2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Enhancing 3D Single Object Tracking with Efficient Point Cloud Segmentation

Yushi Yang, Baojie Fan*, Yuyu Jiang, Wuyang Zhou, Dong Chen, Hongxin Xu

***Abstract*— 3D single object tracking (SOT) based on point cloud has attracted much attention due to its important role in machine vision and autonomous driving. Recently, M^2-Track proposes a two-stage tracking structure centered on motion, but they ignore the effect of segmentation errors in sparse point cloud scenarios, which hinder the ability of networks to accurately represent tracking targets. To solve the problems, we propose an efficient 3D single object tracker (Abbr. EST) that can effectively segment point cloud features. Firstly, the proposed fusion segmentation module makes up for the feature loss caused by the downsampling strategy and enhances the ability of the network to recognize foreground points. In addition, the global embedded module is used to further focus on the crucial features of the target. This module provides global information by using residual networks and adding background information. Numerous experiments conducted on KITTI and NuScenes benchmarks show that EST achieves superior point cloud tracking in both performance and efficiency.**

I. Introduction

Single object tracking in 3D space is essential for a variety of applications, ranging from autonomous driving, mobile robotics and path planning [1]–[3]. Given the ground truth (GT) of the first frame in a tracking sequence, the objective of 3D SOT is to track the three-dimensional pose of the specified target in subsequent frames.

Existing 3D SOT methods [4]–[13] primarily adopt the Siamese paradigm, utilizing a dual-branch structure with shared parameters to extract features from the template and search area. Then, an appearance matching module is employed to model the correlation between them. Finally, the localization of the target is achieved in an end-to-end manner using a localization network such as region proposal network (RPN). Unfortunately, the robustness of this paradigm is limited when there are significant changes in the appearance of the target. Recently, M^2-Track [14] has proposed a motion-centric two-stage tracking paradigm that maintains the consistency of target motion by using point clouds from two consecutive frames as input. It utilizes a segmentation network to obtain foreground points, and in the first stage, predicts the relative motion of the target based on the segmentation results to generate coarse bounding boxes. In the second stage, the coarse boxes are refined to obtain more accurate bounding boxes.

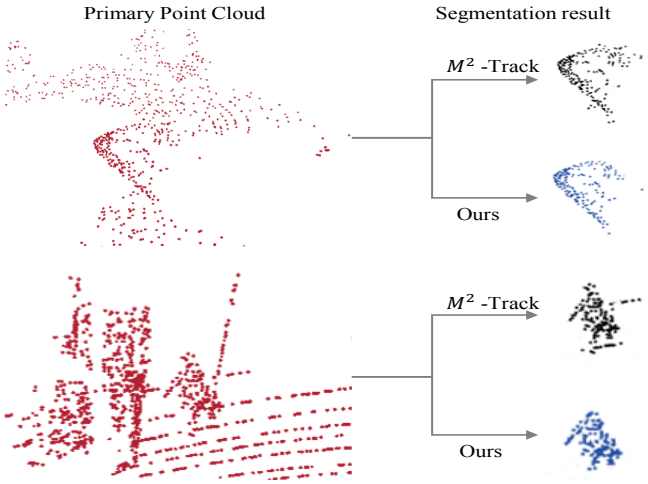

Fig. 1: Comparing the segmentation module of M^2-Track with our model, EST is more accurate in the segmentation of the Car and Pedestrian categories.

The point clouds collected by LiDAR consist of discrete points on the surfaces of objects, and the surrounding background inevitably contains noise. Consequently, the segmentation errors introduced by the motion-centric method pose additional challenges to the accurate identification of the target. To address this issue, we propose a new segmentation method that integrates the PointNet++ segmentation network and our convolutional segmentation (CSE) module, they segment the target foreground point from a local and global perspective, respectively. The proposed CSE module significantly enhances foreground recognition by generating dense point features while compensating for the loss of target features due to the PointNet++ down-sampling strategy. Figure 1 shows the difference between the segmentation results, and our pipeline is shown in Figure 2.

In addition, for the scene where the targets are too sparse, there is still a large error in obtaining the initial target location from a few foreground points. We found that considering the entire scenario makes it easier to identify and locate targets, and the global information provides a comprehensive understanding of the point cloud scenario, so we introduced the global embedding module (GEM) to refine the initial

*This work is supported by the National Natural Science Foundation of China (No.62473205,U2013210, 62103388), and the young and middle-aged leading scholar in Qinglan Project by Jiangsu Province

*corresponding authors. Baojie Fan is with the College of Automation and College, Artificial Intelligence, Nanjing University of Posts and Telecommunications, Nanjing 210023, China jobfbj@gmail.com

Yushi Yang and Dong Chen are with College of Electronic and Optical Engineering, College of Flexible Electronics (Future Technology), Nanjing University of Posts and Telecommunications, Nanjing 210023, China 17855515226@163.com)

Yuyu Jiang and Wuyang Zhou are with the College of Automation and College, Artificial Intelligence, Nanjing University of Posts and Telecommunications, Nanjing 210023, China

Hongxin Xu is with the Delft University of Technology Landbergstraat 152628 CE Delft H.Xu-14@student.tudelft.nl

