PTTR: Relational 3D Point Cloud Object Tracking with Transformer

Changqing Zhou[1,3][†][*]   Zhipeng Luo[2,3][†]   Yueru Luo[1][†]   Tianrui Liu[1,3]
Liang Pan[2][‡]   Zhongang Cai[3]   Haiyu Zhao[3]   Shijian Lu[1]

[1] Nanyang Technological University [2] S-Lab, Nanyang Technological University [3] Sensetime Research

Abstract

*In a point cloud sequence, 3D object tracking aims to predict the location and orientation of an object in the current search point cloud given a template point cloud. Motivated by the success of transformers, we propose **P**oint **T**racking **TR**ansformer (PTTR), which efficiently predicts high-quality 3D tracking results in a coarse-to-fine manner with the help of transformer operations. PTTR consists of three novel designs. **1)** Instead of random sampling, we design Relation-Aware Sampling to preserve relevant points to given templates during subsampling. **2)** Furthermore, we propose a Point Relation Transformer (PRT) consisting of a self-attention and a cross-attention module. The global self-attention operation captures long-range dependencies to enhance encoded point features for the search area and the template, respectively. Subsequently, we generate the coarse tracking results by matching the two sets of point features via cross-attention. **3)** Based on the coarse tracking results, we employ a novel Prediction Refinement Module to obtain the final refined prediction. In addition, we create a large-scale point cloud single object tracking benchmark based on the Waymo Open Dataset. Extensive experiments show that PTTR achieves superior point cloud tracking in both accuracy and efficiency. Our code is available at* `https://github.com/Jasonkks/PTTR`.

1. Introduction

With the rapid development of 3D sensors in the past decade, solving various vision problems [16, 20, 21, 23, 24, 26, 29, 42, 43, 45] with point clouds has attracted increasing attention due to the huge potential in applications such as autonomous driving, motion planning, and robotics. As a long-standing research problem in computer vision, object tracking with point clouds has also drawn wide research interests. 3D object tracking aims to detect not only object poses and positions in each frame but also object motion
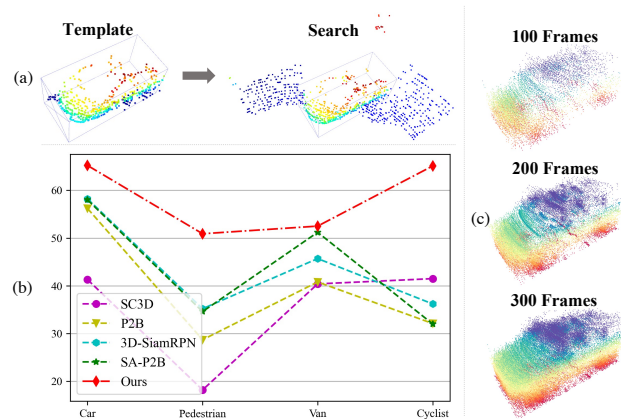
---

[*]Work done during an internship at Sensetime
[†]Equal contribution
[‡]Corresponding author

Figure 1. **(a)** 3D point cloud object tracking aims to track the target object based on a given template point cloud. **(b)** PTTR outperforms existing approaches by large margins on KITTI tracking dataset [9]. **(c)** We visualize our tracking results over consecutive frames, including "100 Frames", "200 Frames" and "300 Frames", which demonstrate the robustness of PTTR for long-term tracking.

trajectories across consecutive frames. However, 3D tracking still faces a number of open and challenging problems such as LiDAR point cloud sparsity, random shape incompleteness, texture feature absence, etc.

Existing 3D object tracking approaches can be largely categorized into two groups: multi-object tracking (MOT) and single-object tracking (SOT). MOT methods [36, 39, 40, 46] generally adopt a detect-to-track strategy by first detecting objects in each frame and then matching the detections across consecutive frames based on the estimated location or speed. In contrast, most SOT methods are only required to process a subset of point clouds, which usually come with much lower computational consumption and higher throughput. We study SOT in this work, and our objective is to estimate the location and orientation of a single object in the search frame given an object template.

The pioneer 3D SOT method SC3D [10] first generates a series of candidates given the last location of a specific object, and the prediction is made by selecting the best-matched candidate in the latent space. However, it is not end-to-end trainable and suffers from low inference speed

due to requiring a large number of candidates. Without using many candidates, P2B [25] first use cosine similarity to fuse features of the search region with the template and then adopts the prediction head of VoteNet [22] to generate the final prediction. Following P2B, SA-P2B [52] adds an extra auxiliary network to predict the object structure. In a similar framework, 3D-SiamRPN [8] uses a cross-correlation module for feature matching and an RPN head for final prediction. These methods [10, 25, 52] essentially perform a linear matching process between features in the search domain and the template, which cannot adapt to different 3D observations caused by random noise, sparsity, and occlusions. Moreover, the inclusion of complex prediction heads as in detection models highly limits their tracking speed, which is a crucial factor for online applications.

In this work, we design Point Tracking TRansformer (PTTR), a novel tracking paradigm that achieves high-quality 3D object tracking in a coarse-to-fine fashion. Specifically, PTTR first extracts point features from the template and search area individually using the PointNet++ [24] backbone. To alleviate the point sparsity issue, we propose a sampling strategy termed *Relation-Aware Sampling*, which can preserve more points that are relevant to the given template by leveraging the relation-aware feature similarities between the search and the template. We then propose a novel *Point Relation Transformer* (PRT) equipped with *Relation Attention Module* to match search and template features and generate a coarse prediction based on the matched feature. PRT first utilizes a self-attention operation to adaptively aggregate point features for the template and the search area individually, and then performs feature matching with a cross-attention operation. Moreover, we propose a lightweight *Prediction Refinement Module* to refine the coarse prediction with local feature pooling. We highlight that PTTR is more efficient than existing methods despite its prediction refinement process.

KITTI tracking dataset [9] has been widely adopted in 3D tracking evaluations. However, it has clear constraints including limited sample size and highly imbalanced class distributions. We create a new point cloud tracking benchmark named Waymo SOT Dataset based on Waymo Open Dataset [30], which has a large sample size as well as balanced class distributions. The new benchmark is thus complementary to the KITTI tracking dataset by offering more holistic and comprehensive evaluations to the 3D tracking research community. Extensive experiments on both datasets demonstrate the superior performance of PTTR in both accuracy and efficiency.

