CMT: Context-Matching-Guided Transformer for 3D Tracking in Point Clouds

Zhiyang Guo[1], Yunyao Mao[1], Wengang Zhou[1,2⋆], Min Wang[2], and Houqiang Li[1,2⋆]

[1] CAS Key Laboratory of Technology in GIPAS, EEIS Department, University of Science and Technology of China
[2] Institute of Artificial Intelligence, Hefei Comprehensive National Science Center
{guozhiyang, myy2016}@mail.ustc.edu.cn, zhwg@ustc.edu.cn, wangmin@iai.ustc.edu.cn, lihq@ustc.edu.cn

Abstract. How to effectively match the target template features with the search area is the core problem in point-cloud-based 3D single object tracking. However, in the literature, most of the methods focus on devising sophisticated matching modules at point-level, while overlooking the rich spatial context information of points. To this end, we propose **C**ontext-**M**atching-Guided **T**ransformer (**CMT**), a Siamese tracking paradigm for 3D single object tracking. In this work, we first leverage the local distribution of points to construct a horizontally rotation-invariant contextual descriptor for both the template and the search area. Then, a novel matching strategy based on shifted windows is designed for such descriptors to effectively measure the template-search contextual similarity. Furthermore, we introduce a target-specific transformer and a spatial-aware orientation encoder to exploit the target-aware information in the most contextually relevant template points, thereby enhancing the search feature for a better target proposal. We conduct extensive experiments to verify the merits of our proposed CMT and report a series of new state-of-the-art records on three widely-adopted datasets.

Keywords: 3D Single Object Tracking, Point Clouds, Context Match

1 Introduction

As an essential task for autonomous driving vehicles and intelligent robotics, 3D single object tracking (SOT) has attracted substantial attention in the past few years. Different from 2D SOT that is developed on images, 3D SOT is generally performed with point clouds data. Although the recent development of deep neural networks [19,37,13] has led to the surge of 2D SOT algorithms [10,2,27,26,39], it is still non-trivial to apply these 2D methods in 3D space, especially when it comes to the sparse 3D point clouds. In general, the point-cloud-based 3D SOT methods [17,33,15,46] follow the Siamese tracking paradigm [1], which has exhibited great success in RGB images. Notably, the pioneering work P2B [33] proposes the first end-to-end 3D object tracker based on template-search comparison

⋆ Corresponding authors: Wengang Zhou and Houqiang Li.

and voting-based region proposal generation. The following work BAT [46] introduces BoxCloud feature representation and a box-aware feature fusion module to make the tracker robust to the sparse and incomplete point clouds.

Despite the significant advance, the state-of-the-art 3D trackers still suffer from non-trivial defects. Specifically, in BAT [46], the BoxCloud feature is not only exploited to facilitate the template-search comparison, but also leveraged in target proposal feature aggregation. However, in the box-aware feature fusion module of BAT, all the features belonging to different template points are simply concatenated and processed with multi-layer perceptrons (MLPs) and max-pooling, which struggles to capture useful target information from multiple template points. Moreover, the pairwise distance calculation relying only on BoxCloud may lead to mismatch for targets that are less distinctive in shape and size, since BoxCloud features merely focus on the relative position of the individual point inside a bounding box, but lack the awareness of spatial context.

Another issue lies in PointNet++ [32], as it is exploited as the backbone network for most of the existing point cloud trackers. Instead of adopting explicit T-Nets to realize geometric transformation invariance as in PointNet [31], PointNet++ achieves invariance through data augmentation. However, in 3D SOT, such geometric transformation invariance is hard to learn from data, which degrades the quality of the extracted feature when the target rotates during tracking. Subsequent modules based on the backbone feature, such as the template-search comparison, are inevitably affected.

To address the above issues, we propose a novel Context-Matching-Guided Transformer (CMT) for robust 3D SOT. Specifically, a descriptor invariant to horizontal rotations is developed for each point to describe its spatial context. Meanwhile, to effectively measure the similarity of such descriptors between points in the template and search area, we design a context matching strategy based on shifted windows. The proposed fine-grained context matching is combined with the efficient BoxCloud comparison [46] to form successive coarse-to-fine matching stages. The most relevant template points are thereby selected to be fused with each search point accordingly. Instead of simply using MLPs for template-search feature fusion, we develop a target-specific transformer to make the best of the target-aware information. Furthermore, we introduce a spatial-aware orientation encoder after the transformer to integrate information from eight spatial orientations. Finally, a region proposal network (RPN) used in [33,46] is exploited to generate the 3D target proposal. We evaluate our method on three prevalent 3D SOT benchmarks,

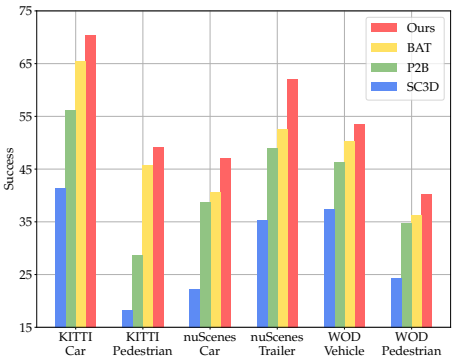

Fig. 1. Our method (CMT) significantly outperforms representative point cloud trackers [17,33,46] on multiple benchmarks.

including KITTI [16], nuScenes [5] and Waymo Open Dataset (WOD) [35]. As shown in Fig. 1, our method sets a series of state-of-the-art records.

Our main contributions can be summarized as follows:

- We introduce a horizontally rotation-invariant descriptor for contextual description, and design a shifted-window matching strategy to effectively measure the spatial contextual similarity between points.
- We propose a context-matching-guided transformer for 3D object tracking, and present an elegant view of how to integrate point contextual cues and target-aware information into the feature of the search area.
- We conduct extensive experiments to validate the merits of our method, and show promising improvements on several prevalent benchmarks.

2 Related Works

2.1 2D Siamese Tracking

Recent years have witnessed the great development of 2D SOT approaches. Within these approaches, the Siamese tracking paradigm has demonstrated its advantages over the traditional discriminative correlation filter [4,20,12,21,11]. Specifically, a Siamese tracker is composed of two branches (*i.e.*, the template and the search area) with a shared feature extraction network to project RGB frames into an implicit feature space. The tracking task is then formulated as a feature matching problem. Following the pioneering work SiamFC [1], many Siamese trackers [27,47,26,39,42,8,18] have proved their competitiveness. In recent state-of-the-art RGB trackers [2,38,7,43], the Siamese network is still employed as a basic paradigm.

