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

Abstract. 3D multi-object tracking aims to uniquely and consistently identify all mobile entities through time. Despite the rich spatiotemporal information available in this setting, current 3D tracking methods primarily rely on abstracted information and limited history, *e.g.* single-frame object bounding boxes. In this work, we develop a holistic representation of traffic scenes that leverages both spatial and temporal information of the actors in the scene. Specifically, we reformulate tracking as a spatiotemporal problem by representing tracked objects as sequences of time-stamped points and bounding boxes over a long temporal history. At each timestamp, we improve the location and motion estimates of our tracked objects through learned refinement over the full sequence of object history. By considering time and space jointly, our representation naturally encodes fundamental physical priors such as object permanence and consistency across time. Our spatiotemporal tracking framework achieves state-of-the-art performance on the Waymo and nuScenes benchmarks.

1 Introduction

3D multi-object tracking (MOT) is an essential task for modern robotic systems designed to operate in the real world. It is a core capability to ensure safe navigation of autonomous platforms in dynamic environments, connecting object detection with downstream tasks such as path-planning and trajectory forecasting. In recent years, new large scale 3D scene understanding datasets of driving scenarios [3,33] have catalyzed research around 3D MOT [35,6,36,12]. Nevertheless, establishing high-fidelity object tracks for this safety-critical application remains a challenge. Notably, recent literature suggests that even small errors in 3D tracking can lead to significant failures in downstream tasks [34,40].

A distinct challenge faced by 3D MOT is that of data association when using LIDAR data as the main source of observation, due to the sparse and irregular scanning patterns inherent in time-of-flight sensors designed for outdoor use. Established works in 2D use appearance-based association [44,1], however, these cannot be directly adapted to 3D MOT. Sensor fusion methods combine camera and LIDAR in an effort to provide appearance-based cues in 3D association [5,36,12]. However, this comes at the cost of additional hardware requirements and increased system complexity.

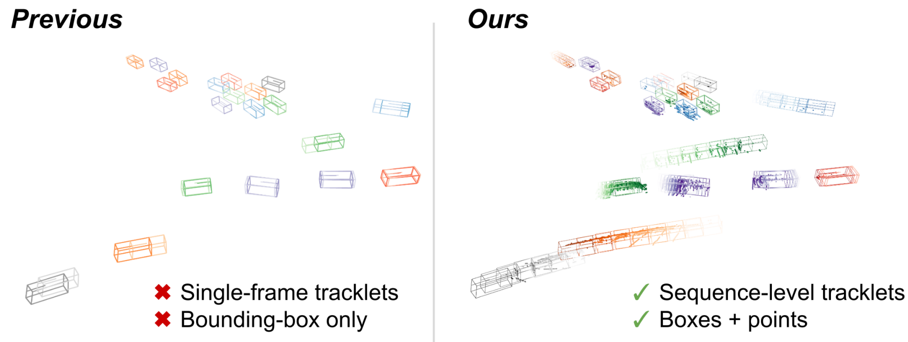

Fig. 1: Previous works use a highly abstracted tracklet representation (*e.g.* bounding boxes) and a compressed motion model (Kalman filter or constant velocity). We efficiently maintain an active history of object-level point clouds and bounding boxes for each tracklet.

Most of the recent works in 3D MOT from LIDAR data address the association problem by matching single-frame tracks to current detection results with close 3D proximity. Single-frame detection results are modeled as bounding boxes [35] or center-points [39] and compared to the same representation of the tracked objects from the last visible frame. Although it touts simplicity, this strategy does not fully leverage the spatiotemporal nature of the 3D tracking problem: temporal context is often over-compressed into a simplified motion model such as a Kalman filter [35,6] or a constant-velocity assumption [39]. Moreover, these approaches largely ignore the low-level information from sensor data in favor of abstracted detection entities, making them vulnerable to crowded scenes and occlusions.

However, improving spatiotemporal context by integrating scene-level LIDAR data over time is challenging due to the large quantity of sampled points along with sparse and irregular scanning patterns. Some methods aggregate LIDAR to improve 3D detection over short time horizons and in static scenes [3,9], as well as over longer time horizons in an *offline* manner [23]. There is also recent work for *single*-object tracking that leverages low-level features in building object representations [38,19], while object-centric 4D canonical representations [28,18,22] have demonstrated the power of spatiotemporal information in object reconstruction. However, these methods are restricted to object-centric datasets, require clean data (*i.e.* low levels of noise), and run on heavy architectures that are not suitable for real time.

In this work, we propose a **spatiotemporal representation** for object tracklets (see Fig. 1). Our method, **SpOT** (**Sp**atiotemporal **O**bject **T**racking), actively maintains the history of *both* object-level point clouds and bounding boxes for each tracked object. At each frame, new object detections are associated with these maintained past sequences, as shown in Fig. 2; the sequences are then updated using a novel 4D backbone to *refine* the entire sequence of bounding boxes and to predict the current velocity, both of which are used to forecast the object into the next frame. This refinement step improves the quality of bounding box and motion estimates by ensuring spatiotemporal consistency, allowing tracklet association to benefit from low-level geometric context over a long time horizon. We perform extensive evaluations on both nuScenes [3] and Waymo Open [33] datasets to demonstrate that maintaining and refining sequences of tracked objects has several advantages, which together enable state-of-the-art tracking performance. Our method is particularly helpful in tracking sparse and occluded objects such as pedestrians, which can especially benefit from temporal priors.

In summary, we contribute: (**i**) a novel tracking algorithm that leverages spatiotemporal object context by storing and updating object bounding boxes and object-level point cloud sequences, (**ii**) a new 4D point cloud architecture for refining object tracks, and (**iii**) state-of-the-art results for 3D multi-object tracking on the standard nuScenes [3] and Waymo [33] benchmark datasets.

