Tracking Objects as Points — Supplementary Materials

Xingyi Zhou[1], Vladlen Koltun[2], and Philipp Krähenbühl[1]

[1]UT Austin, [2]Intel Labs

1 Tracking algorithms

1.1 Private tracking

We adopt a simple greedy id association algorithm based on the center distance, shown in Algorithm 1. We use the same algorithm for both 2D tracking and 3D tracking.

1.2 Public tracking

For public tracking, we follow Tractor [1] to extend a private tracking algorithm to public detection. The id association is exactly the same as private detection (Line 1 to Line 14). The difference lies in how a track can be created. In public detection, we only initialize a track if it is near a provided bounding box (Line 17 to Line 21).

2 Results on MOT16

MOT16 shares the same training and testing sequences with MOT17, but officially supports private detection. As is shown in Table 1, we rank 2nd among all published entries. We remark that all other entries use a heavy detector trained on private data [16] and many rely on slow matching schemes [13, 16]. For example, LMP_p [13] computes person-reidentification features for all pairs of bounding boxes using a Siamese network, requiring $O(n^2)$ forward passes through a deep network. In contrast, CenterTrack involves a single pass through a network and operates online at 17 FPS.

	Time(ms)	MOTA ↑	IDF1 ↑	FP ↓	FN ↓	IDSW ↓
SORT [2]	36+D	60.4	56.1	11183	59867	1135
DeepSORT [15]	59+D	61.4	62.2	12852	56668	781
POI [16]	100+D	66.1	65.1	5061	55914	805
KNDT [16]	1428+D	68.2	60.0	11479	45605	933
LMP_p [13]	2000+D	**71.0**	**70.1**	**7880**	44564	**434**
Ours (Private)	**57**	69.6	60.7	10458	**42805**	2124

Table 1: Evaluation on the MOT16 test sets (private detection). We compare to all published on the leaderboard. The runtime is calculated from the HZ column on the leaderboard. +D means detection time, which is usually > 100ms [9].

Algorithm 1: Private Detection

Input : $T^{(t-1)} = \{(\mathbf{p}, \mathbf{s}, id)_j^{(t-1)}\}_{j=1}^{M}$: Tracked objects in the previous frame, with center $\mathbf{p}$, size $\mathbf{s} = (w, h)$.
$\hat{B}^{(t)} = \{(\hat{\mathbf{p}}, \hat{\mathbf{d}})_i^{(t)}\}_{i=1}^{N}$: Heatmap peaks with offset $\hat{\mathbf{d}}$ in the current frame, sorted in desending confidence.

Output: $T^{(t)} = \{(\mathbf{p}, \mathbf{s}, id)_i^{(t)}\}_{i=1}^{N}$: Tracked objects in the current frame.

// **Initialization:** $T^{(t)}$ and $\mathcal{S}$ are initialized as empty lists.
$T^{(t)} \leftarrow \emptyset$
$S \leftarrow \emptyset$ // Set of matched tracks
$W \leftarrow Cost(B^{(t)}, T^{(t-1)})$// $W_{ij} = ||\hat{\mathbf{p}}_i^{(t)} - \hat{\mathbf{d}}_i^{(t)}, \mathbf{p}_j^{(t-1)}||_2$

for $i \leftarrow 1$ *to* N **do**
  $j \leftarrow \arg\min_{j \notin S} W_{ij}$
  // calculate the distance threshold κ
  $\kappa \leftarrow \min(\sqrt{\hat{w}_i \hat{h}_i}, \sqrt{w_j h_j})$
  // if the cost is smaller the threshold.
  if $w_{ij} < \kappa$ **then**
    // Propagate matched id
    $T^{(t)} \leftarrow T^{(t)} \cup (\hat{\mathbf{p}}_i^{(t)}, \hat{\mathbf{s}}_i^{(t)}, id_j^{(t-1)})$
    $S \leftarrow S \cup \{j\}$ // Mark track j as matched
  end
  else
    // Create a new track.
    $T^{(t)} \leftarrow T^{(t)} \cup (\hat{\mathbf{p}}_i^{(t)}, \hat{\mathbf{s}}_i^{(t)}, NewId)$
  end
end
Return: $T^{(t)}$

Algorithm 2: Public Detection

Input : $T^{(t-1)} = \{(\mathbf{p}, \mathbf{s}, id)_j^{(t-1)}\}_{j=1}^{M}$: Tracked objects in the previous frame, with center $\mathbf{p}$, size $\mathbf{s} = (w, h)$.
$\hat{B}^{(t)} = \{(\hat{\mathbf{p}}, \hat{\mathbf{d}})_i^{(t)}\}_{i=1}^{N}$: Heatmap peaks with offset $\hat{\mathbf{d}}$ in the current frame, sorted in desending confidence.
$\hat{D}^{(t)} = \{(\mathbf{p}, \mathbf{s})_k^{(t)}\}_{k=1}^{K}$: Public detections.