2.2 3D Single Object Tracking

Early 3D SOT methods [30,3,24,28,25] mainly focus on the RGB-D tracking scenario. As a pioneer in point cloud tracking, SC3D [17] generates candidates for template-search comparison by the Kalman filtering and introduces a shape completion network to retain geometric information in point features. However, SC3D is time-consuming and not end-to-end trainable. P2B [33] is proposed as the first end-to-end point cloud tracker based on pair-wise feature comparison and voting-based region proposal. It enhances search feature with target information in template and adapts SiamRPN [27] to 3D cases, thereby achieving significant performance improvement. 3D-SiamRPN [15] is another tracker inspired by SiamRPN and exhibits good generalization ability. MLVSNet [40] performs Hough voting on multi-level features to retain more useful information. BAT [46] presents a box-aware feature named BoxCloud that is robust to sparseness and incompleteness of point clouds. Benefiting from BoxCloud comparison and feature fusion, BAT exhibits state-of-the-art performance in point cloud SOT. V2B [22] builds a Siamese voxel-to-BEV tracker that regresses target center from the dense bird's eye view (BEV) feature map in an anchor-free manner. Nevertheless, none of these methods pays attention to the fine-grained template-search matching other than the calculation of some feature distance.

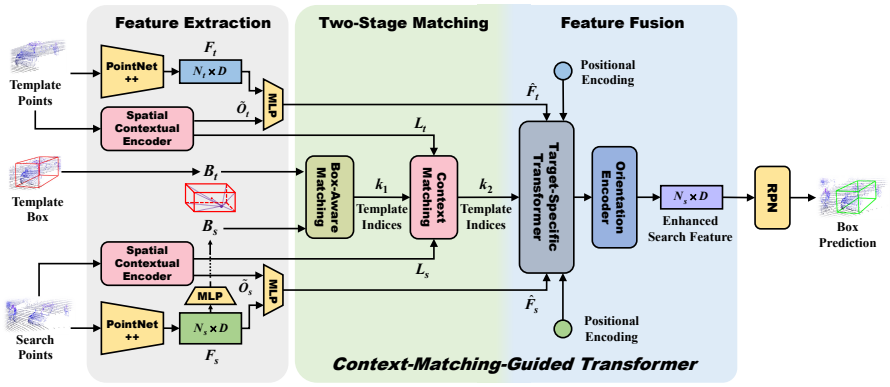

Fig. 2. Pipeline of the proposed CMT tracker. Multiple features are fed into the context-matching-guided transformer for template-search matching and context-aware feature fusion. Finally, a 3D RPN is exploited to generate the target proposal.

2.3 Transformer for Point Cloud Analysis

Since the proposal of transformer [37], such networks using attention mechanism for global awareness have achieved great success in both natural language processing and computer vision [13,6,29]. Recently, many methods [7,38,43] in 2D SOT also shed light on the transformer and show excellent performance. In the 3D domain, Point Transformer [45] is proposed with the vector attention mechanism, improving the performance in point cloud classification and segmentation tasks. Only a few works apply transformer in point cloud SOT. PTT [34] utilizes transformer in voting and proposal generation stage of P2B [33], but does not improve the quality of template-search feature augmentation. LTTR [9] employs transformer framework for feature fusion, but the reported performance is not satisfying due to the design of scalar attention. To this end, we propose a target-specific transformer guided by context matching for feature enhancement, which achieves significant performance improvement in point cloud SOT.

3 Method

Given the initial template point cloud P_t of a target, our CMT tracker localizes it with an input search point cloud P_s for each frame and outputs a 3D bounding box. As depicted in Fig. 2, the proposed method includes three main steps: feature extraction, two-stage template-search matching, and context-aware feature fusion. We will elaborate them in the following sections.

3.1 Feature Extraction

For efficient subsequent processing, four types of features are obtained for template and search point clouds, respectively: 1) *seed feature* denoted by $F_t =$

$\left\{f_t^i \in \mathbb{R}^D\right\}_{i=1}^{N_t}$ and $F_s = \left\{f_s^i \in \mathbb{R}^D\right\}_{i=1}^{N_s}$ (N_t and N_s are the numbers of extracted seed points); 2) *BoxCloud feature* denoted by $B_t = \left\{b_t^i \in \mathbb{R}^9\right\}_{i=1}^{N_t}$ and $B_s = \left\{b_s^i \in \mathbb{R}^9\right\}_{i=1}^{N_s}$; 3) *local contextual descriptor* denoted by $L_t = \left\{l_t^i \in \mathbb{R}^{k_c \times 3}\right\}_{i=1}^{N_t}$ and $L_s = \left\{l_s^i \in \mathbb{R}^{k_c \times 3}\right\}_{i=1}^{N_s}$; 4) *orientational contextual feature* denoted by $\widetilde{O}_t = \left\{\tilde{o}_t^i \in \mathbb{R}^3\right\}_{i=1}^{N_t}$ and $\widetilde{O}_s = \left\{\tilde{o}_s^i \in \mathbb{R}^3\right\}_{i=1}^{N_s}$. Among them, F_t and F_s are extracted by a shared PointNet++ [32]. B depicts the distances from a point to the corners and center of the target bounding box [46]. Specifically, B_t is calculated directly from the template bounding box, while B_s is predicted by an MLP under the supervision of the ground-truth bounding box. The latter two features are produced by the spatial contextual encoder to describe the local context.

Spatial Contextual Encoder (SCE). In tracking, the rotations of targets are usually inevitable. However, many features directly learned from the input points are orientation-sensitive. Networks like PointNet++ [32] achieve geometric transforming invariance through data augmentation, which is difficult to recreate in 3D tracking. BoxCloud proposed in [46] is an interpretable feature invariant to rotations, since it measures the distance between a point and the 3D bounding box. Nevertheless, BoxCloud only focuses on individual points, ignoring the rich contextual information carried by the distribution of nearby points. To this end, we propose a spatial contextual encoder (SCE) to produce point-wise features with the following attributes: 1) invariant to horizontal rotations commonly encountered in 3D SOT; 2) able to effectively describe the spatial context of a point; 3) highly interpretable and easily inferred from point coordinates.

