Supplementary Material for "SpOT: Spatiotemporal Modeling for 3D Object Tracking"

Colton Stearns[1], Davis Rempe[1], Jie Li[2], Rares Ambrus[2], Sergey Zakharov[2], Vitor Guizilini[2], Yanchao Yang[1], and Leonidas J. Guibas[1]

[1] Stanford University
[2] Toyota Research Institute

1 Overview

In this document, we provide additional implementation details, experimental analysis, qualitative results, and discussion. In Sec. 2, we provide further details of our encoding architecture. In Sec. 3, we discuss all implementation details of SpOT not covered in Sec. 4.3 of the main paper. In Sec. 4, we provide metrics definitions for the two benchmarks. Finally, in Sec. 5, we provide a more fine-grained discussion of our method with qualitative results, a reporting of SpOT's performance on the nuScenes test-split, a supplemental analysis of runtime on the nuScenes dataset, and a supplemental ablation study.

2 Additional Architecture Details

Split Self-Attention Positional Encoding We refer the reader to Sec. 3.2 of the main paper for an overview of the split self-attention encoder used by our SSR module. We highlight that our positional encoding differs from previous works that utilize self attention [5]. First, our positional encoding does not utilize a fourier coordinate transformation, *i.e.* there is no $[\sin(x), \cos(x)]$ transformation. Second, we *concatenate*, instead of add, the positional encoding to the anchor features. Experimentally, we find these modifications improve training in our novel 4D setting.

Network Loss Hyperparameters. Sec. 3.2 of the main paper provides an overview of our network's training losses. We set our network loss weights as follows: $w_{\text{c}} = 3.0$, $w_{\theta} = 3.0$, $w_{\text{vel}} = 1.5$, $w_{\text{wlh-cls}} = 1.0$, $w_{\text{wlh-res}} = 1.5$, and $w_{\text{conf}} = 1.0$. We set the confidence-loss temperature to $\alpha = 0.75$ for nuScenes cars, 1.0 for nuScenes pedestrians, 1.2 for Waymo vehicles, and 2.4 for Waymo pedestrians.

3 Additional Implementation Details

3.1 Training-Time Augmentations

Iterative Sequence Refinement. During training, we stochastically update each batch of training sequences multiple times, *i.e.* the network sees its own

output as input. Concretely, for input training sequence $\bar{\mathbf{T}}_t$, we apply our SSR module to generate a refined training tracklet: $\mathbf{SSR}(\bar{\mathbf{T}}_t) = \bar{\mathbf{T}}'_t$. With probability p_{end}, we end the refinement and assign our output $\mathbf{T}_t = \bar{\mathbf{T}}'_t$. Otherwise, we set $\bar{\mathbf{T}}_t \leftarrow \bar{\mathbf{T}}'_t$ and repeat. We limit the maximum number of iterative refinements to 4, and we set $p_{\text{end}} = \min(1 - \frac{\text{EPOCH}}{8}, 0.4)$. We find this iterative strategy noticeably improves training on the Waymo Open dataset. On the nuScenes dataset, we observe little improvement and ultimately leave it out to improve training efficiency.

Training Augmentation. We apply four training sequence augmentations. First, we uniform-randomly drop the leading $[1, K]$ frames of the sequence. Second, we apply a uniform-random rotation, scaling, and reflection to all tracklet bounding boxes and points; we sample rotation between $[-1.57, 1.57]$ radians, sample scaling between $[-5, 5]$ percent, and reflect about the x-axis with probability 0.5. Third, we apply a *single* uniform-random rotation, scaling, and translation to *all* tracklet bounding boxes; we sample translation between $[-0.2, 0.2]$ meters, rotation between $[-0.25, 0.25]$ radians, and scaling between $[-10, 10]$ percent. Finally, we apply *per-frame* uniform-random translations and rotations to each tracklet bounding box; we sample translations between $[-0.1, 0.1]$ meters and rotations between $[-0.1, 0.1]$ radians. We use the same augmentations for all object classes.

3.2 Training Schedule

During training, we use the Adam optimizer [4] with an exponentially decaying learning rate. We set our initial learning rate to 0.0025 and our decay rate to 0.95 per epoch. We train in parallel across 4 Nvidia A100 GPUs and use a global batch size of 300 sequences. We finish training after 10 epochs for pedestrians and 20 epochs for cars/vehicles.