2 Related Work

2.1 3D Object Detection on LIDAR Point Clouds

3D detection is one of the most important modules for most 3D tracking frameworks. While 3D detectors from camera data have seen great improvement in recent years [11,15,21,31], LIDAR-based 3D detection offers significantly better performance, especially in driving scenes [7,47,13,8,30,39]. The majority of works on LIDAR-based detection have centered around improving feature extraction from unorganized point clouds. VoxelNet [47] groups points by 3D voxels and extracts voxel-level features using PointNet [25]. PointPillar [13] organizes point clouds into vertical columns (pillars) to achieve higher efficiency. PV-RCNN [30] aggregates voxel-level and point-level features to achieve better accuracy. On the other hand, CenterPoint [39] improves the 3D detector by looking at the output representation, proposing a point-based object representation at the decoding stage. While our proposed approach is not constrained to a specific input detector, we use CenterPoint in our experiments due to its popularity and to facilitate comparison with other state-of-the-art tracking algorithms.

In addition to improving the 3D detector architecture, aggregating temporal information has also been shown to improve 3D detection results as temporal information compensates for the sparsity of LIDAR sensor inputs. nuScenes [3] uses a simple motion compensation scheme to accumulate multiple LIDAR sweeps, providing a richer point cloud with an added temporal dimension. In addition, multiple detector architectures [13,48] use accumulated point clouds to improve performance and reason over object visibility [9]. Nevertheless, limited by the static scene assumption of motion compensation, temporal aggregation for the detection task is primarily done over short intervals.

Recently in offline perception, Qi et al. [23] address the use of a longer time horizon as a post-processing step. After running an offline detection and tracking algorithm, Qi et al. apply a spatiotemporal sliding window refinement on each tracked object at each frame.

In our work, we explore the utilization of temporal information over a longer time horizon in the task of multi-object tracking. Distinct from Qi et al., we utilize a long-term time horizon *within* our tracking algorithm, we operate in the noisier online setting, and we maintain and refine full object sequences.

2.2 3D Multi-Object Tracking

Thanks to the advances in 3D object detection discussed above, most state-of-the-art 3D MOT algorithms follow the *tracking-by-detection* paradigm. The performance of 3D MOT algorithms is mainly affected by three factors other than detection results: the motion prediction model, the association metric, and the life-cycle management of the tracklets.

CenterPoint [39] proposes a simple, yet effective, approach that gives reliable detection results and estimates velocities to propagate detections between sequential frames. The distance between object centers is used as the association metric. However, CenterPoint's constant velocity can be less robust to missing detections and long-term occlusions (as we demonstrate later in Fig. 4).

The most popular category of 3D MOT algorithms leverages Kalman Filters to estimate the location of tracked objects and their dynamics, providing predictions for future association. AB3DMOT [35] provides the prior baseline in this direction and leverages 3D Intersection-over-Union (IoU) as the association metric. Following this line, Chiu et al. [6] proposes to replace 3D IoU with Mahalanobis distance to better capture the uncertainty of the tracklets. SimpleTrack [20] conducts an analysis on different components of a tracking-by-detection pipeline and proposes corresponding enhancements to different modules.

Other works jointly train detection and tracking in a more data-driven fashion. FaF [16] proposes to jointly solve detection, tracking, and prediction using a multi-frame architecture on a voxel representation. However, the architecture is hard to scale up to a long history. PnPNet [14] extends the idea of FaF and proposes a more general framework with an explicit tracking model. Zaech et al. [41] combines detection and association in a graph structure and employs neural message passing.

Another line of work explores feature learning for data association in sensor fusion scenarios [36,5,43,12]. In this work, we focus on the application scenario of the LIDAR sensor only.

2.3 3D Single-Object Tracking

Given an initial *template* ground-truth bounding box of an object, the goal of single-object tracking (SOT) is to track the template object through all future frames. Unlike MOT, it is common for SOT methods to aggregate object-level information over a large temporal interval.

Many works use a Siamese network to compare the template encoding with a surrounding region of interest. P2B [27] uses a PointNet++ to directly propose

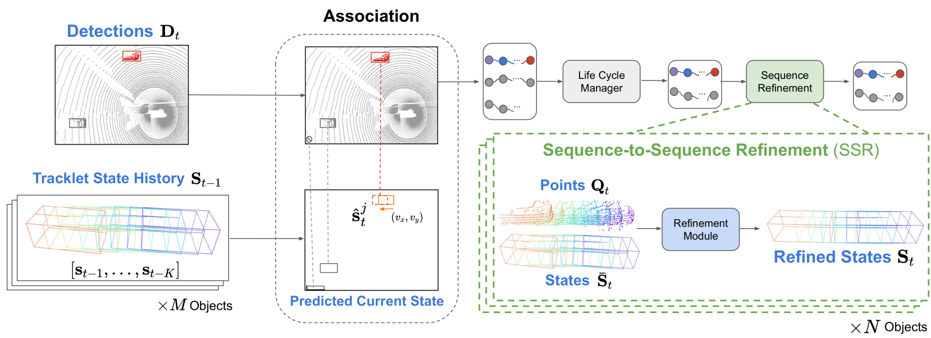

Fig. 2: Tracking algorithm overview. SpOT maintains sequence-level tracklets containing both object bounding box states and object point clouds from the last K timesteps. At each step, tracklets are associated to current detections using the predicted current state, and then the updated tracklets are refined by the learned SSR module to improve spatiotemporal consistency.

seeds within the surrounding region and avoid an exhaustive search. BAT [45] improves the template object representation with a box-aware coordinate space. PTTR [46] uses cross-attention to improve feature comparison.

Recent works propose SOT without direct supervision. Pang et. al [19] perform template matching by optimizing hand-crafted shape and motion terms. Ye et. al [38] extend the work of Pang et. al with a deep SDF matching term. In contrast to SOT methods, we operate in the MOT setting on sequences originally generated from an imperfect 3D detector, and we maintain object sequence histories to avoid propagating error over time.

3 Multi-Object Tracking using Sequence Refinement

In this section, we provide an overview of our basic notation and SpOT tracking pipeline in Sec. 3.1. We then introduce our novel spatiotemporal sequence refinement module in detail in Sec. 3.2.