As illustrated in Fig. 3 (a), the input point coordinates are first fed into two polar transform blocks to construct the *local contextual descriptor* $L = \left\{l^i \in \mathbb{R}^{k_c \times 3}\right\}_{i=1}^{N}$ and the *octant contextual descriptor* $O = \left\{o^i \in \mathbb{R}^{8 \times 3}\right\}_{i=1}^{N}$. Both l and o are a set of polar vectors representing the spatial context. The only difference between them is the way of selecting neighboring points. Local polar transform block searches for k_c nearest neighbors, while octant polar transform block looks for the nearest neighbor in each of eight octants, which is more informative than a set of homogeneous points in only a few directions.

For a point i, after selecting its k ($k = k_c$ for local and $k = 8$ for octant) neighbors based on Euclidean distance, the 3D polar vectors expressed by $\left\{\rho_j^i = (r_j^i, \alpha_j^i, \beta_j^i)\right\}_{j=1}^{k}$ are obtained using the following formulas [14]:

$$r_j^i = \sqrt{{x_j^i}^2 + {y_j^i}^2 + {z_j^i}^2}\,, \tag{1}$$

$$\alpha_j^i = \arctan\frac{y_j^i}{x_j^i} - \arctan\frac{\bar{y^i}}{\bar{x^i}}\,, \tag{2}$$

$$\beta_j^i = \arctan\frac{z_j^i}{\sqrt{{x_j^i}^2 + {y_j^i}^2}} - \arctan\frac{\bar{z^i}}{\sqrt{\bar{x^i}^2 + \bar{y^i}^2}}\,, \tag{3}$$

where (x_j^i, y_j^i, z_j^i) is the relative coordinate of neighbor j in the Cartesian coordinate system with point i as the origin, and $(\bar{x^i}, \bar{y^i}, \bar{z^i})$ denotes the mean values of

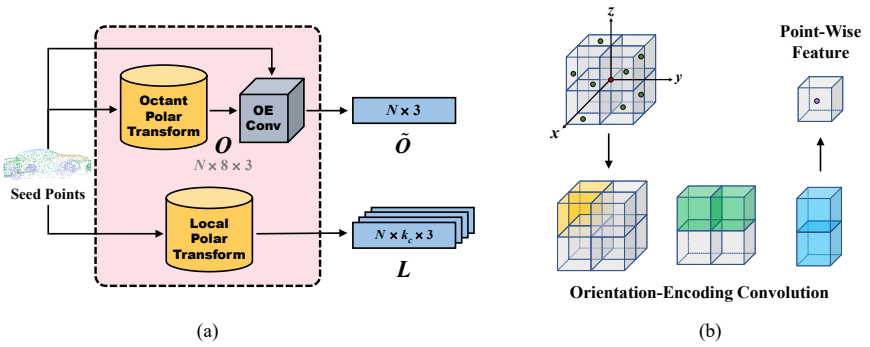

Fig. 3. (a) Architecture of spatial contextual encoder (SCE). Octant contextual descriptor O and local contextual descriptor L are extracted using different ways of selecting neighboring points. O is then further integrated by orientation-encoding convolution to produce the orientational contextual feature $\widetilde{O}$. **(b) Illustration of orientation encoder (OE).** A three-stage convolution along each of three axes is performed successively on the features of neighbors from all eight octants.

neighboring relative coordinate, which is termed as center-of-mass point. Benefiting from the introduction of mass center, the polar vectors remain unchanged when point clouds rotate around z-axis in a 3D Cartesian coordinate system. Center-of-mass point can also effectively reduce the randomness caused by the downsampling before and reflect the general picture of the spatial context [14].

Due to the non-differentiable selecting operation in template-search comparison, we use the directly calculated $l = (\rho_1, \rho_2, \cdots, \rho_{kc}) \in \mathbb{R}^{k_c \times 3}$ to be a criteria in the context matching module, which is further discussed in Section 3.2. Moreover, in order to make use of such informative rotation-invariant feature in target proposal, we further process $o \in \mathbb{R}^{8\times 3}$ with orientation-encoding convolution [23] to integrate information from all eight directions (Fig. 3 (b)), which is detailed interpreted in Section 3.3. After that, the output $\tilde{o} \in \mathbb{R}^3$ is fed into MLPs along with the seed feature $f \in \mathbb{R}^D$ to produce $\hat{f} \in \mathbb{R}^D$.

3.2 Two-Stage Template-Search Matching

Template-search matching is a fundamental operation in Siamese trackers. Instead of comparing the distance between seed features, we introduce a more promising method composed of two stages: box-aware and context matching.

Box-Aware Matching. Considering the success of BoxCloud comparison in BAT [46], we first adopt a box-aware matching based on pairwise L_2 distance between the BoxCloud B_t and B_s. Thereby k_1 nearest template points are selected for each search point. Box-aware matching is effective for extremely sparse point clouds, where meaningful neighboring points can hardly be found. However, template-search matching relying only on BoxCloud may lead to mismatch for targets that are less distinctive in shape and size (*e.g.*, cyclists), since BoxCloud features focus on the relative position of the individual point inside a

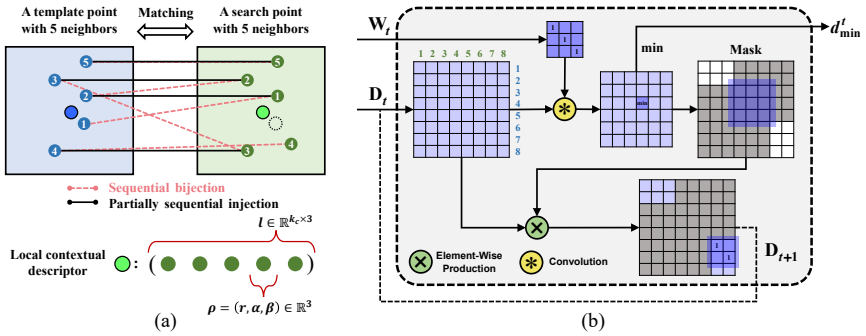

Fig. 4. (a) An exemplified 2D illustration of good template-search matching. In both template and search area, there exist 5 neighbors sorted by their distances to the center seed. Due to target moving or noise, the nearest point in template disappears in search area (dashed point), and a new point (No. 4) appears in exchange. In this case, the reasonable matching (black solid lines) has misalignment with some points ignored. **(b) Illustration of shifted-window context matching.** Rows and columns of $\mathbf{D}_t$ refer to template and search neighbors, respectively. At every stage, the masked output $\mathbf{D}_{t+1}$ serves as the input of the next stage, until all the windows are used. For $k_c = 8$ here, the sizes of two windows are set to 3 and 2. After two stages of convolutions, two matching pairs are found: template $(3, 4, 5)$ and search $(4, 5, 6)$, template $(6, 7)$ and search $(7, 8)$. Finally, the seed-wise contextual distance is calculated as $\hat{d} = d^1_{\min} + d^2_{\min}$.