4 Tracking Metrics

4.1 MOTA and MOTP

The Waymo Open Dataset [7] evaluates tracking using the MOTA and MOTP metrics [1]. Multiple Object Tracking Accuracy (MOTA) is defined as:

$$\text{MOTA} = 1 - \frac{\sum_t (\text{MISS}_t + \text{FP}_t + \text{MISMATCH}_t)}{\text{GT}} \tag{1}$$

where MISS_t, FP_t, and MISMATCH_t respectively denote the number of *missed* tracklets, *false positive* tracklets, and *mismatches* at time t. GT denotes the number of all ground-truth tracklets. A mismatch (also denoted identity-switch) occurs when a current tracklet is assigned to a ground-truth object that differs from its previous ground-truth assignment. Thus, MOTA can be decomposed into three equivalent parts: (1) identifying all objects in the frame, (2) not identifying false-positives, and (3) consistently re-identifying objects between frames.

Multiple Object Tracking Precision (MOTP) is defined as:

$$\text{MOTP} = \frac{\sum_{i,t} d_{i,t}}{\sum_t \text{TP}_t} \tag{2}$$

where $d_{i,t}$ denotes the L2 center error of the i'th true-positive tracklet at time t, and $\sum_t \text{TP}_t$ denotes the total number of true-positive tracklets. Thus, MOTP conveys the quality of center estimates for all correctly predicted object tracklets.

4.2 AMOTA and AMOTP

The nuScenes dataset [2] evaluates tracking using the AMOTA and AMOTP metrics [8]. AMOTA and AMOTP address the issue that the highest achievable MOTA often occurs at a low recall; that is, maximizing MOTA often causes tracking methods to *remove* low confidence detections due to their causing an abundance of false-positives and mismatches. Concretely, Average Multiple Object Tracking Accuracy (AMOTA) averages a recall-weighted MOTA over n evenly-spaced recall thresholds:

$$\text{AMOTA} = \frac{1}{n-1} \sum_{r \in \left\{\frac{1}{n-1}, \frac{2}{n-1} \ldots 1\right\}} \text{MOTAR} \tag{3}$$

For a given recall threshold, r, the recall-weighted MOTA metric, MOTAR, is defined as:

$$\text{MOTAR} = \max\left(0, 1 - \frac{\text{MISS}_r + \text{FP}_r + \text{MISMATCH}_r - (1-r) * \text{GT}}{r * \text{GT}}\right) \tag{4}$$

where MISS_r, FP_r, and MISMATCH_r respectively denote the number of missed tracklets, false positive tracklets, and mismatches over *all* times for a recall threshold r. GT denotes the total number of ground-truth tracklets.

Average Multiple Object Tracking Precision (AMOTP) averages MOTP over all recall thresholds, *i.e.*:

$$\text{AMOTP} = \frac{1}{n-1} \sum_{r \in \left\{\frac{1}{n-1}, \frac{2}{n-1}, \ldots, 1\right\}} \text{MOTP}_r \tag{5}$$

4.3 Discussion

While evaluating on AMOTA reflects the *average* MOTA over all recall thresholds, evaluating on MOTA incites selection of the *maximum* MOTA over all recall thresholds. Although correlated, modern tracking algorithms often encounter a substantial tradeoff between the two metrics. For instance, greedy and center-distance association strategies have been shown to be more effective for AMOTA [9,6]. Hungarian-matching and Intersection-Over-Union are more effective for MOTA [8,6]. We highlight this as a concern in LIDAR 3D multi-object tracking: methods often only evaluate one of these metrics and offer no analysis of the other. Contrary to this trend, we showcase SpOT's robustness across both metrics via our evaluation on the nuScenes (AMOTA) and Waymo Open (MOTA) datasets.

Method	*Car* AMOTA↑	*Car* AMOTP↓	*Pedestrian* AMOTA↑	*Pedestrian* AMOTP↓
Centerpoint [9]	82.9	0.384	76.7	0.378
MultimodalTracking [3]	83.0	0.388	74.1	0.507
SimpleTrack-10Hz* [6]	82.3	**0.383**	79.6	**0.364**
SpOT (Ours)	**83.5**	0.390	**80.6**	0.373

Table S1: Tracking results on the nuScenes test split. * denotes a preprint.

5 Additional Experiments

Please refer to Sec. 4 of the main paper for a comprehensive reporting of SpOT's tracking performance and an extensive ablation study of SpOT's design choices.

5.1 SpOT on the nuScenes Test Split

Tab. 1 of the main paper reports the tracking results on the validation split of the nuScenes dataset. In Tab. S1, we report results on the nuScenes leaderboard test split. SpOT outperforms previous methods in correctly tracking objects (AMOTA) and is on-par with previous methods in estimating high quality object tracklets (AMOTP). Note that SimpleTrack [6] uses 10Hz CenterPoint detections while SpOT only uses 2Hz; SimpleTrack reports worse validation-split performance with 2Hz detections.

5.2 Influence of Sequence Length on Efficiency

Sec. 4.5 of the main paper reports the runtime of SpOT's SSR module on an Nvidia RTX3090 GPU for the chosen sequence lengths of 40 on the nuScenes dataset and 10 on the Waymo dataset. In the right panel of Fig. S1, we show how increasing the sequence length affects the SSR module's runtime for nuScenes pedestrians on a single Nvidia V100 GPU. As shown, there is a strong linear relationship between sequence length and the SSR module runtime.