Output: $T^{(t)} = \{(\mathbf{p}, \mathbf{s}, id)_{i'}^{(t)}\}_{i'=1}^{N'}$: Tracked objects in the current frame.

// **Initialization:** $T^{(t)}$ and $\mathcal{S}$ are initialized as empty lists.
$T^{(t)} \leftarrow \emptyset$
$S \leftarrow \emptyset$ // Set of matched tracks
$W \leftarrow Cost(B^{(t)}, T^{(t-1)})$// $W_{ij} = ||\hat{\mathbf{p}}_i^{(t)} - \hat{\mathbf{d}}_i^{(t)}, \mathbf{p}_j^{(t-1)}||_2$
$W' \leftarrow Cost(B^{(t)}, D^{(t)})$// $W'_{ik} = ||\hat{\mathbf{p}}_i^{(t)}, \mathbf{p}_k^{(t)}||_2$

for $i \leftarrow 1$ *to* N **do**
  $j \leftarrow \arg\min_{j \notin S} W_{ij}$
  // calculate the distance threshold κ
  $\kappa \leftarrow \min(\sqrt{\hat{w}_i \hat{h}_i}, \sqrt{w_j h_j})$
  // if the cost is smaller the threshold.
  if $w_{ij} < \kappa$ **then**
    // Propagate matched id
    $T^{(t)} \leftarrow T^{(t)} \cup (\hat{\mathbf{p}}_i^{(t)}, \hat{\mathbf{s}}_i^{(t)}, id_j^{(t-1)})$
    $S \leftarrow S \cup \{j\}$ // Mark track j as matched
  end
  else
    $k \leftarrow \arg\min_{k=1}^{K} \leftarrow W'_{ik}$
    $\kappa' \leftarrow \min(\sqrt{\hat{w}_i \hat{h}_i}, \sqrt{w_k h_k})$
    if $W'_{ik} < \kappa'$ **then**
      // Create a new track.
      $T^{(t)} \leftarrow T^{(t)} \cup (\hat{\mathbf{p}}_i^{(t)}, \hat{\mathbf{s}}_i^{(t)}, NewId)$
    end
  end
end
Return: $T^{(t)}$

	Modality	mAP ↑	mATE ↓	mASE ↓	mAOE ↓	mAVE ↓	mAAE ↓	NDS ↑
Megvii [18]	LiDAR	52.8	0.300	0.247	0.379	0.245	0.140	63.3
PointPillars [6]	LiDAR	30.5	0.517	0.290	0.500	0.316	0.368	45.3
Mappilary [12]	Camera	30.4	0.738	0.263	0.546	1.	0.134	38.4
CenterNet [17]	Camera	33.8	0.658	0.255	0.629	1.	0.141	40.1

Table 2: 3D detection results on nuScenes test set. We show 3D bounding box mAP, mean translation error (mATE), mean size error (mASE), mean orientation error (mAOE), mean velocity error (mATE), mean attributes error (mAAE), and their weighted (with weight 5 on mAP and 1 on others) average NDS.

3 3D detection

We follow CenterNet [17] to regress to object depth $\hat{D} \in R^{\frac{W}{R} \times \frac{H}{R}}$, 3d extent $\hat{\Gamma} \in R^{\frac{W}{R} \times \frac{H}{R} \times 3}$, orientation (encoded as an 8-dimension vector) $\hat{A} \in R^{\frac{W}{R} \times \frac{H}{R} \times 8}$. The training loss for these are identical to CenterNet [17]. Since the 2D bounding box center does not align with the projected 3D bounding box center due to perspective projection, we in addition regress to an offset from the 2D center to the projected 3D bounding box center$\hat{F} \in R^{\frac{W}{R} \times \frac{H}{R} \times 2}$. We use L1Loss:

$$L_{off3d} = \frac{1}{N} \sum_{k=1}^{N} |\hat{f}_k - f_k|, \tag{1}$$

where $f_k \in \mathcal{R}^2$ is the ground truth offset of object k, and $\hat{f}_k = \hat{F}_{\mathbf{p}_k}$ is the value in $\hat{F}$ at location $\mathbf{p}_k$.