3.1 Tracking Pipeline

In this work, we address the problem of 3D multi-object tracking (MOT) from LIDAR sensor input. The goal of this task is to uniquely and consistently identify every object in the scene in the form of tracklets $\mathbf{O}_t = \{\mathbf{T}_t\}$ at each frame of input t. As input, the current LIDAR point cloud $\mathbf{P}_t$ is given along with detection results $\mathbf{D}_t = \{\mathbf{b}_t\}$, in the form of 7-DoF amodal bounding boxes $\mathbf{b}_i = (x, y, z, l, w, h, \theta)_i$ and confidences s_i, from a given detector. In contrast to previous works that maintain only single-frame tracklets, we model our tracklet as $\mathbf{T}_t = \{\mathbf{S}_t, \mathbf{Q}_t\}$ to include both a low-level history of *object* points $\mathbf{Q}_t$ and the corresponding sequence of detections $\mathbf{S}_t$. In each tracklet, $\mathbf{S}_t$ includes the estimated state trajectory of the tracked object within a history window:

$$\mathbf{S}_t = \{\mathbf{s}_i = (\mathbf{b}, v_x, v_y, c, s)_i\}, \quad t - K \le i \le t \tag{1}$$

where K is a pre-defined length of maximum history. Tracklet state has 11 elements and includes a 7-DoF bounding box $\mathbf{b}$, a birds-eye-view velocity (v_x, v_y), an object class c (*e.g.* "car"), and a confidence score $s \in [0, 1]$. On the other hand, $\mathbf{Q}_t$ encodes the spatiotemporal information from raw sensor observations in the form of time-stamped points:

$$\mathbf{Q}_t = \{\hat{\mathbf{P}}_i = \{(x, y, z, i)\}\}, \quad t - K \le i \le t \tag{2}$$

where $\hat{\mathbf{P}}_i$ is the cropped point cloud region from $\mathbf{P}_i$ according to the associated detected bounding box $\mathbf{b}_i$ at time i. We enlarge the cropping region by a factor of 1.25 to ensure that $\hat{\mathbf{P}}_i$ is robust through imperfect detection results.

As depicted in Fig. 2, we propose a tracking framework that follows the tracking-by-detection paradigm while leveraging low-level sensory information. At each timestep t, we first predict the current tracklets $\hat{\mathbf{T}}_t$ based on the stored previous ones $\mathbf{T}_{t-1}$:

$$\hat{\mathbf{T}}_t = \mathbf{Predict}(\mathbf{T}_{t-1}) = \{\hat{\mathbf{S}}_t, \mathbf{Q}_{t-1}\} \tag{3}$$

$$\hat{\mathbf{S}}_t = \{\hat{\mathbf{s}}_i = (x + v_x, y + v_y, z, l, w, h, \theta, v_x, v_y, c, s)_{i-1}\}, \quad t - K \le i \le t. \tag{4}$$

We compare the *last* state of the predicted tracklets $\hat{\mathbf{s}}_t$ to off-the-shelf detection results $\mathbf{D}_t$ to arrive at associated tracklets:

$$\bar{\mathbf{T}}_t = \mathbf{Association}(\hat{\mathbf{S}}_t, \mathbf{D}_t, \mathbf{P}_t) = \{\bar{\mathbf{S}}_t, \mathbf{Q}_t\} \tag{5}$$

where $\bar{\mathbf{S}}_t$ is the previous state history $\mathbf{S}_{t-1}$ concatenated with its associated detection and $\mathbf{Q}_t$ is the updated spatiotemporal history. Without loss of generality, we follow CenterPoint's [39] association strategy in our experiments. Finally, we conduct the posterior tracklet update using a novel sequence-to-sequence refinement (SSR) module:

$$\mathbf{S}_t = \mathbf{SSR}(\bar{\mathbf{S}}_t, \mathbf{Q}_t), \tag{6}$$

which provides the final updated tracklet estimation $\mathbf{T}_t = \{\mathbf{S}_t, \mathbf{Q}_t\}$. In the following section, we will provide technical details of the SSR module.

3.2 Sequence-to-Sequence Refinement (SSR) Module

We propose a novel algorithm to update a full tracklet history of estimated states by accounting for its spatiotemporal context, including raw sensor observations. Fig. 3 displays our spatiotemporal sequence-to-sequence refinement (SSR) module, which takes the *associated* tracklet states $\bar{\mathbf{S}}_t$ and the time-stamped object point cloud segments $\mathbf{Q}_t$ as input and outputs refined final tracklet states $\mathbf{S}_t$.

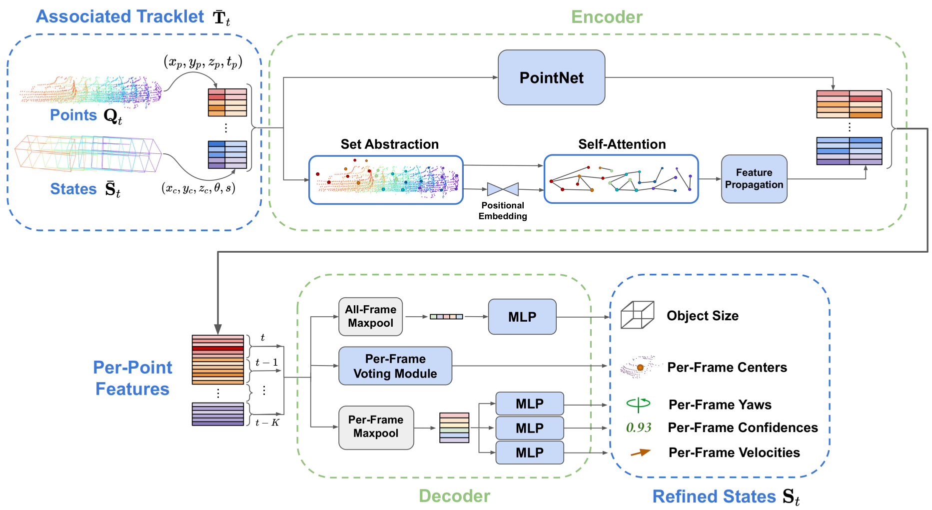

Fig. 3: Architecture of the sequence-to-sequence refinement (SSR) network. Given a tracklet containing object points and bounding boxes after association to a detection, the encoder first extracts spatiotemporal features corresponding to each input point. The features are given to the decoder which predicts a refined state trajectory and velocities to be used for subsequent association.