5.3 Further Analysis of Waymo Tracking

Tab. 2 of the main paper reports the tracking results of SpOT on the Waymo Open dataset comparing the state-of-the-arts. In our experiments on the Waymo dataset, we found that tracking performance of some state-of-the-art algorithms tends to be sensitive to the tracklet birth confidence threshold, c_{thresh}, which determines when an unmatched detection will become a tracklet (*e.g.* a lower c_{thresh} allows more unmatched detections to become tracklets). This is not suprising as MOTA focuses more on the high-confidence region of tracking results. For a fair comparison, in Tab. 2 we report results with the optimized c_{thresh} value for each tracking algorithm.

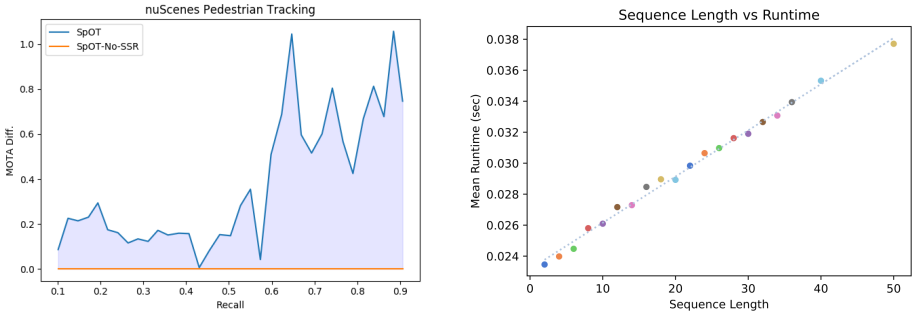

Fig. S1: **Left:** Analysis of pedestrian tracking on the nuScenes dataset. Shown is the difference in MOTA between SpOT and the SpOT-No-SSR baseline for tracklet recall thresholds in [0.10, 0.91]. **Right:** Varying the sequence length on the nuScenes pedestrian class has a linear effect on the SSR module's runtime.

In Tab. S2, we provide ablation analysis on c_{thresh} and show our robustness towards this parameter. We evaluate performance with the original threshold used in CenterPoint, $c_{thresh} = 0.75$, as well as a lower threshold, $c_{thresh} = 0.60$. The lower threshold creates a more challenging setting as more unmatched detections will be treated as tracklets; this creates a more cluttered environment during association. As shown in Tab. S2, our method is able to provide comparable results across both thresholds while CenterPoint's performance is negatively affected by the lower threshold. This example showcases SpOT's robustness against cluttered scenes thanks to the use of dense spatiotemporal information.

Method / Pedestrian	MOTA↑	FP%↓	Miss%↓	Mismatch%↓
	Tracklet Birth Threshold 0.75			
CenterPoint [9]	54.9	10.0	34.0	1.13
SpOT-No-SSR (Ours)	55.8	10.5	33.3	0.38
SpOT (Ours)	**60.4**	**9.5**	**29.8**	**0.34**
	Tracklet Birth Threshold 0.60			
CenterPoint [9]	51.1	9.8	35.2	3.80
SpOT-No-SSR (Ours)	56.5	11.4	31.5	0.61
SpOT (Ours)	**60.5**	**11.3**	**27.6**	**0.56**

Table S2: Tracking performance on the pedestrian class of the Waymo Open dataset validation split. We compare SpOT and CenterPoint with controlled tracklet birth thresholds. Best result in each threshold is bolded.

5.4 Further Analysis of nuScenes Tracking

In Tab. 1 of the main paper, we report the final AMOTA tracking metric of SpOT on the nuScenes dataset. In this section, we offer more fine-grained analysis on nuScenes to better analyze the behavior of our proposed algorithm.

In the left panel of Fig. S1, we visualize the *difference* in the MOTA metric with respect to the SpOT-No-SSR baseline at different recall thresholds. As shown, SpOT consistently improves the tracking results at different recall levels. It's also worth noticing that SpOT improves MOTA disproportionately at higher recall thresholds. This observation furthers the claim that SpOT is robust in cluttered scenes due to the use of dense spatiotemporal information, which is consistent with what we observe in Tab. S2.

In addition, we also provide some qualitative examples showcasing SpOT's improvements in individual tracking scenarios.

In Fig. S2, we provide two illustrative examples of how SpOT's bounding-box refinement improves tracking compared to the SpOT-No-SSR baseline. In the SpOT-No-SSR column of both examples, we observe that poor motion estimates and poor sequence continuity cause tracklet fragmentation and mis-association. In contrast, due to the sequence-to-sequnce refinement, SpOT avoids fragmentation and establishes more accurate tracklets.