bounding box, yet ignoring the spatial context. Therefore, in our work, the box-aware matching acts as a robust and efficient base for the following fine-grained context matching. In this way, two stages of matching benefit from each other.

Shifted-Window Context Matching. Among the k_1 template points selected for each search point by the coarse box-aware matching, we adopt the context matching to further select k_2 template points in a fine-grained manner using the local contextual descriptors l_t and $l_s \in \mathbb{R}^{k_c \times 3}$. Therefore, it becomes a problem how to exploit these two sets of polar vectors to describe the similarity or distance between two points' local contexts. Points selection is not a differentiable operation, and there is no supervision available here. Therefore, no learnable parameters can be introduced, which means the comparison is directly performed between $2 \times k_c$ polar vectors. However, we cannot calculate the summation of distances directly between vectors in l_t and l_s, since the sequential bijection between the neighbors sorted by distance is unreasonable, as is illustrated by the pink dashed lines in Fig. 4 (a). In this case, we first define the distance between any two polar vectors ρ_m and ρ_n as

$$d_{mn} \triangleq \sqrt{r_m^2 + r_n^2 - 2r_m r_n \left(\cos\beta_m \cos\beta_n \cos\left(\alpha_m - \alpha_n\right) + \sin\beta_m \sin\beta_n\right)}, \quad (4)$$

which is derived from the Euclidean distance in spherical coordinate system.

For a template point and a search point, by calculating the distance between each template-search neighbor pair using Eq. (4), we obtain $d_{mn}, \forall m, n \in [1, k_c]$,

and the distance matrix is constructed as $\mathbf{D} = (d_{mn})_{k_c \times k_c}$. We then aim at designing a matching algorithm mapping $\mathbf{D}$ to a scalar $\hat{d}$ that describes the distance between two distributions of neighbors. Considering the intrinsic defect of bijective matching that some of the matched pairs are meaningless due to target moving or noise, we propose the shifted-window matching strategy based on the assumption that partially sequential injective correspondences exist in similar spatial contexts, with noise points inserted between them (Fig. 4 (a)).

As illustrated in Fig. 4 (b), given a distance matrix $\mathbf{D}_{k_c \times k_c}$, a series of identity matrices (windows) denoted by $\{\mathbf{W}_t\}$ are generated with empirical sizes $S = \{s_t\}$ according to k_c. For example, when $k_c = 16$, we set $S = \{4, 3, 3\}$, which indicates that 10 of the neighbors have partially sequential injective matching solution, while the remaining 6 neighbors are considered as noise. Then we perform a 2D convolution on $\mathbf{D}_1 = \mathbf{D}$ using $\mathbf{W}_1$ as the kernel. The result is a distance map for different partial matchings, and its minimum element is denoted by $d^1_{\min}$. After that, we mask $\mathbf{D}_1$ to construct $\mathbf{D}_2$, so that only the top-left and bottom-right area of the window producing $d^1_{\min}$ is available for the next convolution using $\mathbf{W}_2$. The masking operation prevents unreasonable crossed matching. For example, once template neighbors $(3, 4, 5)$ have already been matched with search neighbors $(4, 5, 6)$, in no case should template neighbors 1 and 2 be matched with search neighbors 7 or 8. During convolutions for $t = 2, 3, \cdots$, if no area is available in $\mathbf{D}_t$ after masking, $d^t_{\min}$ is set to $d^1_{\max}$ to show that this is not a good matching. After finishing convolutions with all $\mathbf{W}_t$, we define that $\hat{d} = \sum_t d^t_{\min}$. Once $\hat{d}$ is obtained for all template-search pairs, k_2 contextually nearest template points are selected for each search point as the result of this matching stage.

3.3 Matching-Guided Feature Fusion

After the coarse-to-fine matching, we develop a target-specific transformer to perform feature fusion among the matched template-search pairs to exploit the target-aware information for a better target proposal. Then a spatial-aware orientation encoder is introduced to further enhance the attentional feature by integrating spatial information from all directions.

Target-Specific Transformer. As shown in Fig. 5, we adopt the vector attention mechanism [44] to enhance the search feature with attention to relevant template feature that contains potential target information useful for region proposal. Given the template feature $\widehat{F}_t = \left\{\hat{f}^i_t\right\}_{i=1}^{N_t}$, the search feature $\widehat{F}_s = \left\{\hat{f}^i_s\right\}_{i=1}^{N_s}$ and k_2 relevant template indices for each search point selected by box-aware and context matching, the relevant template features $\widehat{F}^*_t = \left\{\left\{\hat{f}^j_{t,i}\right\}_{j=1}^{k_2}\right\}_{i=1}^{N_s}$ are first gathered. Then with three different MLPs, the embedding Q (Query) is generated from $\widehat{F}_s$, while K (Key) and V (Value) are generated from $\widehat{F}^*_t$.