Our **key** contributions are summarized as: **1)** We propose PTTR, a transformer-based 3D point cloud object tracking method, which employs a novel coarse-to-fine tracking paradigm to first generate coarse global prediction and refine it with Local Pooling. **2)** We design two novel modules

in PTTR including Point Relation Transformer for effective feature aggregation and matching, and Relation-Aware Sampling for preserving more template-relevant points. **3)** PTTR surpasses previous SoTA methods by large margins in performance with lower computational cost. **4)** We generate a new large-scale point cloud tracking dataset based on the Waymo Open Dataset [30] to facilitate more comprehensive evaluations of 3D object tracking approaches.

2. Related Works

2D Object Tracking. Majority of the recent 2D object tracking methods follow the Siamese network paradigm, composed of two CNN branches with shared parameters that help project inputs into the same feature space. [31] employs Siamese network to learn a generic matching function for different objects. At inference, a bunch of candidates are used to match the original target and the one that matches best is chosen as the prediction. [2] proposes a fully-convolutional Siamese architecture to locate the target object in a larger search area. [12] proposes a dynamic Siamese network that learns to transform the target appearance and suppress the background. [17, 18] apply Siamese network to extract features and use pair-wise correlation separately for classification branch and regression branch of the region proposal network (RPN). 2D tracking approaches are not directly applicable to point clouds as they are driven by 2D CNN architectures and they are not designed to address the unique challenges of 3D tracking.

3D Object Tracking. 3D object tracking can be roughly divided into two categories: multi-object tracking (MOT) and single-object tracking (SOT). Most MOT approaches adopt a detect-to-track strategy and mainly focus on data association [14, 41]. [38] first proposes a 3D detection module to provide the 3D bounding boxes, then use 3D Kalman Filter to predict current estimation, and match them using Hungarian algorithm. [36] proposes to use GNN to model relationships among different objects both spatially and temporally, while [46] uses a closest distance matching after speed compensation. SOT methods focus on tracking a single object given a template. SC3D [10] proposes to match feature distance between candidates and target and regularize the training using shape completion. P2B [25] matches search and template features with cosine similarity and employs Hough Voting [22] to predict the current location. SA-P2B [52] proposes to learn the object structure as an auxiliary task. 3D-SiamRPN [8] uses a RPN [27] head to predict the final results. BAT [51] encodes box information in Box Cloud to incorporate structural information. MLVSNet [37] proposes to perform multi-level Hough voting for aggregating information from different levels. PTT [28] proposes a Point-Track-Transformer module to weight features' importance. Most existing SOT methods either use cosine similarity or cross-correlation to match the search and template features,
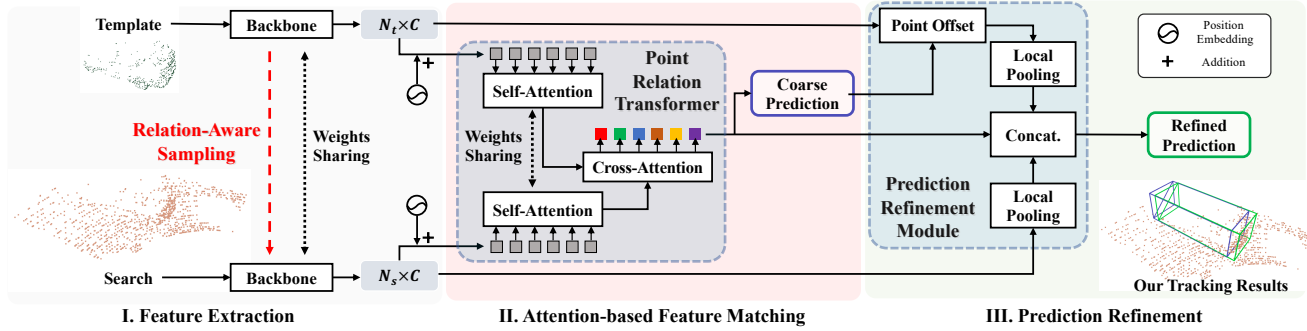
Figure 2. **Overview of our proposed PTTR.** The network mainly consists of three parts: feature extraction, attention-based feature matching and prediction refinement. The backbone is used to extract features from input point clouds. We modify PointNet++ [24] with our proposed *Relation-Aware Sampling* to help select more positive points from the search area. For feature matching, we propose *Point Relation Transformer* equipped with *Relation Attention Module* to match search and template features. In the prediction stage, we propose a *Prediction Refinement Module* to generate predictions in a coarse-to-fine manner.

which are essentially linear matching processes and cannot adapt to complex situations where random noise and occlusions are involved. Moreover, the use of detection model prediction heads leads to high computation overheads. Our proposed method in this paper address the above limitations.

Vision Transformers. Transformer [32] was first proposed as an attention-based building block in machine translation to replace the RNN architecture. Recently, a number of works [3, 5, 19, 44, 47, 48, 53] apply transformer on 2D vision tasks and achieve great success. Most of these attempts divide the images into overlapping patches and then regard each patch as a token to further apply the transformer architecture. In the 3D domain, PCT [11] generates positional embedding using 3D coordinates of points and adopts transformer with an offset attention module to enrich features of points from its local neighborhood. Point Transformer [50] adopted vectorized self-attention network [49] for local neighbours and designed a Point Transformer layer that is order-invariant to suit point cloud processing. [7] proposes SortNet to gather spatial information from point clouds, which sorts the points by learned scores to achieve order invariance. All of these works focus on shape classification or part segmentation tasks. The attention mechanism in transformer offers correlation modeling with global receptive field, which makes it a good candidate for the 3D tracking problem where feature matching is required. We propose a novel transformer-based module to utilize the attention mechanism for feature aggregation and matching.

3. Method

3.1. System Overview

Given a 3D point cloud sequence, 3D object tracking aims to estimate the object location and orientation in each point cloud observation, *i.e.* the search point cloud $P^s \in \mathbb{R}^{N_s \times 3}$, by predicting a bounding box conditioned on a

template point cloud $P^t \in \mathbb{R}^{N_t \times 3}$. To this end, we propose PTTR, a novel coarse-to-fine framework for 3D object tracking. As shown in Fig. 2, PTTR performs 3D point cloud tracking with three main stages: 1) Feature Extraction (Sec. 3.2); 2) Attention-based Feature Matching (Sec. 3.3); and 3) Prediction Refinement (Sec. 3.4).

Feature Extraction. Following previous methods [8, 10, 25, 52], we employ PointNet++ [24] as the backbone to extract multi-scale point features from the template and the search. However, important information loss may occur during random subsampling in the original PointNet++. We therefore propose a novel *Relation-Aware Sampling* to preserve more points relevant to the given template by leveraging relation-aware feature similarities.