In Fig. S3, we provide two illustrative examples of how SpOT's confidence refinement reduces the number of false-positive tracklets. In the SpOT-No-SSR column of both examples, we observe many false-positive tracklets, *i.e.* tracklets with confidence-scores that lie within the visualized recall threshold of 91.1%. After updating tracklet confidence-scores with its sequence-to-sequence refinement, SpOT is able to remove many false-positive tracklets.

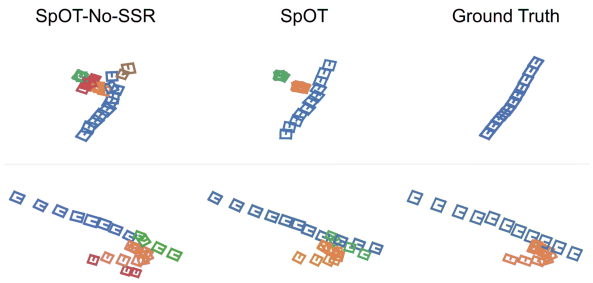

Fig. S2: Two example birds-eye-view visualizations of pedestrians tracked over many frames on the nuScenes dataset. Tracking predictions are colored consistently. SpOT-No-SSR shuffles tracklets, resulting in mismatches and additional false-positives. In contrast, SpOT establishes cleaner sequences via its bounding-box refinement.

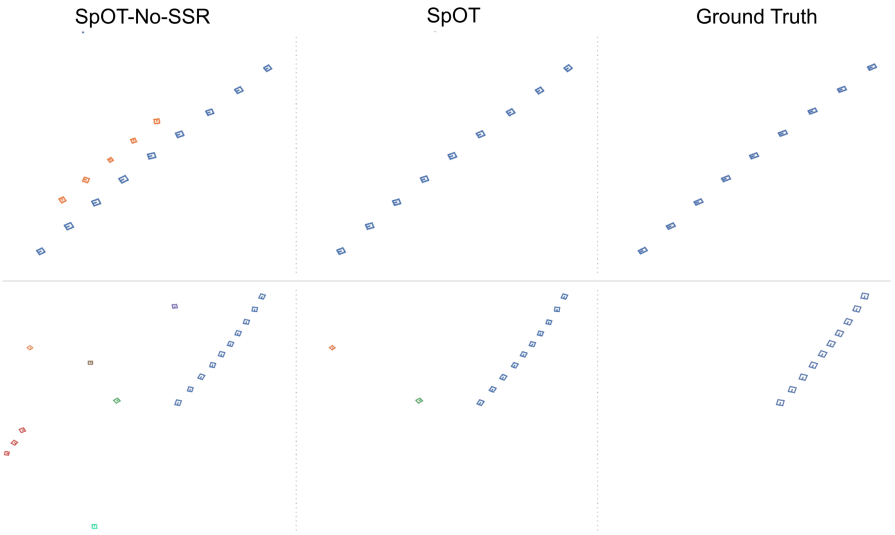

Fig. S3: Two example birds-eye-view visualizations of pedestrian tracking over many frames on the nuScenes dataset. Tracking predictions are colored consistently. For the visualized recall threshold of 91.1%, SpOT's confidence refinement successfully identifies and removes false-positive tracklets.

References

1. Bernardin, K., Stiefelhagen, R.: Evaluating multiple object tracking performance: The clear mot metrics. EURASIP Journal on Image and Video Processing (2008)
2. 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)
3. Chiu, H., Li, J., Ambrus, R., Bohg, J.: Probabilistic 3d multi-modal, multi-object tracking for autonomous driving. CoRR **abs/2012.13755** (2020), https://arxiv.org/abs/2012.13755
4. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. International Conference on Learning Representations (12 2014)
5. Misra, I., Girdhar, R., Joulin, A.: An End-to-End Transformer Model for 3D Object Detection. In: ICCV (2021)
6. Pang, Z., Li, Z., Wang, N.: Simpletrack: Understanding and rethinking 3d multi-object tracking. arXiv preprint arXiv:2111.09621 (2021)
7. Sun, P., Kretzschmar, H., Dotiwalla, X., Chouard, A., Patnaik, V., Tsui, P., Guo, J., Zhou, Y., Chai, Y., Caine, B., et al.: Scalability in perception for autonomous driving: Waymo open dataset. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2446–2454 (2020)
8. Weng, X., Wang, J., Held, D., Kitani, K.: AB3DMOT: A Baseline for 3D Multi-Object Tracking and New Evaluation Metrics. ECCVW (2020)
9. Yin, T., Zhou, X., Krähenbühl, P.: Center-based 3d object detection and tracking. CVPR (2021)