Generally, the standard transformer uses scalar dot-product attention to obtain the relationship (attention map) between Q and K expressed as $A = \mathrm{Softmax}(Q^T K)$. Instead of that, we adopt the subtraction operation with an

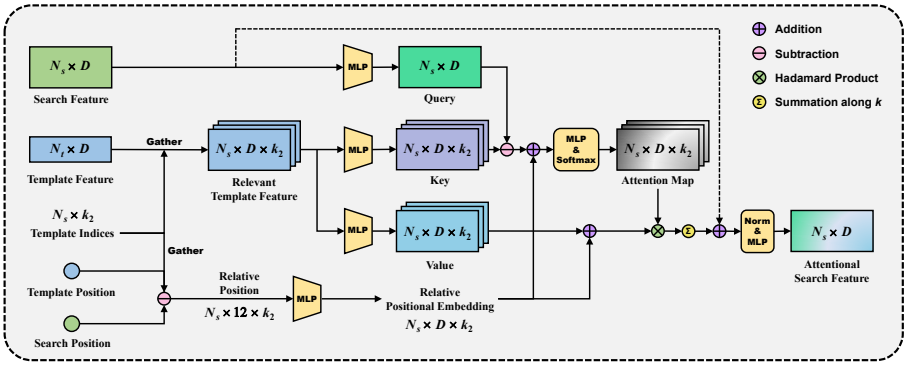

Fig. 5. Illustration of the target-specific transformer. k_2 relevant template points are gathered and aggregated with each search point. The positional encoding is learned from 3D coordinates and 9D BoxCloud. Note that the dimensionality of embedding space is set to be the same as that of the input feature space (D).

extra MLP to improve the fitting ability:

$$A = \text{Softmax}(\text{MLP}(Q - K)) . \tag{5}$$

It is reported in [45] that such vector attention is more suitable for point clouds than scalar attention, since it supports adaptive modulation of individual feature channels in V, rather than a shared scalar weight.

Positional Encoding. Positional encoding plays an important role in transformer. For point clouds, 3D coordinates themselves are naturally suitable for positional encoding. Meanwhile, the success of BoxCloud [46] implies that, the manually crafted 9D feature indicating a point's distance to the bounding box can be another candidate. Therefore, we integrate point coordinates and BoxCloud feature to generate a trainable relative positional encoding expressed as

$$PE = \text{MLP}\left((P_s, B_s) - (P_t^*, B_t^*)\right) , \tag{6}$$

where P_t^* and B_t^* are the coordinates and BoxCloud feature of k_2 relevant template points for each search point, and $(\cdot, \cdot)$ denotes the concatenation operation. We follow [45] to add PE to both $(Q - K)$ and V, since position encoding is important for both attention generation and feature transformation. Thereby Eq. (5) can be rewritten as

$$A = \text{Softmax}(\text{MLP}(Q - K + PE)) . \tag{7}$$

A is then used as a channel-wise weight for V. Normalization and another MLP are applied with a skip connection from the input $\widehat{F}_s$, producing the attentional target-aware feature $\widehat{F}_a = \left\{\hat{f}_a^i\right\}_{i=1}^{N_s}$:

$$\widehat{F}_a = \mathrm{MLP}(\mathrm{Norm}(\sum_{j=1}^{k_2} A \odot (V + PE) + F_s)) \,, \tag{8}$$

where $\odot$ means Hadamard product; $\mathrm{Norm}(\cdot)$ means Instance Normalization [36].

Spatial-Aware Orientation Encoder (OE). Apart from fusing contextually nearby f_t into f_s, we also introduce an orientation encoder (OE) that aggregates features from spatially nearby seeds. As illustrated in Fig. 3 (b), OE first adopts stacked 8-neighborhood (S8N) search to find the nearest neighbor in each of the eight octants partitioned by three axes. If no point exists in some octant within a searching radius, the center point is duplicated as the nearest neighbor of itself. Then we perform a three-stage orientation-encoding convolution [23] on $\hat{f}_a$ of all eight neighbors along each of three axes successively. After squeezing, the output feature vector has the same size as $\hat{f}_a$. In OE, several orientation-encoding convolution blocks are stacked for a more extensive spatial awareness.

4 Experiments

4.1 Experimental Settings

Datasets. To evaluate our tracker, we conduct experiments on three datasets of point clouds scanned by LiDAR sensors, *i.e.*, KITTI [16], nuScenes [5] and WOD [35]. The KITTI dataset contains 21 outdoor scenes and 8 types of targets, and we follow [17] to set up the training, valid and test splits. The nuScenes dataset contains 1000 driving scenes across 23 target classes, and we train the models on the *train_track* split of its training set and test on its validation set. For WOD, we extract tracklets from its detection dataset to produce a SOT dataset with three target classes (Vehicle, Pedestrian and Cyclist). To alleviate the issue of extreme class imbalance, we use 10% of the Vehicle and Pedestrian samples to match the magnitude of Cyclist samples for a fair comparison.

Metrics. We follow [17,33,46] to use one-pass evaluation (OPE) [41], measuring *Success* and *Precision* of trackers. Specifically, given a predicted and a ground-truth 3D bounding box, *Success* is defined as the AUC (area under curve) for the percentage of frames where the IoU (intersection over union) between two boxes is within a given threshold as the threshold varies from 0 to 1. *Precision* is defined as the AUC for the percentage of frames where distance between two boxes' centers is within a given threshold as the threshold varies from 0 to 2 m.

Implementation Details. For training, the loss of our tracker is set to $\mathcal{L} = \mathcal{L}_{bc} + \mathcal{L}_{rpn}$ to count both the Huber loss for BoxCloud [46] and the region-proposal loss [33]. During both training and testing, the search area is formed by enlarging the previous predicted bounding box, and the template is updated for each frame by merging the points inside the first (ground-truth) and the previous predicted box. For template-search matching, we set $k_1 = 16$ for box-aware stage and $k_c = 16$, $k_2 = 4$ for contextual stage. Our model is trained in an end-to-end manner with batch size 192 for 60 epochs using Adam optimizer. Other

Table 1. Performance (Success / Precision) comparison on the KITTI, nuScenes and WOD benchmarks. Bold denotes the best performance.