Attention-based Feature Matching. Different from previous methods that often use cosine similarity [10, 25, 52] or linear correlation [8] for matching the template and the search, we utilize novel attention operations and propose *Point Relation Transformer* (PRT). PRT first utilizes a self-attention operation to adaptively aggregate point features for the template and the search area individually, and then performs feature matching with cross-attention. The coarse prediction is generated based on the output of PRT.

Prediction Refinement. The coarse prediction is further refined with a lightweight *Prediction Refinement Module* (PRM), which results in a coarse-to-fine tracking framework. Based on the coarse predictions, we first conduct a Point Offset operation for seed points from the search to estimate their corresponding seed points in the template. Afterwards, we employ a Local Pooling operation for the seed points from both point clouds respectively, and then concatenate the pooled features with the matched features from PRT for estimating our final prediction.

3.2. Relation-Aware Feature Extraction

As one of the most successful backbones, Point-Net++ [24] introduces a hierarchical architecture with mul-
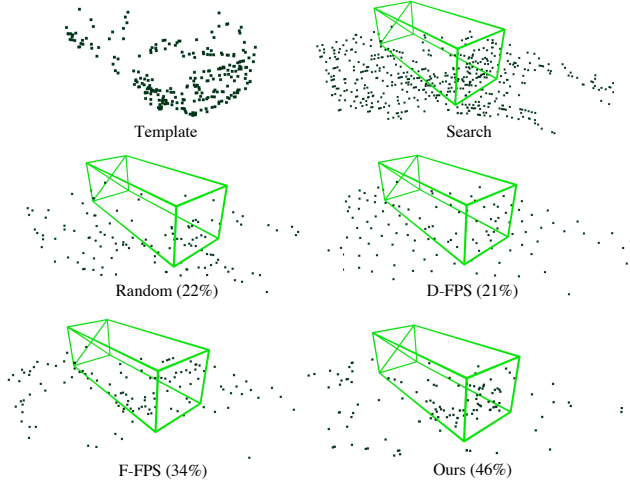
Figure 3. **Comparison of sampling methods**. We show the sampled points from the search area using different sampling methods. Our proposed sampling method preserves the most number of points belonging to the object. The percentage in the figure represents the ratio of positive points to all sampled points.

tiple distance-farthest point sampling (D-FPS) and ball query operations, which effectively exploits multi-scale point features. Most existing 3D object tracking methods [8, 25, 52] use PointNet++ for feature extraction. However, it has a non-negligible disadvantage for object tracking: the D-FPS sampling strategy used in PointNet++ tends to generate random samples that are uniformly-distributed in the euclidean space, which often leads to important information loss during the sampling process. In particular, the search point cloud often has a much larger size than the template, and therefore D-FPS sampling inevitably keeps a substantial portion of background points and leads to sparse point distribution for the object of interest, which further challenges the subsequent template searching using feature matching. To alleviate this problem, previous methods use either random point sampling [25, 52] or feature-farthest point sampling (F-FPS) [45]. However, the problem of substantial foreground information loss during sampling is not fully resolved.

Relation-Aware Sampling. In contrast, we propose to use a novel sampling method dubbed *Relation-Aware Sampling* (RAS) to preserve more points relevant to the given template by considering relational semantics. Our key insight is that the region of interest in the search point cloud should have similar semantics with the template. Therefore, points in search area with higher semantic feature similarities to the template points are more likely to be foreground points. Specifically, given the template point features $\mathbf{X}^t \in \mathbb{R}^{N_t \times C}$ and search area point features $\mathbf{X}^s \in \mathbb{R}^{N_s \times C}$, we first calculate the pairwise point feature distance matrix $\mathbf{D} \in \mathbb{R}^{N_s \times N_t}$:

$$\mathbf{D}_{ij} = ||\mathbf{x}_i^s - \mathbf{x}_j^t||_2, \quad \forall \mathbf{x}_i^s \in \mathbf{X}^s, \quad \forall \mathbf{x}_j^t \in \mathbf{X}^t, \quad (1)$$

where $||\cdot||_2$ denotes L2-norm, and N_s and N_t are the current number of points from the search area and the template, respectively. Afterwards, we compute the minimum distance $\mathbf{V} \in \mathbb{R}^{N_s}$ by considering the distance between each point from search and its nearest point from template in the feature space:

$$\mathbf{V}_i = \min_{j=1}^{N_t}(\mathbf{D}_{ij}), \quad \forall i \in \{1, 2, ..., N_s\}. \quad (2)$$

Following previous methods [8, 10, 25, 52], we update the template point cloud for each frame by using the tracking result from the previous search point cloud. In the case that low-quality tracking predictions are encountered in difficult situations, the newly formed templates might mislead the RAS and lead to unfavorable sampling results. Moreover, the inclusion of background information offers useful contextual information for the localization of the tracked object. To improve the robustness of the sampling process, we adopt a similar strategy as in [45] to combine our proposed RAS with randomly sampling. In practice, we sample half of the points with RAS, while the rest of the points are obtained via random sampling. We show the effects of different sampling approaches in Fig. 3. It can be observed that the proposed sampling method can preserve the most object points.

3.3. Relation-Enhanced Feature Matching

Existing 3D object tracking methods perform feature matching between the search point cloud and template by using cosine similarity [10, 25, 52] or linear correlation [8]. On the other hand, motivated by the success of various attention-based operations for computer vision applications [11, 21, 32, 50], we strive to explore attention-based mechanism for 3D tracking, which can adapt to different noisy point cloud observations. Although PTT [28] utilizes transformer in their model, they still match template and search point cloud by cosine similarity and the transformer module is only used for feature enhancement.