SSR first processes the sequential information with a 4D backbone to extract per-point context features. In the decoding stage, it predicts a global object size across all frames, as well as per-frame time-relevant object attributes including center, pose, velocity, and confidence.

Split Self-Attention Encoder. The top part of Fig. 3 illustrates the encoding backbone, which processes each associated tracklet independently. Since the inputs contain two streams of information $\bar{\mathbf{S}}_t$, $\mathbf{Q}_t$, which are at different levels of abstraction (object vs. point), we first append the bounding-box-level information as an additional dimension to each point in $\mathbf{Q}_t$. This yields a set of object-aware features:

$$\mathbf{f}_p = [x_p, y_p, z_p, t_p, x_c, y_c, z_c, \sin(\theta), \cos(\theta), s], \tag{7}$$

where (x_p, y_p, z_p, t_p) denotes the 4D geometric point and $(x_c, y_c, z_c, \theta, s)$ is the center location, yaw, and confidence score of the corresponding bounding box at frame t_p.

Similar to previous works on spatiotemporal representation learning [28], the encoder is a two-branch point cloud backbone as depicted in Fig. 3. In the first branch, we apply a PointNet [25] to directly encode the high-dimensional inputs into per-point features. For the second branch, we apply a novel self-attention architecture inspired by the encoder of 3Detr [17]. First, we apply a per-frame PointNet++ [26] set abstraction layer, so that at each frame i we have a subsampled set of anchor-point features $\{\mathbf{a}_i^k\}_{k=1}^A$ where A is a hyperparameter for the number of anchor points. For each anchor point, a 4D positional embedding is generated using a 3-layer multi-layer perceptron (MLP):

$$\mathbf{pos}_{\mathbf{a}_i^k} = \mathbf{MLP}(\mathbf{a}_i^k). \tag{8}$$

The anchor features and positional embedding are concatenated as $[\mathbf{a}_i^k, \mathbf{pos}_{\mathbf{a}_i^k}]$ before applying four layers of self-attention across *all* anchor features. Notably, this self-attention allows information flow across both space and time. Finally, updated anchor features are propagated back to the full resolution point cloud via a feature propagation layer [26]. Layer normalization is applied to the features from each branch before concatenating to get the final per-point features.

Each branch of the encoder uses a 256-dim feature, yielding a concatenated 512-dim feature at the output. Set abstraction uses $A = 10$ anchor points per frame and a feature radius of $1.5m$ for cars/vehicles and $0.6m$ for pedestrians. Additional architectural details are provided in the supplemental material.

Sequence Decoder. The SSR decoder outputs a refined, ordered sequence of object states $\mathbf{S}_t$ that is amenable to association in subsequent frames. To output object state trajectories, some recent works use explicit priors on temporal continuity, such as anchor-based trajectories [4] or an autoregressive motion rollout [29]. In contrast, we choose a decoder without an explicit prior: the decoder directly predicts the ordered sequence of bounding boxes in one forward pass. This choice allows the model to learn temporal priors where needed through training. Our design is motivated by the discontinuous nature of many sequences that SSR operates on, which contain identity switches, false-positives, and occlusions.

As depicted in the bottom portion of Fig. 3, given an encoded set of per-point spatiotemporal features, we group features by their time of acquisition (*i.e.* by frame). We pass our time-grouped point features into 5 decoding heads. The first decoding head performs a max-pool on the entire feature set to regress a single object size (l, w, h), which is used for every output frame. The second head applies a voting module [24] to each set of time-grouped features; this outputs per-timestep object center predictions (x_c, y_c, z_c). The remaining heads perform a max-pool on each set of time-grouped features to obtain a single feature per timestep. This feature is passed through 2-layer MLPs to regress a yaw, confidence, and velocity (θ, s, v_x, v_y) for each frame.

Training Losses. Our sequence refinement module balances two loss terms: a bounding-box loss and a confidence-score loss. Our total loss is as follows:

$$L = w_{\text{conf}} L_{\text{conf}} + L_{\text{box}}, \tag{9}$$

where w_{conf} is a hyperparameter that balances the two losses.

We formulate our bounding-box loss similar to standard 3D detection works [39,10,13,23,37]. We apply an L1 loss on 3D box center $[x, y, z]$. We apply a cross-entropy loss on the predicted size bin and an L1 loss on the predicted size residual. We apply an L1 loss on the polar angle representation $\sin(\theta), \cos(\theta)$. This yields a bounding box loss of:

$$L_{\text{box}} = w_{\text{c}} L_{\text{c}} + w_{\theta} L_{\theta} + w_{\text{vel}} L_{\text{vel}} + w_{\text{wlh-cls}} L_{\text{wlh-cls}} + w_{\text{wlh-res}} L_{\text{wlh-res}}, \tag{10}$$

where w_{c}, w_{θ}, w_{vel}, $w_{\text{wlh-cls}}$, and $w_{\text{wlh-res}}$ balance losses for the bounding box center, yaw, velocity, size-bin, and size-residual, respectively.

We desire the model's predicted confidence to match the prediction's quality. To achieve this, we set a target confidence score $\bar{s}$ in a manner proportional to the accuracy of the bounding-box estimate. Concretely, if a bounding box is not close to a ground-truth object, we assign the target confidence to 0. Otherwise, we follow [32] and assign the target confidence to be proportional to the L2 distance from the closest ground-truth object as $\bar{s} = e^{-\alpha \mathbf{b_{err}}}$, where α is a temperature hyperparameter and $\mathbf{b_{err}}$ is the L2 box center error. Our confidence loss is then a binary cross-entropy loss, $L_{\text{conf}} = \text{BCE}(s, \bar{s})$.