KITTI					
Category Frames	Car 6424	Pedestrian 6088	Van 1248	Cyclist 308	Average 14068
SC3D [17]	41.3 / 57.9	18.2 / 37.8	40.4 / 47.0	41.5 / 70.4	31.2 / 48.5
P2B [33]	56.2 / 72.8	28.7 / 49.6	40.8 / 48.4	32.1 / 44.7	42.4 / 60.0
3D-SiamRPN [15]	58.2 / 76.2	35.2 / 56.2	45.7 / 52.9	36.2 / 49.0	46.7 / 64.9
MLVSNet [40]	56.0 / 74.0	34.1 / 61.1	52.0 / 61.4	34.3 / 44.5	45.7 / 66.6
BAT [46]	65.4 / 78.9	45.7 / 74.5	52.4 / **67.0**	33.7 / 45.4	55.0 / 75.2
V2B [22]	70.5 / 81.3	48.3 / 73.5	50.1 / 58.0	40.8 / 49.7	58.4 / 75.2
PTT [34]	67.8 / 81.8	44.9 / 72.0	43.6 / 52.5	37.2 / 47.3	55.1 / 74.2
LTTR [9]	65.0 / 77.1	33.2 / 56.8	35.8 / 45.6	**66.2** / **89.9**	48.7 / 65.8
CMT (Ours)	**70.5** / **81.9**	**49.1** / **75.5**	**54.1** / 64.1	55.1 / 82.4	**59.4** / **77.6**
nuScenes					
Category Frames	Car 64159	Truck 13587	Trailer 3352	Bus 2953	Average 84051
SC3D [17]	22.3 / 21.9	30.7 / 27.7	35.3 / 28.1	29.4 / 24.1	24.4 / 23.2
P2B [33]	38.8 / 43.2	43.0 / 41.6	49.0 / 40.1	33.0 / 27.4	39.7 / 42.2
BAT [46]	40.7 / 43.3	45.3 / 42.6	52.6 / 44.9	35.4 / 28.0	41.8 / 42.7
V2B*[22]	36.5 / 38.8	40.8 / 36.7	48.2 / 39.9	31.4 / 26.1	37.5 / 38.1
PTT [34]	40.2 / 45.8	46.5 / 46.7	51.7 / 46.5	39.4 / 36.7	41.6 / 45.7
CMT (Ours)	**47.0** / **51.7**	**52.5** / **52.5**	**62.0** / **58.2**	**46.3** / **42.9**	**48.5** / **51.8**
Waymo Open Dataset					
Category Frames	Vehicle 142664	Pedestrian 58497	Cyclist 13340	Average 214501	
SC3D*[17]	37.4 / 46.0	24.4 / 37.7	26.3 / 36.5	33.2 / 43.1	
P2B*[33]	46.4 / 53.9	34.8 / 54.4	31.5 / 47.8	42.3 / 53.7	
BAT*[46]	50.4 / 57.6	36.2 / 56.3	32.6 / 50.7	45.4 / 56.8	
CMT (Ours)	**53.5** / **62.1**	**40.2** / **62.2**	**34.1** / **53.1**	**48.7** / **61.6**	

* Reproduced with the official code provided by the authors.

hyperparameters are consistent with the settings of [46]. All the experiments are conducted using NVIDIA GTX 1080Ti GPUs.

4.2 Comparison Results

Comparison with State-of-the-Art Methods. We compare our tracker with existing 3D SOT methods [17,33,15,40,46,22,34,9] on the KITTI dataset. Tracking on nuScenes and WOD is more challenging, since much more distractors exist in a frame. For these two datasets, some methods are not included, since

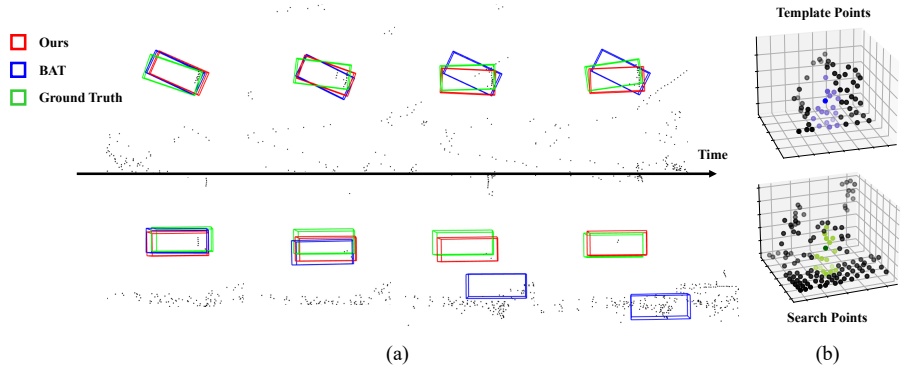

Fig. 6. (a) Advantage cases of our method compared with BAT on KITTI-Car. The upper tracklet belongs to a scene where the target car is turning a corner, and the lower tracklet targets another car in the same scene. **(b) Visualization of context matching.** The blue and the green points are a template and a search seed, respectively, surrounded by their k_c nearest neighbors with lighter colors.

they are either not competitive enough against our baseline BAT [46], or do not provide open source code and have only reported their results on KITTI.

Table 1 summarizes the results on all three datasets. Our method shows significant advantage over the competitors on most categories on KITTI and all categories on nuScenes and WOD. For nuScenes, CMT even outperforms BAT by over 16% in Success and 21% in Precision on average. Notably, some methods [34,22] work well on the car category of KITTI, but struggle in other categories. In contrast, with the carefully-designed context-matching-guided transformer, CMT well adapts to different scales of samples with promising results.

Visualization and Qualitative Analysis. As shown in Fig. 6 (a), we visualize CMT and BAT [46] on the car category of KITTI. In the upper case, the target car is turning a corner. Our CMT tracker captures the rotation accurately, while BAT makes mistakes in the orientation of bounding box. In the lower case, BAT fails to keep tracking due to a distractor, but our method works well consistently. We also visualize a case of template-search matching in Fig. 6 (b). Despite the difference in point quantity and the existence of noise (in this case, plenty of ground points are included in the search area), our method can capture the contextual similarity between template and search area, mark the two seed points as relevant and aggregate their features for a better target proposal, which is mainly attributed to our design of context encoding and matching strategy.

4.3 Ablation Study

In order to validate the design of our CMT tracker, we conduct comprehensive ablative experiments on the truck class of nuScenes, which is much larger than any category of KITTI and can produce more stable results.

Table 2. Results of different ablations. In the bottom part, our baseline BAT and the best results of CMT are presented.