Relation Attention Module. Inspired by recent works studying feature matching [4, 33–35], we propose the *Relation Attention Module* (RAM) (shown in Fig. 4) to adaptively aggregate features by predicted attention weights. Firstly, RAM employs linear projection layers to transform the input feature vectors "Query", "Key" and "Value". Instead of naively calculating the dot products between "Query" and "Key", RAM predicts the attention map by calculating the cosine distances between the two sets of L2-normalized feature vectors. With the help of L2-normalization, RAM can prevent the dominance of a few feature channels with extremely large magnitudes. Subsequently, the attention map is normalized with a Softmax operation. In order to sharpen the attention weights and meanwhile reduce the influence of noise [11], we employ the off-
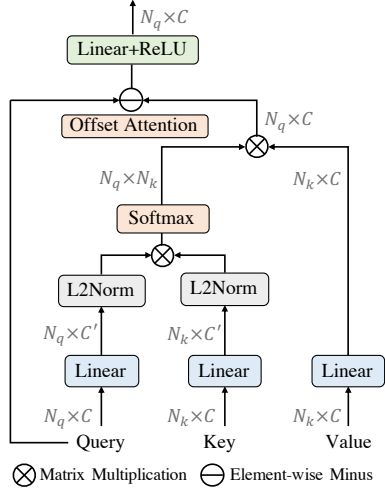
Figure 4. **Architecture of the Relation Attention Module (RAM).** RAM first projects query, key and value into a latent feature space and then estimates the attention matrix by multiplying the L2-normalized query and key features. The attention matrix is then applied on the value feature to obtain the attention product before the offset attention [11] operation and Linear and ReLU layers for injection of non-linearity.

set attention to predict the final attention map by subtracting the query features with the previously normalized attention map. Consequently, the proposed RAM can be formulated as:

$$\text{Attn}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \phi\big(\mathbf{Q} - \text{softmax}(\mathbf{A}) \cdot (W_v \mathbf{V})\big), \quad (3)$$

where ϕ represents the linear layer and ReLU operation applied to the output features, the attention matrix $\mathbf{A} \in \mathbb{R}^{N_q \times N_k}$ is obtained by:

$$\mathbf{A} = \overline{\mathbf{Q}} \cdot \overline{\mathbf{K}}^{\top}, \ \overline{\mathbf{Q}} = \frac{W_q \mathbf{Q}}{||W_q \mathbf{Q}||_2}, \ \overline{\mathbf{K}} = \frac{W_k \mathbf{K}}{||W_k \mathbf{K}||_2}, \quad (4)$$

where $|| \cdot ||_2$ is the L2-norm, $\mathbf{Q}, \mathbf{K}, \mathbf{V}$ represent the input "Query", "Key" and "Value" respectively, and W_q, W_k and W_v denote the corresponding linear projections.

Point Relation Transformer. By incorporating RAM, we propose the *Point Relation Transformer* (PRT) module to adaptively exploit the correlations between point features for context enhancement. The PRT module firstly performs a self-attention operation for the search and the template features, respectively. Subsequently, PRT employs a cross-attention operation for gathering cross-contextual information between the two point clouds. Both operations use global attentions, where all input point feature vectors are considered as tokens. Formally, PRT is formulated as:

$$\overline{\mathbf{X}}^s = \text{Attn}(\mathbf{X}^s), \ \text{and} \ \overline{\mathbf{X}}^t = \text{Attn}(\mathbf{X}^t), \quad (5)$$

$$\hat{\mathbf{X}}^s = \text{Attn}(\overline{\mathbf{X}}^s, \overline{\mathbf{X}}^t, \overline{\mathbf{X}}^t), \quad (6)$$

where $\text{Attn}(\mathbf{Q}, \mathbf{K}, \mathbf{V})$ denotes our proposed *Relation Attention Module*, $\hat{\mathbf{X}}^s$ denotes the matched features, and $\overline{\mathbf{X}}^s$ and $\overline{\mathbf{X}}^t$ denote the enhanced search and template features respectively. By using global self-attention, the exploited features can obtain a global understanding for the current observation. Note that the self-attention use the same point features as $\mathbf{Q}, \mathbf{K}, \mathbf{V}$, and both self-attention operations share weights so as to project the search and template features into the same latent space. Thereafter, the cross-attention performs pairwise matching between query tokens $\overline{\mathbf{X}}^s$ and key tokens $\overline{\mathbf{X}}^t$, which exploits cross-contextual information for $\hat{\mathbf{X}}^s$ by capturing correlations between the two sets of point features. Based on the relation enhanced point features $\hat{\mathbf{X}}^s$, we can generate our coarse prediction results for 3D object tracking.

3.4. Coarse-to-Fine Tracking Prediction

Majority of the existing point tracking approaches adopt prediction heads of detection models to generate the predictions, e.g. P2B [25] adopts the clustering and voting operations of VoteNet [22] and 3D-SiamRPN [8] uses a RPN [18, 27] head. However, these prediction heads introduce extra computation overheads, which largely limits their efficiency. To circumvent this issue, we propose a novel coarse-to-fine tracking framework. The coarse prediction $\mathbf{Y}^c$ is predicted by directly regressing the relation enhanced features $\hat{\mathbf{X}}^s$ from the proposed PRT module with Multi-Layer-Perceptron (MLP). Remarkably, $\mathbf{Y}^c$ provides faithful tracking predictions for most cases, and also surpasses the tracking performance of SoTA methods.

Prediction Refinement Module. To further refine the tracking predictions, we propose a lightweight *Prediction Refinement Module* (PRM) to predict our final predictions $\mathbf{Y}^f$ based on $\mathbf{Y}^c$. Specifically, we use the sampled points from the search point cloud as seed points, and then we estimate their correspondences in the template by using an offset operation for $\mathbf{Y}^c$. Afterwards, we encode local discriminative feature descriptors for the seed points from both sources, which is achieved by using Local Pooling operations for grouped neighboring point features. The neighboring features are grouped by using ball-query operations with a fixed radius r. In the end, we concatenate $\hat{\mathbf{X}}^s$ with the pooled features from the source and the target, based on which we generate the final prediction $\mathbf{Y}^f$:

$$\mathbf{Y}^f = \gamma([\mathbf{F}^s, \mathbf{F}^t, \hat{\mathbf{X}}^s]), \quad (7)$$

where $\mathbf{F}^s$ and $\mathbf{F}^t$ are the pooled features from search and template respectively, $[\cdot]$ denotes concatenation operation, and γ represents the MLP networks. We highlight that even with the refinement stage, our proposed method still has lower computational complexity than existing methods thanks to the lightweight design.

Table 1. **Dataset Statistics.** Number of *tracklets / samples* from different categories in KITTI [9] and Waymo SOT Dataset.