We show the 3D detection performance of CenterNet [17] with the offset prediction in Table 2 for reference. The 3D detection performance is on-par with Mappilary [12] and PointPillars [6], but far below the LiDAR based state-of-the-art Megvii [18].

4 Amodal bounding box regression

CenterNet [17] requires the bounding box center to be within the image. While in MOT [7], the center of the annotated bounding box (Amodal bounding box) can be outside of the image. To accommodate this case, We extend the 2-channel bounding box size head in CenterNet to a 4-channel head $\hat{A} \in \mathbb{R}^{\frac{W}{R} \times \frac{H}{R} \times 4}$ for the distance to the top-, left-, bottom-, right-bounding box border. Note that we still detect the in-frame bounding box center and regress to the in-frame bounding box size. With this 4-dimensional bounding box formulation, the output bounding box is not necessarily centered on the detected center. The training loss for the 4-dimensional bounding box formulation is L1Loss:

$$L_{amodal\_size} = \frac{1}{N} \sum_{i=1}^{N} |\hat{A}_{p_i} - a_i| \tag{2}$$

where $a_i \in \mathbb{R}^4$ is the ground truth border distance.

5 CrowdHuman dataset

CrowdHuman [10]contains 15k training images with common pose annotations. The dataset is featured of high density and large occlusion. Both visible bounding box and the Amodal bounding box are annotated. We use the Amodal bounding box annotation in our experiments to align with MOT [7].

6 Pretraining experiments

For pretraining on CrowdHuman [10], we use input resolution 512×512, false positive ratio $\lambda_{fp} = 0.1$, false negative ratio $\lambda_{fn} = 0.4$, random scaling ratio 0.05, and random translation ratio 0.05. The training follows Section.4.4 of the main paper. As shown in Table 3, the model trained on CrowdHuman achieves a decent 52.2 MOTA in MOT dataset, without seeing any MOT data.

Without CrowdHuman [10] pretraining, our performance drops to 60.7% MOTA on the validation set. Pretraining help improve detection quality by decreasing the false negatives. Note that most entries on MOT challenges use external data for pretraining, and some of them use private data [16]. For reference, we also show our public detection results without pretraining in Table 3, last row. This model corresponds to the entry we submitted to MOT17 public detection challenge.

7 Additional experiments on KITTI

In Table 4, we show results of the same additional experiments (Section. 5.5 of the main paper) on KITTI dataset [4]. The conclusions are the same as on MOT [7]. Training on static images now performs slightly worse than training on video, mostly due to that KITTI has larger inter-frame motion than MOT. Training without random heatmap noise is much worse than the full model, with a high false-negative rate. And using the Hungarian algorithm works the same as using a greedy matching. Our model without nuScenes [3] achieves 84.5% MOTA on the validation set, this is on-par with other state-of-the-art trackers on KITTI [5, 11, 14] with a heavy detector [8].

	MOTA ↑	IDF1 ↑	MT ↑	ML ↓	FP ↓	FN ↓	IDSW ↓
Ours	66.1	64.2	41.3	21.2	4.5%	28.4%	1.0%
only CrowdH.	52.2	53.8	33.6	25.1	6.7%	39.7%	1.4%
scratch	60.7	62.8	33.0	22.4	4.0%	34.2%	1.0%
scratch-Pub.	57.4	59.6	31.1	27.1	2.1%	39.6%	1.0%

Table 3: Additional experiments on the MOT17 validation set. From top to bottom: our full model, the model trained only on CrowdHuman dataset, our model trained from scratch, and the public detection mode of our model trained from scratch.

	MOTA ↑	MOTP ↑	MT ↑	ML ↓	FP ↓	FN ↓	IDSW ↓
Ours	88.7	86.7	90.3	2.1	5.4%	5.8%	0.1%
Static image	86.8	86.5	88.5	2.2	4.8%	7.9%	0.4%
w.o. noisy hm	80.1	85.3	76.2	7.6	3.8%	16.1%	0.1%
Hungarian	88.7	86.7	90.3	2.1	5.4%	5.8%	0.1%
scratch	84.5	83.2	83.4	2.8	5.7%	9.6%	0.3%

Table 4: Additional experiments on the KITTI validation set. From top to bottom: our full model, the public-detection configuration of our model, our model trained on static images only, our model trained without simulating heatmap noise, our model with the Hungarian algorithm used for matching, and our model trained from scratch.