Training Data. During online tracking, the refinement module must robustly handle noisy inputs from the 3D detector, which may contain false-positives, identity switches, occlusions, and more. Therefore, we must use a set of suitable training sequences that faithfully capture these challenging test-time phenomena. To achieve this, we use the outputs of previous tracking methods to generate object tracks that are used as training sequences. For all experiments, we generate data using the CenterPoint [39] tracker with varied track-birth confidence threshold, $c_{thresh} \in \{0.0, 0.3, 0.45, 0.6\}$, and varied track-kill age, $t_{kill} \in \{1, 2, 3\}$. We additionally augment these tracks with transformations, noise, and random frame dropping. Our final training set averages 750k sequences per object class. These augmentation methods are detailed in the supplementary material.

4 Experimental Evaluation

4.1 Datasets and Evaluation Metrics

nuScenes Dataset. The nuScenes dataset contains 1000 sequences of driving data, each 20 seconds in length. 32-beam LIDAR data is provided at 20Hz, but 3D labels are only given at 2Hz. The relatively sparse LIDAR data and low temporal sampling rate make our proposed method particularly suitable for data like that in nuScenes, where leveraging spatiotemporal history provides much-needed additional context. We follow the official nuScenes benchmark protocol for tracking, which uses the AMOTA and AMOTP metrics [35]. For a thorough definition of AMOTA and AMOTP, we refer the reader to the supplementary material. We evaluate on the two most observed classes: car and pedestrian.

Waymo Open Dataset. The Waymo Open Dataset [33] contains 1150 sequences, each with 20 seconds of contiguous driving data. Unlike nuScenes, the Waymo Open Dataset provides sensor data for four short-range LIDARs and one long-range LIDAR. Each LIDAR is sampled at 10Hz, and the long-range LIDAR is significantly denser than the nuScenes 32-beam device. 3D labels are provided for every frame at 10Hz. We follow official Waymo Open Dataset benchmark protocol, which uses MOTA and MOTP [2] to evaluate tracking. For a thorough definition of MOTA and MOTP, we refer the reader to the supplementary material. We evaluate on the two most observed classes: vehicle and pedestrian.

Note that AMOTA averages MOTA at different recall thresholds. In our experiments, different sets of parameters were used between nuScenes and Waymo. For nuScenes, lower thresholds were used to balance the recall.

4.2 Implementation Details

In this section, we highlight the most important implementation details, and refer the reader to the supplementary material for additional information.

SSR Training Details. We train a different network for each object class using sequences of length $K = 40$ for nuScenes and $K = 15$ for Waymo (we investigate the effect sequence length has on tracking performance in Tab. 3). To improve robustness and mimic test time when stored tracklets are iteratively refined each time a new frame is observed, during training we refine a sequence a random number of times before backpropagation, *i.e.* the network sees its own output as input. In practice, we find that *not* refining bounding-box size is beneficial for Waymo vehicles, due to the large variance in sizes.

Test-Time Tracking Details. Similar to prior works [5,42,20], we use off-the-shelf detections from CenterPoint [39] as input at each frame of tracking. For nuScenes, CenterPoint provides detections at 2Hz so we upsample to 20Hz by backtracking the estimated velocities to match the LIDAR sampling rate. CenterPoint detections are pre-processed with a birds-eye-view non-maximal-suppression using thresholds of 0.3 IoU for nuScenes and 0.5 IoU for Waymo.

Detections are associated with the last frame of object tracklets using a greedy bipartite matching algorithm over L2 center-distances that uses detection confidence [39]. We set a maximum matching distance of $1m$ and $4m$ for nuScenes pedestrians and cars, respectively, and $0.4m$ and $0.8m$ for Waymo Open pedestrians and vehicles. We use a track-birth confidence threshold of 0.0 for nuScenes. For Waymo Open, this is 0.6 for pedestrians and 0.7 for vehicles. The track-kill age is 3 for both datasets. For nuScenes, we start refining tracklets at a minimum age of 30 frames with a maximum temporal context of 40 frames.

	Car		*Pedestrian*	
Method	AMOTA↑	AMOTP↓	AMOTA↑	AMOTP↓
AB3DMOT [35]	72.5	0.638	58.1	0.769
Centerpoint [39]	84.2	**0.380**	77.3	0.392
ProbabalisticTracking [6]	84.2	–	75.2	–
MultimodalTracking [5]	84.3	–	76.6	–
SimpleTrack-2Hz* [20]	83.8	0.396	79.4	0.418
SpOT-No-SSR (Ours)	84.5	**0.380**	81.1	0.391
SpOT (Ours)	**85.1**	0.390	**82.5**	**0.386**

Table 1: Tracking performance on the nuScenes dataset validation split. An asterisk* denotes a preprint.

Method	MOTA↑	FP%↓	Miss%↓	Mismatch%↓
		Vehicle		
AB3DMOT [35]	55.7	–	–	0.40
CenterPoint [39]	55.1	10.8	33.9	0.26
SimpleTrack* [20]	**56.1**	**10.4**	33.4	**0.08**
SpOT-No-SSR (Ours)	55.1	10.8	33.9	0.21
SpOT (Ours)	55.7	11.0	**33.2**	0.18
		Pedestrian		
AB3DMOT [35]	52.2	–	–	2.74
CenterPoint [39]	54.9	**10.0**	34.0	1.13
SimpleTrack* [20]	57.8	10.9	30.9	**0.42**
SpOT-No-SSR (Ours)	56.5	11.4	31.5	0.61
SpOT (Ours)	**60.5**	11.3	**27.6**	0.56

Table 2: Tracking performance on the Waymo Open dataset validation split. An asterisk* denotes a preprint.

For Waymo, the minimum refinement age is 5 for vehicles and 2 for pedestrians and the maximum temporal context is 10 frames.

As discussed in Sec. 3.1 the tracklet state trajectory $\mathbf{S}_t$ stores the history of bounding boxes for each object. On nuScenes, these boxes are the output of our refinement network such that all boxes continue to be refined at each new frame. On Waymo, we instead directly store the given CenterPoint detections.