Dataset	Car / Vehicle	Pedestrian	Van	Cyclist
KITTI (train)	441 / 19522	96 / 4600	38 / 1994	27 / 1529
KITTI (test)	120 / 6424	62 / 6088	16 / 1248	8 / 308
Waymo (train)	16119 / 241544	15452 / 249800	-	1066 / 22389
Waymo (test)	1658 / 53377	949 / 27308	-	138 / 5374

Training Loss. Our PTTR is trained in an end-to-end manner. The coarse prediction $\mathbf{Y}^c$ and the final prediction $\mathbf{Y}^f$ are in the same form that each contains a classification component $\mathbf{Y}_{cls} \in \mathbb{R}^{N_s \times 1}$ and a regression component $\mathbf{Y}_{reg} \in \mathbb{R}^{N_s \times 4}$, where N_s denotes the number of sampled points from the search area. $\mathbf{Y}_{cls}$ predicts the objectiveness of each point, and $\mathbf{Y}_{reg}$ consists of the predicted offsets along each axis $\{\Delta x, \Delta y, \Delta z\}$ with an additional rotation angle offset $\Delta \theta$. For each prediction, we use a classification loss $\mathcal{L}_{cls}$ defined by binary cross entropy, and a regression loss $\mathcal{L}_{reg}$ computed by mean square error. Consequently, our overall loss function is formulated as:

$$\mathcal{L}_{total} = \mathcal{L}_{cls}(\mathbf{Y}^c_{cls}, \mathbf{Y}^{gt}_{cls}) + \mathcal{L}_{reg}(\mathbf{Y}^c_{reg}, \mathbf{Y}^{gt}_{reg}) + \quad (8)$$

$$\lambda\big(\mathcal{L}_{cls}(\mathbf{Y}^f_{cls}, \mathbf{Y}^{gt}_{cls}) + \mathcal{L}_{reg}(\mathbf{Y}^f_{reg}, \mathbf{Y}^{gt}_{reg})\big), \quad (9)$$

where $\mathbf{Y}^{gt}_{(\cdot)}$ denotes the corresponding ground truth, and λ is a weighting parameter.

4. Waymo 3D Single Object Tracking Dataset

Existing Benchmark. Existing methods carry out their evaluations on the KITTI [9] tracking dataset. The data split, tracklet generation, and evaluation metrics are specified in [10]. However, we find this dataset offers a limited number of samples while the object classes are highly imbalanced. Tab. 1 shows the statistics of the datasets and we can observe that the car category accounts for the majority of the tracklets and samples, while some classes have few examples. [25] studies the impact of the limited training samples and finds the performance severely degrades with insufficient training data. Therefore, we believe it will be beneficial if a large-scale dataset is available.

Constructing Waymo SOT Dataset. Fortunately, we find that the recently released Waymo Open Dataset [30] is able to fulfill this need. Although Waymo does not directly contain a SOT dataset, in its detection dataset, every object is not only annotated with the bounding box but also with a unique object ID, which makes it feasible to extract tracklets from the point cloud sequences. To alleviate the class imbalance issue, we use 10% of the training and validation sequences to produce Vehicle tracklets, 20% of the sequences for Pedestrian, and all the sequences for Cyclist since it is the rarest. We eliminate objects with less than 10 points

inside the ground truth bounding box and remove tracklets with lengths less than 3 frames.

Properties and Advantages. The statistics of the generated dataset are reported in Tab. 1. Although the Waymo dataset does not differentiate different vehicles, such as cars and vans, the Waymo SOT dataset is of a significantly larger scale with a more balanced class distribution than the KITTI tracking dataset. For example, there are 15,452 "Pedestrian" tracklets in the proposed Waymo SOT Dataset, while KITTI Tracking dataset only has 96 tracklets.

5. Experiments

We evaluate our proposed method on both KITTI [9] and Waymo SOT Dataset for comprehensive comparison.

Evaluation Metrics. Following the evaluation metrics of P2B [25] and measure the "Success" and "Precision". Specifically, "Success" is defined as the IoU between predicted boxes and the ground truth, and "Precision" measures the AUC (Area Under Curve) of the distance between prediction and ground truth box centers from 0 to 2 meters.

Template and Search. During training, we use the ground truth bounding box to crop the point cloud to form the template. In order to simulate the disturbances the model might encounter, we add random distortions to augment the bounding boxes with a range of [-0.3 to 0.3] along x, y, z axes. For both training and testing, we extend the box with a ratio of 0.1 to include some background points. We enlarge the template bounding box by 2 meters in all directions to form the search area.

Model Details. We use PointNet++ [24] with 3 set-abstraction layers as the backbone. The radius of these SA layers is set to 0.3, 0.5, 0.7 meters, respectively. In the first stage, we use a 3-layer MLP for classification and regression, respectively. Each layer is followed by a BN [13] layer and an ReLU [1] activation layer. In PRM, the Local Pooling is conducted with a ball-query operation and a grouping operation [24] with a radius of 1.0 meter. After pooling, we obtain the concatenated features that are fed into a 5-layer MLP for generating the final predictions.

Training and Testing. For KITTI tracking dataset, we train the model for 160 epochs with a batch size of 64. We use Adam optimizer [15] with an initial learning rate of 0.001 and reduce it by 5 every 40 epochs. For the Waymo SOT Dataset, we train the model for 80 epochs with the same initial learning rate and reduce the learning rate every 20 epochs. During testing, we use the previous prediction result as the next template. In line with [10, 25], we use the ground truth bounding box as the first template.

5.1. 3D Tracking on KITTI Tracking Dataset

To achieve fair comparisons against previous methods, we follow the data split and processing specified in [10, 25].

Table 2. **Performance comparison on the KITTI dataset.** Success / Precision are used for evaluation.

Method	Car	Pedestrian	Van	Cyclist	Average
SC3D [10]	41.3 / 57.9	18.2 / 37.8	40.4 / 47.0	41.5 / 70.4	35.4 / 53.3
P2B [25]	56.2 / 72.8	28.7 / 49.6	40.8 / 48.4	32.1 / 44.7	39.5 / 53.9
3D-SiamRPN [8]	58.2 / 76.2	35.2 / 56.2	45.7 / 52.9	36.2 / 49.0	43.8 / 58.6
SA-P2B [52]	58.0 / 75.1	34.6 / 63.3	51.2 / 63.1	32.0 / 43.6	44.0 / 61.3
MLVSNet [37]	56.0 / 74.0	34.1 / 61.1	52.0 / 61.4	34.3 / 44.5	44.1 / 60.3
BAT [51]	60.5 / 77.7	42.1 / 70.1	52.4 / **67.0**	33.7 / 45.4	47.2 / 65.1
PTT [28]	**67.8 / 81.8**	44.9 / 72.0	43.6 / 52.5	37.2 / 47.3	48.4 / 63.4
LTTR [6]	65.0 / 77.1	33.2 / 56.8	35.8 / 45.6	**66.2** / 89.9	50.0 / 67.4
Ours	65.2 / 77.4	**50.9 / 81.6**	**52.5** / 61.8	65.1 / **90.5**	**58.4 / 77.8**

As reported in Tab. 2, PTTR surpasses the previous state-of-the-art method by a significant margin of 8.4 and 10.4 in terms of average Success and Precision. In particular, PTTR significantly outperforms SA-P2B on the challenging categories, Pedestrian and Cyclist. As shown in Fig. 6, we compare the proposed PTTR against P2B [25] over two pedestrian point cloud sequences. P2B often makes wrong predictions when multiple instances are close, while PTTR is able to generate stable and reliable predictions. The impressive performance gains are largely attributed to the proposed PRT and RAS modules, which will be further explained in the ablation studies in Sec. 5.2. Moreover, we visualize the coarse and refined predictions in Fig. 5. The coarse predictions are further corrected in the refinement stage, especially when point sparsity or large movements are present. It demonstrates that our proposed prediction refinement via local feature pooling is able to adapt to challenging situations and generate robust predictions.