θ	τ	MOTA ↑	IDF1 ↑	MT ↑	ML ↓	FP ↓	FN ↓	IDSW ↓
0.4	0.4	62.6	64.9	44.0	18.9	10.3%	26.4%	0.7%
0.4	0.6	65.5	63.2	38.6	22.4	2.5%	30.5%	1.5%
0.4	0.5	66.1	64.2	41.3	21.2	4.5%	28.4%	1.0%
0.3	0.5	66.2	64.3	43.1	19.2	5.7%	26.9%	1.2%
0.5	0.5	65.2	62.1	39.8	23.0	3.7%	30.2%	0.9%

Table 5: Experiments with different output thresholds (θ) and rendering thresholds (τ) on the MOT [7] validation set. We search θ and τ locally in a step of 0.1.

8 Output and rendering threshold

As the tracking evaluation metric (MOTA) does not consider the confidence of predictions, picking an output threshold is essential in all tracking algorithms (see discussion in AB3D [14]). In our case, we also need a threshold to render predictions to the prior heatmap. We search the optimal thresholds on MOT [7] in Table 5. Basically, increasing both thresholds results in fewer outputs, thus increases the false negatives while decreases the false positives. We find a good balance at $\theta = 0.4$ and $\tau = 0.5$.

References

1. Bergmann, P., Meinhardt, T., Leal-Taixe, L.: Tracking without bells and whistles. In: ICCV (2019)
2. Bewley, A., Ge, Z., Ott, L., Ramos, F., Upcroft, B.: Simple online and realtime tracking. In: ICIP (2016)
3. Caesar, H., Bankiti, V., Lang, A.H., Vora, S., Liong, V.E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., Beijbom, O.: nuScenes: A multimodal dataset for autonomous driving. In: CVPR (2020)
4. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? The KITTI vision benchmark suite. In: CVPR (2012)
5. Hu, H.N., Cai, Q.Z., Wang, D., Lin, J., Sun, M., Krähenbühl, P., Darrell, T., Yu, F.: Joint monocular 3D detection and tracking. In: ICCV (2019)
6. Lang, A.H., Vora, S., Caesar, H., Zhou, L., Yang, J., Beijbom, O.: Pointpillars: Fast encoders for object detection from point clouds. In: CVPR (2019)

7. Milan, A., Leal-Taixé, L., Reid, I., Roth, S., Schindler, K.: MOT16: A benchmark for multi-object tracking. arXiv:1603.00831 (2016)
8. Ren, J., Chen, X., Liu, J., Sun, W., Pang, J., Yan, Q., Tai, Y.W., Xu, L.: Accurate single stage detector using recurrent rolling convolution. In: CVPR (2017)
9. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards real-time object detection with region proposal networks. In: NIPS (2015)
10. Shao, S., Zhao, Z., Li, B., Xiao, T., Yu, G., Zhang, X., Sun, J.: Crowdhuman: A benchmark for detecting human in a crowd. arXiv:1805.00123 (2018)
11. Sharma, S., Ansari, J.A., Murthy, J.K., Krishna, K.M.: Beyond pixels: Leveraging geometry and shape cues for online multi-object tracking. In: ICRA (2018)
12. Simonelli, A., Bulò, S.R.R., Porzi, L., López-Antequera, M., Kontschieder, P.: Disentangling monocular 3d object detection. In: ICCV (2019)
13. Tang, S., Andriluka, M., Andres, B., Schiele, B.: Multiple people tracking by lifted multicut and person re-identification. In: CVPR (2017)
14. Weng, X., Kitani, K.: A baseline for 3d multi-object tracking. arXiv:1907.03961 (2019)
15. Wojke, N., Bewley, A., Paulus, D.: Simple online and realtime tracking with a deep association metric. In: ICIP (2017)
16. Yu, F., Li, W., Li, Q., Liu, Y., Shi, X., Yan, J.: Poi: Multiple object tracking with high performance detection and appearance feature. In: ECCV Workshops (2016)
17. Zhou, X., Wang, D., Krähenbühl, P.: Objects as points. arXiv:1904.07850 (2019)
18. Zhu, B., Jiang, Z., Zhou, X., Li, Z., Yu, G.: Class-balanced grouping and sampling for point cloud 3D object detection. arXiv:1908.09492 (2019)