4.3 Comparison with State-of-the-Art Tracking

In this section, all reported tracking results are obtained with CenterPoint detections [39]. We build our tracking pipeline based on CenterPoint and adopt NMS pre-processing to detection results as suggested in SimpleTrack [20]. We denote this baseline version of CenterPoint tracking with NMS pre-processing as *SpOT-No-SSR* and report its performance for a fair comparison.

In Tab. 1, we compare SpOT to various tracking methods on the nuScenes dataset. SpOT significantly outperforms all previous methods in correctly tracking objects (AMOTA) and is on-par with previous methods in estimating high-quality object tracklets (AMOTP). In Tab. 2, we compare SpOT to various tracking methods on the Waymo Open dataset. For pedestrians, SpOT significantly outperforms all previous methods. For vehicles, SpOT notably improves tracking over the CenterPoint baseline. Examining metric breakdowns, it becomes clear that SpOT is able to robustly track objects in more cluttered environments compared to previous methods. That is, we remove fewer detections in pre-processing to yield fewer *Misses*, yet we still maintain a very low *Mismatch* score. Additionally, we note that many contributions of the competitive method SimpleTrack [20], such as a 2-stage association strategy and a generalized-IoU similarity metric, could be seamlessly integrated into SpOT.

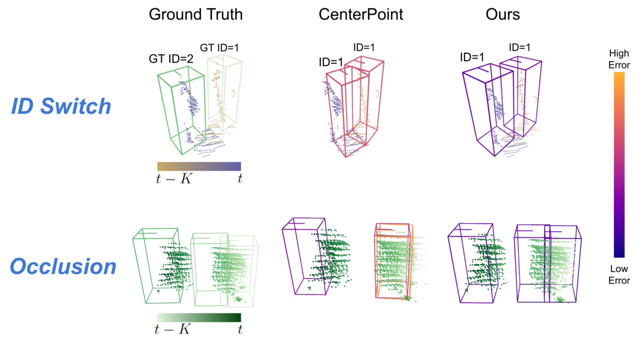

Fig. 4: Examples of discontinuous object tracklets. We visualize every 10th prediction for clarity, and predicted boxes are colored according to L2 center error. Our refinement is robust to different types of input sequence discontinuities. In the first row, our refinement immediately recovers after an identity-switch by correctly updating the bounding boxes to reflect the existence of two disjoint objects. In the second row, it correctly updates bounding boxes to reflect single-object continuity through occlusion.

The SSR module of SpOT exhibits greater improvements on the nuScenes dataset and on the pedestrian class in general. This is unsurprising because sparser LIDAR frames and smaller objects are expected to benefit disproportionately from increased temporal context. Fig. 5 illustrates examples of our refined sequences compared to tracklets composed of off-the-shelf CenterPoint detections. We observe the greatest improvement in sequence quality when individual frames are sparse. Furthermore, we can qualitatively observe improved temporal consistency within sequences.

Additionally, we observe that our SSR module can handle noisy input detections and/or associations by learning *when* to make use of physical priors on object permanence and consistency. We provide some examples illustrating this property in Fig. 4. The first row displays an example when both CenterPoint and SpOT encounter an ID-switch error in the tracklet. For CenterPoint, this error will be propagated to future prediction and association. For SpOT, even though we can not retroactively correct the misassociation, the SSR module still refines the sequence bounding boxes in a manner that accurately reflects two disjoint objects; this accurate update will help to avoid future tracking errors. The second row shows a discontinuous sequence due to *occlusion* where different parts of an object are observed. Our SSR module refines the occluded region in a manner that reflects temporal continuity of a single object.

4.4 Ablative Analysis

Length of Maximum History. Tab. 3 reports how the length of maximum history affects tracking performance. The table emphasizes the significant advan-

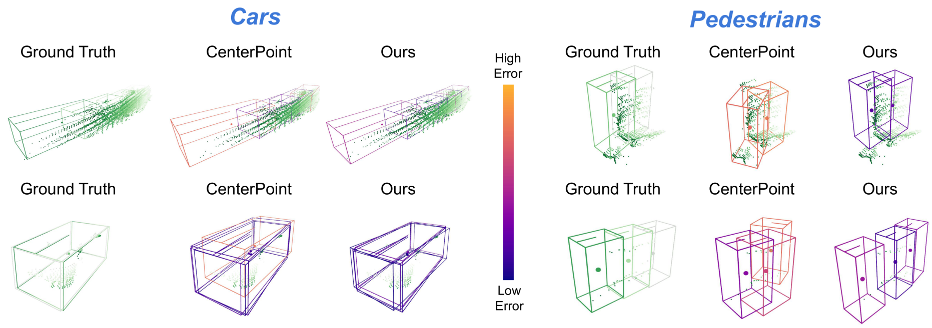

Fig. 5: Qualitative results of our spatiotemporal sequence refinement. We visualize every 10th prediction for clarity. Predicted bounding boxes are colored according to their L2 center error. Refinement improves the temporal consistency of detections, especially for sparse sequences.

tage of using a large history. Because nuScenes is sampled at 20Hz and uses a sparse 32-beam LIDAR, we observe tracking performance monotonically improve up to a 40-frame history (2 seconds). In contrast, Waymo tracking performance peaks at a 10-frame history (1 second) and declines beyond 10 frames.

SSR Update Components. Recall that for each object tracklet, our SSR module predicts per-timestep refinements consisting of a bounding box (with velocity) and a confidence score. Tab. 4a displays an ablation study comparing these two SSR refinements. All reported values are the AMOTA metric on the nuScenes dataset. As observed, both bounding box and confidence refinements contribute to tracking, and we achieve the best performance when we refine both.

SSR Backbone Architecture. Tab. 4b displays an ablation analysis of our SSR backbone on the nuScenes dataset. All reported values are the AMOTA metric. The first row shows tracking metrics with only a PointNet backbone. The second row corresponds to a two-branch backbone where the second branch consists of set abstraction and feature propagation. The third row corresponds to our full backbone. As observed, each part of our backbone improves refinement.