Table 3. **Inference Time.**

SC3D [10]	P2B [25]	PTTR (Ours)
66.3ms	23.6ms	**19.9ms**

Inference Time. Speed is a key factor in object tracking tasks. Hence, we test the model inference time on the KITTI test dataset with a Tesla V100 GPU. As reported in Tab. 3, under the same configurations, PTTR achieves the shortest average runtime of 19.9ms.

5.2. Ablation Studies

To evaluate the effectiveness of the components proposed in PTTR, we conduct ablation studies on the KITTI [9] dataset and report the Success and Precision.
Sampling Methods. We compare our proposed Relation-Aware Sampling (RAS) method with existing sampling approaches, including random sampling [25], distance-farthest point sampling (D-FPS) [24] and feature-farthest point sampling (F-FPS) [45]. As shown in Tab. 4, RAS yields the best performance with a clear margin. By utilizing RAS, our method achieves an increase of 5.5/8.8 in Success/Precision than the random sampling baseline. Small

Table 4. **Performance comparison on different sampling methods.** D-FPS refers to distance-farthest point sampling and F-FPS denotes feature-farthest point sampling. RAS refers to our proposed Relation-Aware Sampling.

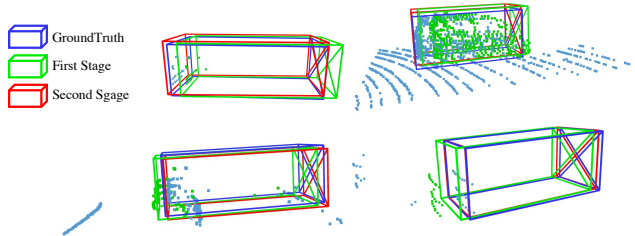| Method	Car	Pedestrian	Van	Cyclist	Average
Random [25]	62.4 / 74.0	36.6 / 59.9	50.4 / 58.3	62.2 / 83.9	52.9 / 69.0
D-FPS [24]	61.3 / 73.0	42.5 / 68.8	41.8 / 47.2	59.8 / 78.5	51.4 / 66.9
F-FPS [45]	59.3 / 72.5	41.9 / 68.6	52.1 / 60.5	63.8 / 83.8	54.3 / 71.4
RAS (Ours)	**65.2 / 77.4**	**50.9 / 81.6**	**52.5 / 61.8**	**65.1 / 90.5**	**58.4 / 77.8**

Figure 5. **Visualization of prediction refinement.** We present 4 different objects to show that the refined stage further corrects the coarse predictions through local feature pooling.

objects usually consist of fewer points, and hence are more sensitive to the point sparsity challenge. For the pedestrian class, which is the class of the smallest object size, RAS significantly boosts the results from 36.6/59.9 to 50.9/81.6.

Table 5. **Ablation studies on model components.** For experiments that disable the Point Relation Transformer (PRT), we replace PRT with cosine similarity for feature correlation extraction as in existing methods [10, 25, 52]. We also compare the performance w/ or w/o the Prediction Refinement Module (PRM).

PRT	PRM	Car	Pedestrian	Van	Cyclist	Average
		40.2 / 52.0	23.0 / 41.6	25.9 / 34.7	30.0 / 57.8	29.8 / 46.5
✓		62.9 / 74.3	49.1 / 77.7	50.7 / 58.7	64.1 / 90.0	56.7 / 75.2
	✓	60.6 / 73.1	39.2 / 66.9	43.5 / 48.9	58.7 / 87.2	50.5 / 69.0
✓	✓	65.2 / 77.4	50.9 / 81.6	52.5 / 61.8	65.1 / 90.5	58.4 / 77.8

Model Components. We conduct experiments to investigate the effectiveness of the proposed Point Relation Transformer (PRT) and Prediction Refinement Module (PRM). For the ablation studies on PRT, we replace PRT with cosine similarity for feature correlation computation as in existing methods [10, 25, 52]. We evaluate with the coarse prediction in experiments w/o PRM. As shown in Tab. 5, when both PRT and PRM are disabled, the performance degrades sharply from 58.4 to 29.8 in Success. Both PRT and PRM improve the model performance significantly compared to the base case. Note that replacing our PRT with cosine similarity will decrease the performance by 8.8 on average in terms of Success, which shows the strong feature matching capability of the PRT module. In addition, the two components are also complementary to each other that the best results are obtained when both are enabled. We highlight that even without prediction refinement, our model still achieves the best average performance (56.7/75.2).
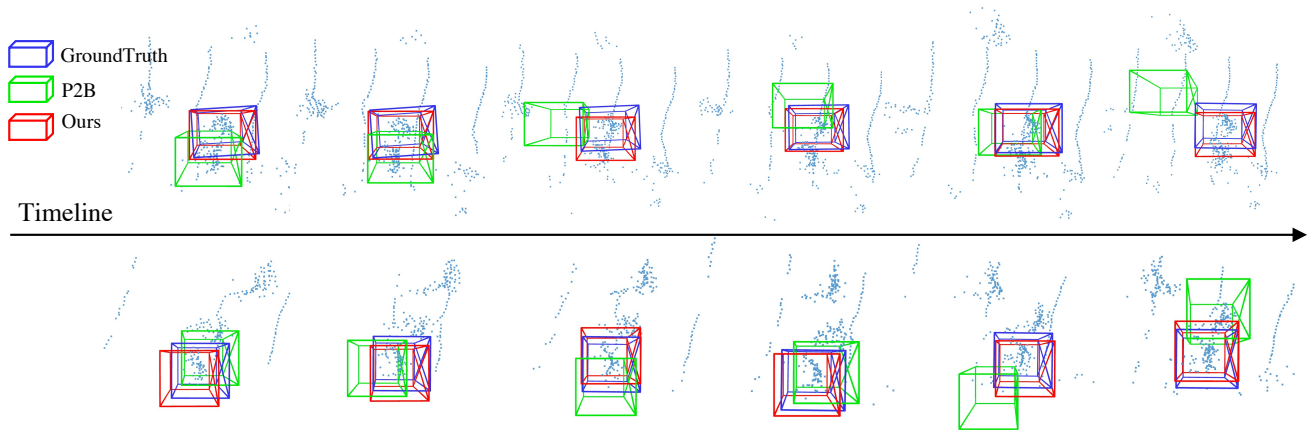
Figure 6. **Qualitative comparison with P2B [25].** We compare PTTR with P2B over two pedestrian tracking sequences. It can be observed that P2B is prone to matching mistakes when multiple instances are close, while PTTR generates more reliable predictions.