	Method	Success	Precision
Components	BAT + Context Matching	46.8	44.5
	BAT + Transformer	48.7	49.2
	BAT + Transformer + OE	49.9	50.7
	CMT without $\widetilde{O}$	51.6	52.0
Inputs of Transformer	P & B as feature	50.2	50.7
	P as PE, B as feature	51.7	51.9
	P as PE without B	50.9	51.3
	B as PE without P	51.5	51.6
Baseline & Best	BAT	45.3	42.6
	CMT (Ours)	**52.5**	**52.5**

Effectiveness of Components. To illustrate the effectiveness of the components in CMT tracker, four ablative settings are applied: 1) BAT + Context Matching: we enhance BAT with our two-stage template-search matching module; 2) BAT + Transformer: instead of BAFF [46], we use the transformer for feature fusion; 3) BAT + Transformer + OE: everything in CMT is used except context matching; 4) CMT without $\widetilde{O}$: the orientational contextual feature is not integrated. As shown in the upper part of Table 2, BAT equipped with our context matching module defeats the original BAT, which confirms that a better template-search matching helps to improve the quality of proposal feature. Moreover, the result of CMT without $\widetilde{O}$ implies that the proposed spatial contextual encoder can mine useful clues in the orientational context of points, and the third setting further proves the effectiveness of our orientation encoder.

The Choice of k_c and k_1. k_c decides the number of neighbors that we use to construct L in the spatial contextual encoder. A larger k_c means a more extensive contextual awareness and additional computational overhead caused by a larger-scale context matching problem. Moreover, a global context matching is quite expensive. Therefore, we only compare k_1 template points selected by box-aware matching for each search seed. That means with a larger k_1, the context matching plays a more important role and the overhead is also increased.

According to the experiment results in Fig. 7 (a), when k_c is too small, the constructed local contextual descriptor is not discriminative enough for matching. The performance also declines when $k_c = 20$, probably because more noise points are included and the designed shifted windows fail to filter all of them out. Meanwhile, Fig. 7 (b) demonstrates that performance is improved when we set a larger k_1, which further indicates the effectiveness of context matching on the basis of box-aware matching. In general, the best performance is achieved with the setting $k_c = k_1 = 16$. Besides, k_1's effect to the inference speed is larger than k_c's. However, considering the performance improvement, the drop of FPS from 41.7 ($k_1 = 4$) to 32 ($k_1 = 16$) is acceptable since it is still real-time level.

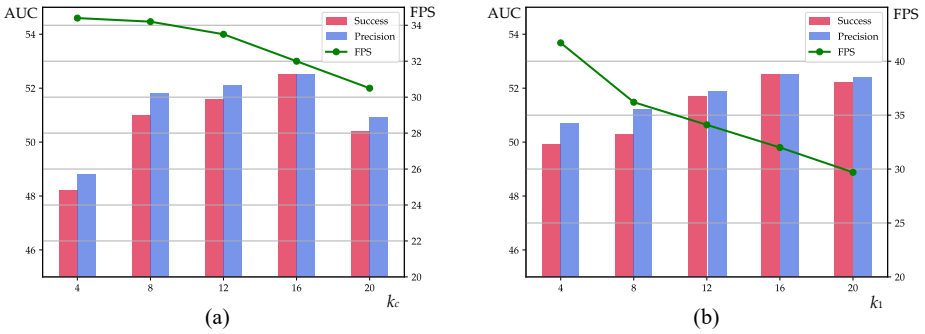

Fig. 7. (a) Comparison between choices of k_c ($k_1 = 16$). Specific window sizes are designed for each k_c. **(b) Comparison between choices of k_1 ($k_c = 16$).** Note that when $k_1 = 4$, the context matching module is not working since k_2 is also 4.

Inputs of Transformer. The feature aggregation submodule in [46] takes the 3D coordinates P and the 9D BoxCloud B as part of the input features, while in our transformer, P and B are concatenated to learn a trainable positional encoding (PE). Taking these into consideration, we test different settings for the inputs of the transformer, including 1) P and B as feature without PE; 2) P as PE while B as feature; 3) P as PE without using B; 4) B as PE without using P. As shown in the middle part of Table 2, the absence of PE leads to a slump in performance, while the introduction of B significantly improves the performance, especially when B plays the role of PE along with P, which demonstrates that BoxCloud contains useful information for location discrimination.

5 Conclusions

In this paper, we propose a context-matching-guided transformer (CMT) tracker for 3D SOT on LiDAR-based point clouds. A horizontally rotation-invariant spatial contextual descriptor, as well as a novel shifted-window matching strategy, is designed to effectively measure the contextual similarity between the template and the search area. Then we introduce a transformer to aggregate the target-aware information from the most contextually relevant template points into the search area with vector attention mechanism. Furthermore, we develop an orientation encoder to integrate spatial information from all directions. Extensive experiments on KITTI, nuScenes and WOD demonstrate that our tracker achieves promising improvement compared with previous state-of-the-art methods.

Acknowledgements. This work was supported by the National Natural Science Foundation of China under Contract U20A20183, 61836011 and 62021001. It was also supported by the GPU cluster built by MCC Lab of Information Science and Technology Institution, USTC.

References

1. Bertinetto, L., Valmadre, J., Henriques, J.F., Vedaldi, A., Torr, P.H.S.: Fully-convolutional siamese networks for object tracking. In: Proceedings of the European Conference on Computer Vision Workshops. pp. 850–865 (2016)
2. Bhat, G., Danelljan, M., Gool, L.V., Timofte, R.: Learning discriminative model prediction for tracking. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 6182–6191 (2019)
3. Bibi, A., Zhang, T., Ghanem, B.: 3d part-based sparse tracker with automatic synchronization and registration. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1439–1448 (2016)
4. Bolme, D.S., Beveridge, J.R., Draper, B.A., Lui, Y.M.: Visual object tracking using adaptive correlation filters. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2544–2550 (2010)
5. 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: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 11621–11631 (2020)
6. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: Proceedings of the European Conference on Computer Vision. pp. 213–229. Springer (2020)
7. Chen, X., Yan, B., Zhu, J., Wang, D., Yang, X., Lu, H.: Transformer tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 8126–8135 (2021)
8. Chen, Z., Zhong, B., Li, G., Zhang, S., Ji, R.: Siamese box adaptive network for visual tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 6667–6676 (2020)
9. Cui, Y., Fang, Z., Shan, J., Gu, Z., Zhou, S.: 3d object tracking with transformer. arXiv preprint arXiv:2110.14921 (2021)
10. Danelljan, M., Bhat, G., Khan, F.S., Felsberg, M.: Atom: Accurate tracking by overlap maximization. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4660–4669 (2019)
11. Danelljan, M., Häger, G., Khan, F., Felsberg, M.: Accurate scale estimation for robust visual tracking. In: Proceedings of the British Machine Vision Conference (2014)
12. Danelljan, M., Shahbaz Khan, F., Felsberg, M., Van de Weijer, J.: Adaptive color attributes for real-time visual tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1090–1097 (2014)
13. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)
14. Fan, S., Dong, Q., Zhu, F., Lv, Y., Ye, P., Wang, F.Y.: SCF-Net: Learning spatial contextual features for large-scale point cloud segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 14504–14513 (2021)
15. Fang, Z., Zhou, S., Cui, Y., Scherer, S.: 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)

16. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the KITTI vision benchmark suite. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3354–3361 (2012)
17. Giancola, S., Zarzar, J., Ghanem, B.: Leveraging shape completion for 3d siamese tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1359–1368 (2019)
18. Guo, D., Wang, J., Cui, Y., Wang, Z., Chen, S.: SiamCAR: Siamese fully convolutional classification and regression for visual tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 6269–6277 (2020)
19. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 770–778 (2016)
20. Henriques, J.F., Caseiro, R., Martins, P., Batista, J.: Exploiting the circulant structure of tracking-by-detection with kernels. In: Proceedings of the European Conference on Computer Vision. pp. 702–715. Springer (2012)
21. Henriques, J.F., Caseiro, R., Martins, P., Batista, J.: High-speed tracking with kernelized correlation filters. IEEE Transactions on Pattern Analysis and Machine Intelligence **37**(3), 583–596 (2014)
22. Hui, L., Wang, L., Cheng, M., Xie, J., Yang, J.: 3d siamese voxel-to-BEV tracker for sparse point clouds. In: Advances in Neural Information Processing Systems. vol. 34 (2021)
23. Jiang, M., Wu, Y., Zhao, T., Zhao, Z., Lu, C.: PointSIFT: A SIFT-like network module for 3d point cloud semantic segmentation. arXiv preprint arXiv:1807.00652 (2018)
24. Kart, U., Kamarainen, J.K., Matas, J.: How to make an RGBD tracker? In: Proceedings of the European Conference on Computer Vision Workshops (2018)
25. Kart, U., Lukezic, A., Kristan, M., Kamarainen, J.K., Matas, J.: Object tracking by reconstruction with view-specific discriminative correlation filters. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1339–1348 (2019)
26. Li, B., Wu, W., Wang, Q., Zhang, F., Xing, J., Yan, J.: SiamRPN++: Evolution of siamese visual tracking with very deep networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4282–4291 (2019)
27. Li, B., Yan, J., Wu, W., Zhu, Z., Hu, X.: High performance visual tracking with siamese region proposal network. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 8971–8980 (2018)
28. Liu, Y., Jing, X.Y., Nie, J., Gao, H., Liu, J., Jiang, G.P.: Context-aware three-dimensional mean-shift with occlusion handling for robust object tracking in RGB-D videos. IEEE Transactions on Multimedia **21**(3), 664–677 (2018)
29. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 10012–10022 (2021)
30. Pieropan, A., Bergström, N., Ishikawa, M., Kjellström, H.: Robust 3d tracking of unknown objects. In: IEEE International Conference on Robotics and Automation. pp. 2410–2417. IEEE (2015)
31. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: PointNet: Deep learning on point sets for 3d classification and segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 652–660 (2017)
32. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: PointNet++: Deep hierarchical feature learning on point sets in a metric space. In: Advances in Neural Information Processing Systems. vol. 30 (2017)

33. Qi, H., Feng, C., Cao, Z., Zhao, F., Xiao, Y.: P2B: Point-to-box network for 3d object tracking in point clouds. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 6329–6338 (2020)
34. Shan, J., Zhou, S., Fang, Z., Cui, Y.: PTT: Point-track-transformer module for 3d single object tracking in point clouds. In: IEEE International Conference on Intelligent Robots and Systems. pp. 1310–1316 (2021)
35. Sun, P., Kretzschmar, H., Dotiwalla, X., Chouard, A., Patnaik, V., Tsui, P., Guo, J., Zhou, Y., Chai, Y., Caine, B.: Scalability in perception for autonomous driving: Waymo open dataset. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2446–2454 (2020)
36. Ulyanov, D., Vedaldi, A., Lempitsky, V.: Instance normalization: The missing ingredient for fast stylization. arXiv preprint arXiv:1607.08022 (2016)
37. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L.u., Polosukhin, I.: Attention is all you need. In: Advances in Neural Information Processing Systems. pp. 5998–6008 (2017)
38. Wang, N., Zhou, W., Wang, J., Li, H.: Transformer meets tracker: Exploiting temporal context for robust visual tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1571–1580 (2021)
39. Wang, Q., Zhang, L., Bertinetto, L., Hu, W., Torr, P.H.: Fast online object tracking and segmentation: A unifying approach. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1328–1338 (2019)
40. Wang, Z., Xie, Q., Lai, Y.K., Wu, J., Long, K., Wang, J.: MLVSNet: Multi-level voting siamese network for 3d visual tracking. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 3101–3110 (2021)
41. Wu, Y., Lim, J., Yang, M.H.: Online object tracking: A benchmark. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2411–2418 (2013)
42. Xu, Y., Wang, Z., Li, Z., Yuan, Y., Yu, G.: SiamFC++: Towards robust and accurate visual tracking with target estimation guidelines. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34, pp. 12549–12556 (2020)
43. Yan, B., Peng, H., Fu, J., Wang, D., Lu, H.: Learning spatio-temporal transformer for visual tracking. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 10448–10457 (2021)
44. Zhao, H., Jia, J., Koltun, V.: Exploring self-attention for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 10076–10085 (2020)
45. Zhao, H., Jiang, L., Jia, J., Torr, P., Koltun, V.: Point transformer. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 16259–16268 (2021)
46. 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. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 13199–13208 (2021)
47. Zhu, Z., Wang, Q., Li, B., Wu, W., Yan, J., Hu, W.: Distractor-aware siamese networks for visual object tracking. In: Proceedings of the European Conference on Computer Vision. pp. 101–117 (2018)