4.5 Runtime Analysis

All components of SpOT except our SSR module are benchmarked in previous real-time tracking algorithms [20,39,35]. We benchmark the real-time performance of our SSR module on an Nvidia RTX3090 GPU. On the nuScenes dataset, our SSR module averages at 51Hz for pedestrians and 28Hz for cars. On the Waymo dataset, it averages at 26Hz for pedestrian and 17Hz for vehicles.

5 Discussion

We have introduced SpOT, a method for 3D multi-object tracking in LIDAR data that leverages a spatiotemporal tracklet representation in the form of ob-

nuScenes (20Hz): AMOTA↑			**Waymo** (10Hz): MOTA↑		
Num Sweeps	*Car*	*Pedestrian*	Num Sweeps	*Vehicle*	*Pedestrian*
10	84.7	81.1	1	53.4	58.4
20	85.0	81.4	5	54.6	59.8
30	85.0	81.7	10	**55.7**	**60.4**
40	**85.1**	**82.5**	15	54.5	60.2

Table 3: Ablation on how the length of maximum history (number of input LIDAR sweeps to SSR) affects the quality of tracking. Tracking performance peaks at a 2 second history for nuScenes and a 1 second history for Waymo.

SSR Refinement Components

Refinement Type	*Car*	*Pedestrian*
No Refinement	84.5	81.1
Boxes Only	84.9	81.9
Confidences Only	84.9	81.6
Boxes and Confidences	**85.1**	**82.5**

SSR Encoder

Architecture	*Car*	*Pedestrian*
PointNet only	84.8	81.6
+ SA and FP	84.8	81.7
+ Self-attention	**85.1**	**82.5**

Table 4: Ablation experiments on the nuScenes dataset. All reported values are the AMOTA tracking metric. (a) Ablation holding out box and confidence-score refinements of our SSR module. (b) Ablation holding out parts of our refinement backbone. We denote set abstraction as SA and feature propagation as FP.

ject bounding boxes and point cloud history. Furthermore, we have proposed a 4D refinement network to iteratively update stored object sequences after associating new detections at each frame. Through evaluations on standard tracking benchmarks, SpOT compares favorably to prior works that use only single-frame tracks, thanks to the ability to leverage larger spatiotemporal context and use low-level geometry cues to improve bounding box and motion estimates. Our method particularly excels given longer temporal history and when operating on pedestrians due to naturally increased occlusions and sparsity.

Although our results indicate a promising first step to improving 3D tracking with spatiotemporal representations, we believe there are many future directions to explore, such as building end-to-end association in lieu of a 2-stage strategy and more explicitly utilizing aggregated tracklet geometry for downstream tasks.

Acknowledgments This work was supported by grants from the Toyota Research Institute (TRI) University 2.0 program, a Vannevar Bush Faculty Fellowship, and a gift from the Amazon Research Awards program. Toyota Research Institute ("TRI") provided funds to assist the authors with their research but this article solely reflects the opinions and conclusions of its authors and not TRI or any other Toyota entity.

References

1. Bergmann, P., Meinhardt, T., Leal-Taixé, L.: Tracking without bells and whistles. In: The IEEE International Conference on Computer Vision (ICCV) (October 2019)
2. Bernardin, K., Stiefelhagen, R.: Evaluating multiple object tracking performance: The clear mot metrics. EURASIP Journal on Image and Video Processing (2008)
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. Chai, Y., Sapp, B., Bansal, M., Anguelov, D.: Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction. CoRR **abs/1910.05449** (2019), http://arxiv.org/abs/1910.05449
5. 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
6. Chiu, H., Prioletti, A., Li, J., Bohg, J.: Probabilistic 3d multi-object tracking for autonomous driving. 2021 IEEE International Conference on Robotics and Automation (ICRA) (2020), https://arxiv.org/abs/2001.05673
7. Engelcke, M., Rao, D., Wang, D.Z., Tong, C.H., Posner, I.: Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks. In: 2017 IEEE International Conference on Robotics and Automation (ICRA). pp. 1355–1361. IEEE (2017)
8. Ge, R., Ding, Z., Hu, Y., Wang, Y., Chen, S., Huang, L., Li, Y.: Afdet: Anchor free one stage 3d object detection. CoRR **abs/2006.12671** (2020), https://arxiv.org/abs/2006.12671
9. Hu, P., Ziglar, J., Held, D., Ramanan, D.: What you see is what you get: Exploiting visibility for 3d object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11001–11009 (2020)
10. Huang, R., Zhang, W., Kundu, A., Pantofaru, C., Ross, D.A., Funkhouser, T., Fathi, A.: An lstm approach to temporal 3d object detection in lidar point clouds. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.M. (eds.) Computer Vision – ECCV 2020. pp. 266–282. Springer International Publishing, Cham (2020)
11. Kehl, W., Manhardt, F., Tombari, F., Ilic, S., Navab, N.: Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. In: Proceedings of the IEEE international conference on computer vision. pp. 1521–1529 (2017)
12. Kim, A., Ošep, A., Leal-Taixé, L.: Eagermot: Real-time 3d multi-object tracking and segmentation via sensor fusion. In: CVPR-Workshops. vol. 1, p. 3 (2020)
13. Lang, A.H., Vora, S., Caesar, H., Zhou, L., Yang, J., Beijbom, O.: Pointpillars: Fast encoders for object detection from point clouds. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 12689–12697 (2019). https://doi.org/10.1109/CVPR.2019.01298
14. Liang, M., Yang, B., Zeng, W., Chen, Y., Hu, R., Casas, S., Urtasun, R.: Pnpnet: End-to-end perception and prediction with tracking in the loop. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 11550–11559 (2020). https://doi.org/10.1109/CVPR42600.2020.01157
15. Liu, Z., Wu, Z., Tóth, R.: Smoke: single-stage monocular 3d object detection via keypoint estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops. pp. 996–997 (2020)