Table 6. **Ablation experiments on Relation Attention.** We investigate the effectiveness of the two major modifications in our proposed Relation Attention module. *Offset* refers to offset attention and *Norm* refers to feature normalization.

Offset	Norm	Car	Pedestrian	Van	Cyclist	Average
		55.4 / 68.0	36.6 / 65.1	34.6 / 38.6	55.9 / 78.8	45.6 / 62.6
✓		56.6 / 69.1	40.3 / 67.3	48.3 / 59.6	63.7 / 90.3	52.2 / 71.6
	✓	63.7 / 75.3	47.1 / 73.5	53.0 / 60.4	64.1 / 89.5	57.0 / 74.7
✓	✓	65.2 / 77.4	50.9 / 81.6	52.5 / 61.8	65.1 / 90.5	58.4 / 77.8

Relation Attention. The main differences between our proposed Relation Attention and regular transformer attention are the L2-normalization applied on query and key features and the offset attention. Ablation studies on each component are reported in Tab. 6. Both two operations improve the model performance, especially the L2-normalization. It reveals that the cosine distance facilitates point cloud feature matching.

Table 7. **Performance comparison on the Waymo SOT Dataset.** Success / Precision are used for evaluation.

Method	Vehicle	Pedestrian	Cyclist	Average
SC3D [10]	46.5 / 52.7	26.4 / 37.8	26.5 / 37.6	33.1 / 42.7
P2B [25]	55.7 / 62.2	35.3 / 54.9	30.7 / 44.5	40.6 / 53.9
Ours	**58.7 / 65.2**	**49.0 / 69.1**	**43.3 / 60.4**	**50.3 / 64.9**

5.3. 3D Tracking on Waymo SOT Dataset

For the Waymo SOT Dataset, we compare PTTR with SC3D [10] and P2B [25] by re-implementing the methods with the official codes. As shown in Tab. 7, PTTR again achieves the best results among all comparing methods with a large margin of 9.7 and 11.0 in average Success and Precision. Similar to KITTI, the Pedestrian and Cyclist classes see higher gains. Overall, the consistent performance improvements on different benchmarks demonstrate the effectiveness and robustness of our proposed method.

6. Limitation Discussion

We show in Fig. 7 the failure cases encountered by our model, which mainly occur when the point clouds are too sparse that the model can hardly capture enough patterns to effectively match template and search point clouds. One possible way to further mitigate this issue could be utilizing complementary multi-frame information for object tracking, which can be explored in future researches.
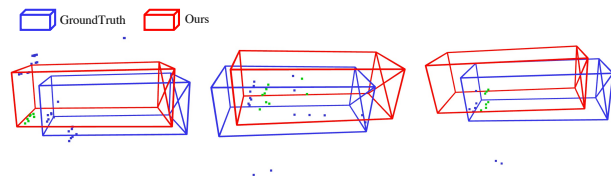

Figure 7. Our tracking failures mainly occur when the point clouds are too sparse.

7. Conclusion

In this paper, we propose PTTR, a novel framework for 3D point cloud single object tracking, which contains a designed Relation-Aware Sampling strategy to tackle point sparsity, a novel Point Relation Transformer for feature matching, and a lightweight Prediction Refinement Module. PTTR not only obtains the new state-of-the-art performance but also achieves improved efficiency. We also generate a large-scale SOT tracking dataset based on the Waymo Open Dataset to facilitate more comprehensive evaluations of 3D tracking methods. We hope our method and the Waymo SOT Dataset can help motivate further researches.

References

[1] Abien Fred Agarap. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*, 2018. 6

[2] Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr. Fully-convolutional siamese networks for object tracking. In *European conference on computer vision*, pages 850–865. Springer, 2016. 2

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pages 213–229. Springer, 2020. 3

[4] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020. 4

[5] Xiangxiang Chu, Zhi Tian, Yuqing Wang, Bo Zhang, Haibing Ren, Xiaolin Wei, Huaxia Xia, and Chunhua Shen. Twins: Revisiting the design of spatial attention in vision transformers. *arXiv preprint arXiv:2104.13840*, 1(2):3, 2021. 3

[6] Yubo Cui, Zheng Fang, Jiayao Shan, Zuoxu Gu, and Sifan Zhou. 3d object tracking with transformer. *arXiv preprint arXiv:2110.14921*, 2021. 7

[7] Nico Engel, Vasileios Belagiannis, and Klaus Dietmayer. Point transformer. *IEEE Access*, 9:134826–134840, 2021. 3

[8] Zheng Fang, Sifan Zhou, Yubo Cui, and Sebastian Scherer. 3d-siamrpn: An end-to-end learning method for real-time 3d single object tracking using raw point cloud. *IEEE Sensors Journal*, 21(4):4995–5011, 2020. 2, 3, 4, 5, 7

[9] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3354–3361. IEEE, 2012. 1, 2, 6, 7

[10] Silvio Giancola, Jesus Zarzar, and Bernard Ghanem. Leveraging shape completion for 3d siamese tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1359–1368, 2019. 1, 2, 3, 4, 6, 7, 8

[11] Meng-Hao Guo, Jun-Xiong Cai, Zheng-Ning Liu, Tai-Jiang Mu, Ralph R Martin, and Shi-Min Hu. Pct: Point cloud transformer. *Computational Visual Media*, 7(2):187–199, 2021. 3, 4, 5

[12] Qing Guo, Wei Feng, Ce Zhou, Rui Huang, Liang Wan, and Song Wang. Learning dynamic siamese network for visual object tracking. In *Proceedings of the IEEE international conference on computer vision*, pages 1763–1771, 2017. 2

