracking. In *ECCV Workshop*.

Sadeghian, A., Alahi, A., and Savarese, S. (2017). Tracking the untrackable: Learning to track multiple cues with long-term dependencies. In *ICCV*.

Shi, J. and Tomasi, C. (1994). Good features to track. In *CVPR*.

Sun, S., Akhtar, N., Song, H., Mian, A., and Shah, M. (2018). Deep affinity network for multiple object tracking. *TPAMI*.

Tang, S., Andres, B., Andriluka, M., and Schiele, B. (2016). Multi-person tracking by multicut and deep matching. In *ECCV*.

Tian, Z., Shen, C., Chen, H., and He, T. (2019). Fcos: Fully convolutional one-stage object detection. In *ICCV*.

Tomasi, C. and Kanade, T. (1991). Detection and tracking of point features. Technical report, Technical Report CMU-CS-91-132, Carnegie Mellon University.

Voigtlaender, P., Krause, M., Ošep, A., and Luiten, J. (2019). Mots: Multi-object tracking and segmentation. In *CVPR*.

Wang, G., Wang, Y., Zhang, H., Gu, R., and Hwang, J.-N. (2019a). Exploit the connectivity: Multi-object tracking with trackletnet. In *ACMMM*.

Wang, Z., Zheng, L., Liu, Y., and Wang, S. (2019b). Towards real-time multi-object tracking. *arXiv preprint arxiv:1909.12605*.

Wojke, N. and Bewley, A. (2018). Deep cosine metric learning for person re-identification. In *WACV*.

Wojke, N., Bewley, A., and Paulus, D. (2017). Simple online and realtime tracking with a deep association metric. In *ICIP*.

Xu, J., Cao, Y., Zhang, Z., and Hu, H. (2019). Spatial-temporal relation networks for multi-object tracking. In *ICCV*.

Xu, Y., Ban, O. Y., Horaud, R., Leal-Taixé, L., and Alameda-Pineda, X. (2020). How to train your deep multi-object tracker. In *CVPR*.

Yang, F., Choi, W., and Lin, Y. (2016). Exploit all the layers: Fast and accurate cnn object detector with scale dependent pooling and cascaded rejection classifiers. In *CVPR*.

Zhang, R., Wu, L., Yang, Y., Wu, W., Chen, W. Y., and Xu, M. (2020a). Multi-camera multi-player tracking with deep player identification in sports video. *Pattern Recognition*.

Zhang, Y., Wang, C., Wang, X., Zeng, W., and Liu, W. (2020b). A simple baseline for multi-object tracking. *arXiv preprint arXiv:2004.01888*.

Zhou, X., Koltun, V., and Krähenbühl, P. (2020). Tracking objects as points. *arXiv preprint arXiv:2004.01177*.

Zhou, X., Wang, D., and Krähenbühl, P. (2019a). Objects as points. In *arXiv preprint arXiv:1904.07850*.

Zhou, X., Zhuo, J., and Krähenbühl, P. (2019b). Bottom-up object detection by grouping extreme and center points. In *CVPR*.

Zhu, P., Wen, L., Bian, X., Ling, H., and Hu, Q. (2018). Vision meets drones: A challenge. *arXiv preprint arXiv:1804.07437*.