16. Luo, W., Yang, B., Urtasun, R.: Fast and furious: Real time end-to-end 3d detection, tracking and motion forecasting with a single convolutional net. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 3569–3577 (2018)
17. Misra, I., Girdhar, R., Joulin, A.: An End-to-End Transformer Model for 3D Object Detection. In: ICCV (2021)
18. Niemeyer, M., Mescheder, L., Oechsle, M., Geiger, A.: Occupancy flow: 4d reconstruction by learning particle dynamics. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 5379–5389 (2019)
19. Pang, Z., Li, Z., Wang, N.: Model-free vehicle tracking and state estimation in point cloud sequences. IROS (2021)
20. Pang, Z., Li, Z., Wang, N.: Simpletrack: Understanding and rethinking 3d multi-object tracking. arXiv preprint arXiv:2111.09621 (2021)
21. Park, D., Ambrus, R., Guizilini, V., Li, J., Gaidon, A.: Is pseudo-lidar needed for monocular 3d object detection? In: IEEE/CVF International Conference on Computer Vision (ICCV) (2021)
22. Pumarola, A., Corona, E., Pons-Moll, G., Moreno-Noguer, F.: D-nerf: Neural radiance fields for dynamic scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10318–10327 (2021)
23. Qi, C., Zhou, Y., Najibi, M., Sun, P., Vo, K.T., Deng, B., Anguelov, D.: Offboard 3d object detection from point cloud sequences. 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 6130–6140 (2021)
24. Qi, C.R., Litany, O., He, K., Guibas, L.J.: Deep hough voting for 3d object detection in point clouds. In: Proceedings of the IEEE International Conference on Computer Vision (2019)
25. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. CVPR (2017)
26. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. arXiv preprint arXiv:1706.02413 (2017)
27. Qi, H., Feng, C., Cao, Z.G., Zhao, F., Xiao, Y.: P2b: Point-to-box network for 3d object tracking in point clouds. pp. 6328–6337 (06 2020). https://doi.org/10.1109/CVPR42600.2020.00636
28. Rempe, D., Birdal, T., Zhao, Y., Gojcic, Z., Sridhar, S., Guibas, L.J.: Caspr: Learning canonical spatiotemporal point cloud representations. In: Advances in Neural Information Processing Systems (NeurIPS) (2020)
29. Rempe, D., Philion, J., Guibas, L.J., Fidler, S., Litany, O.: Generating useful accident-prone driving scenarios via a learned traffic prior. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2022)
30. Shi, S., Guo, C., Jiang, L., Wang, Z., Shi, J., Wang, X., Li, H.: Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2020)
31. Simonelli, A., Bulò, S.R., Porzi, L., Kontschieder, P., Ricci, E.: Demystifying pseudo-lidar for monocular 3d object detection. arXiv preprint arXiv:2012.05796 (2020)
32. Simonelli, A., Bulò, S.R., Porzi, L., Antequera, M.L., Kontschieder, P.: Disentangling monocular 3d object detection: From single to multi-class recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence **44**(3), 1219–1231 (2022). https://doi.org/10.1109/TPAMI.2020.3025077
33. 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)
34. Weng, X., Ivanovic, B., Pavone, M.: MTP: multi-hypothesis tracking and prediction for reduced error propagation. CoRR **abs/2110.09481** (2021)
35. Weng, X., Wang, J., Held, D., Kitani, K.: AB3DMOT: A Baseline for 3D Multi-Object Tracking and New Evaluation Metrics. ECCVW (2020)
36. Weng, X., Wang, Y., Man, Y., Kitani, K.M.: GNN3DMOT: graph neural network for 3d multi-object tracking with 2d-3d multi-feature learning. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020. pp. 6498–6507. Computer Vision Foundation / IEEE (2020). https://doi.org/10.1109/CVPR42600.2020.00653
37. Yan, Y., Mao, Y., Li, B.: Second: Sparsely embedded convolutional detection. Sensors **18**(10) (2018). https://doi.org/10.3390/s18103337, https://www.mdpi.com/1424-8220/18/10/3337
38. Ye, J., Chen, Y., Wang, N., Wang, X.: Online adaptation for implicit object tracking and shape reconstruction in the wild. IEEE Robotics and Automation Letters pp. 1–8 (2022). https://doi.org/10.1109/LRA.2022.3189185
39. Yin, T., Zhou, X., Krähenbühl, P.: Center-based 3d object detection and tracking. CVPR (2021)
40. Yu, C., Ma, X., Ren, J., Zhao, H., Yi, S.: Spatio-temporal graph transformer networks for pedestrian trajectory prediction. In: Proceedings of the European Conference on Computer Vision (ECCV) (August 2020)
41. Zaech, J.N., Liniger, A., Dai, D., Danelljan, M., Van Gool, L.: Learnable online graph representations for 3d multi-object tracking. IEEE Robotics and Automation Letters (2022)
42. Zaech, J.N., Liniger, A., Dai, D., Danelljan, M., Van Gool, L.: Learnable online graph representations for 3d multi-object tracking. IEEE Robotics and Automation Letters **7**(2), 5103–5110 (2022). https://doi.org/10.1109/LRA.2022.3145952
43. Zhang, W., Zhou, H., Sun, S., Wang, Z., Shi, J., Loy, C.C.: Robust multi-modality multi-object tracking. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 2365–2374 (2019)
44. Zhang, Y., Wang, C., Wang, X., Zeng, W., Liu, W.: Fairmot: On the fairness of detection and re-identification in multiple object tracking. International Journal of Computer Vision **129**(11), 3069–3087 (2021)
45. Zheng, C., Yan, X., Gao, J., Zhao, W., Zhang, W., Li, Z., Cui, S.: Box-aware feature enhancement for single object tracking on point clouds. CoRR **abs/2108.04728** (2021), https://arxiv.org/abs/2108.04728
46. Zhou, C., Luo, Z., Luo, Y., Liu, T., Pan, L., Cai, Z., Zhao, H., Lu, S.: Pttr: Relational 3d point cloud object tracking with transformer. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2022)
47. Zhou, Y., Tuzel, O.: Voxelnet: End-to-end learning for point cloud based 3d object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4490–4499 (2018)
48. Zhu, B., Jiang, Z., Zhou, X., Li, Z., Yu, G.: Class-balanced Grouping and Sampling for Point Cloud 3D Object Detection. arXiv e-prints arXiv:1908.09492 (Aug 2019)