[13] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015. 6

[14] Hasith Karunasekera, Han Wang, and Handuo Zhang. Multiple object tracking with attention to appearance, structure, motion and size. *IEEE Access*, 7:104423–104434, 2019. 2

[15] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[16] Loic Landrieu and Martin Simonovsky. Large-scale point cloud semantic segmentation with superpoint graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4558–4567, 2018. 1

[17] Bo Li, Wei Wu, Qiang Wang, Fangyi Zhang, Junliang Xing, and Junjie Yan. Siamrpn++: Evolution of siamese visual tracking with very deep networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4282–4291, 2019. 2

[18] Bo Li, Junjie Yan, Wei Wu, Zheng Zhu, and Xiaolin Hu. High performance visual tracking with siamese region proposal network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8971–8980, 2018. 2, 5

[19] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *arXiv preprint arXiv:2103.14030*, 2021. 3

[20] Zhipeng Luo, Zhongang Cai, Changqing Zhou, Gongjie Zhang, Haiyu Zhao, Shuai Yi, Shijian Lu, Hongsheng Li, Shanghang Zhang, and Ziwei Liu. Unsupervised domain adaptive 3d detection with multi-level consistency. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8866–8875, 2021. 1

[21] Liang Pan, Xinyi Chen, Zhongang Cai, Junzhe Zhang, Haiyu Zhao, Shuai Yi, and Ziwei Liu. Variational relational point completion network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8524–8533, 2021. 1, 4

[22] Charles R Qi, Or Litany, Kaiming He, and Leonidas J Guibas. Deep hough voting for 3d object detection in point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9277–9286, 2019. 2, 5

[23] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 1

[24] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*, 2017. 1, 2, 3, 6, 7

[25] Haozhe Qi, Chen Feng, Zhiguo Cao, Feng Zhao, and Yang Xiao. P2b: Point-to-box network for 3d object tracking in point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6329–6338, 2020. 2, 3, 4, 5, 6, 7, 8

[26] Jiawei Ren, Liang Pan, and Ziwei Liu. Benchmarking and analyzing point cloud classification under corruptions. *arXiv preprint arXiv:2202.03377*, 2022. 1

[27] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28:91–99, 2015. 2, 5

[28] Jiayao Shan, Sifan Zhou, Zheng Fang, and Yubo Cui. Ptt: Point-track-transformer module for 3d single object tracking in point clouds. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1310–1316. IEEE, 2021. 2, 4, 7

[29] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointr-cnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 770–779, 2019. 1

[30] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2446–2454, 2020. 2, 6

[31] Ran Tao, Efstratios Gavves, and Arnold WM Smeulders. Siamese instance search for tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1420–1429, 2016. 2

[32] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszko-reit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 3, 4

[33] Feng Wang, Xiang Xiang, Jian Cheng, and Alan Loddon Yuille. Normface: L2 hypersphere embedding for face verification. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 1041–1049, 2017. 4

[34] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5265–5274, 2018. 4

[35] Ning Wang, Wengang Zhou, Jie Wang, and Houqiang Li. Transformer meets tracker: Exploiting temporal context for robust visual tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1571–1580, 2021. 4

[36] Yongxin Wang, Kris Kitani, and Xinshuo Weng. Joint object detection and multi-object tracking with graph neural networks. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13708–13715. IEEE, 2021. 1, 2

[37] Zhoutao Wang, Qian Xie, Yu-Kun Lai, Jing Wu, Kun Long, and Jun Wang. Mlvsnet: Multi-level voting siamese network for 3d visual tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3101–3110, 2021. 2, 7

[38] Xinshuo Weng and Kris Kitani. A baseline for 3d multi-object tracking. *arXiv preprint arXiv:1907.03961*, 1(2):6, 2019. 2

[39] Xinshuo Weng, Jianren Wang, David Held, and Kris Kitani. 3d multi-object tracking: A baseline and new evaluation metrics. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10359–10366. IEEE, 2020. 1

[40] Xinshuo Weng, Yongxin Wang, Yunze Man, and Kris Kitani. Graph neural networks for 3d multi-object tracking. *arXiv preprint arXiv:2008.09506*, 2020. 1

[41] Yu Xiang, Alexandre Alahi, and Silvio Savarese. Learning to track: Online multi-object tracking by decision making. In *Proceedings of the IEEE international conference on computer vision*, pages 4705–4713, 2015. 2

[42] Aoran Xiao, Jiaxing Huang, Dayan Guan, and Shijian Lu. Unsupervised representation learning for point clouds: A survey. *arXiv preprint arXiv:2202.13589*, 2022. 1

[43] Aoran Xiao, Jiaxing Huang, Dayan Guan, Fangneng Zhan, and Shijian Lu. Synlidar: Learning from synthetic lidar sequential point cloud for semantic segmentation. *arXiv preprint arXiv:2107.05399*, 2021. 1

[44] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. *arXiv preprint arXiv:2105.15203*, 2021. 3

[45] Zetong Yang, Yanan Sun, Shu Liu, and Jiaya Jia. 3dssd: Point-based 3d single stage object detector. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11040–11048, 2020. 1, 4, 7

[46] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Center-based 3d object detection and tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11784–11793, 2021. 1, 2

[47] Gongjie Zhang, Zhipeng Luo, Kaiwen Cui, and Shijian Lu. Meta-detr: Few-shot object detection via unified image-level meta-learning. *arXiv preprint arXiv:2103.11731*, 2(6), 2021. 3

[48] Gongjie Zhang, Zhipeng Luo, Yingchen Yu, Kaiwen Cui, and Shijian Lu. Accelerating detr convergence via semantic-aligned matching. *arXiv preprint arXiv:2203.06883*, 2022. 3

[49] Hengshuang Zhao, Jiaya Jia, and Vladlen Koltun. Exploring self-attention for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10076–10085, 2020. 3

[50] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16259–16268, 2021. 3, 4

[51] Chaoda Zheng, Xu Yan, Jiantao Gao, Weibing Zhao, Wei Zhang, Zhen Li, and Shuguang Cui. Box-aware feature enhancement for single object tracking on point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13199–13208, 2021. 2, 7

[52] Xiaoyu Zhou, Ling Wang, Zhian Yuan, Ke Xu, and Yanxin Ma. Structure aware 3d single object tracking of point cloud. *Journal of Electronic Imaging*, 30(4):043010, 2021. 2, 3, 4, 7

[53] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020. 3