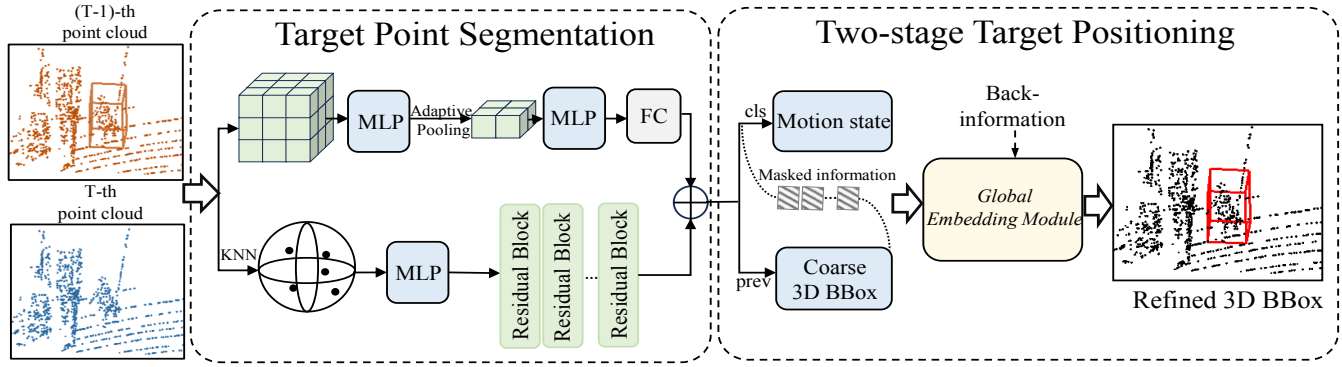

Fig. 2: Overview of our trackers. The network mainly comprises two parts: Firstly, the foreground segmentation fusion module is designed, the foreground extraction is realized through PointNet++, and the segmentation accuracy is further improved by combining the CSE module. After that, we use the attention mechanism to embed context clues in the GEM module to ensure effective dissemination and enhancement of the target information.

positioning box. Specifically, we use the residual network target to provide global information, focus on the crucial features of the target through the cross-attention mechanism, and refine the final target proposal.

In summary, our contributions are summarized as follows:

- We introduce EST, a novel 3D single object tracking tracker, which enhances the foreground recognition capability of the network by fusing segmentation modules.
- The proposed GEM module utilizes an attention mechanism to propagate target information and embed context clues, thereby improving tracking performance.
- Our trackers achieve excellent performance on KITTI [15] and NuScenes [16] benchmark datasets.

II. Related Works

A. 3D Single Object Tracking.

SC3D [5] firstly introduces the Siamese network into 3D SOT, matching a series of target candidates with templates based on feature similarity. However, the heuristic search sampling process is time-consuming and does not allow end-to-end training. P2B [6] utilizes PointNet++ [17] as the backbone and then propagates target cues from the template to the search area through target-specific feature enhancement techniques. Finally, high-quality 3D proposals are generated using Hough voting [18]. BAT [9] introduces a box-aware feature fusion module to enhance correlation learning. V2B [19] proposes a Voxel-Bird's Eye View (BEV) target localization network to address the sparse point cloud problem. PTT [7] PTTR [20] and CMT [21] use attention mechanisms for better matching, C2FT [3] designs a two-stage RPN module for local-to-global feature refinement. CorpNet [22] uses a pyramid structure encoder for multi-scale feature integration. STNet [23] develops an iterative coarse-to-fine correlation network for robust learning. CXTrack [24] introduces a transformer-based 3D object tracking network leveraging continuous information. STTrack [25] proposes a multi-frame feature fusion method for implicit learning of target motion and establishing inter-frame correlation. While these methods have demonstrated excellent performance, point clouds are often incomplete and lack texture to provide sufficient appearance information for matching. In addition, the template clipping method breaks the motion consistency between frames. In this context, M^2-Track [14] proposes a two-stage motion-centric paradigm that splits foreground points and fuses target point clouds across frames to obtain accurate bounding boxes. In this paper, we deeply study the motion tracking paradigm and propose EST, an efficient LiDAR point cloud 3D single-target tracker.

B. Point Cloud Segmentation

3D segmentation is a fundamental and challenging task in computer vision, which is widely used in the fields of autonomous driving, robotics, augmented reality, and medical image analysis. Due to the large amount of point cloud data, MLP-like networks often use k-nearest neighbor sampling or spherical sampling methods to determine the perceived field. In point cloud analysis, PointNet [26] and PointNet++ [17]are pioneering works, models that use Multi-layer perceptrons to extract features and aggregate these features through symmetric functions.PointCNN [27]addresses the issues of shape information loss and order variance in point cloud data by learning X-transformations and applying typical convolution operations. KPConv [28] flexibly utilizes any number of kernel points and learns continuous kernel point positions through the network, enabling adaptive convolution on local geometric shapes. PointSIFT [29] learns local shape information by encoding information in different directions, transforming each point into a new shape representation. PointASNL [30] proposes a local-non-local module with adaptive sampling to capture local and global dependencies of sampling points. PointNorm [31] uses the average of local and global points (that is, all points) to perform normalization operations. RepSurf [32] provides geometric information by matching surface information with triangular planes to model an umbrella surface. PointNeXt [33] introduces a reverse residual bottleneck design and employs a separable MLP for efficient and effective model scaling.

III. METHOD

A. Overview

In the field of point cloud-based SOT, typically, the initial frame (template frame) provides the bounding box (BBox) of the object. The tracker's objective is to continuously track the 3D pose of the object in subsequent frames. Specifically, for the point cloud at time t, we can represent the object's state with seven parameters. These include the center coordinates (x, y, z) of the object, dimensions (l, w, h), and orientation angle θ (assuming the target's rotation is only around the z-axis). Considering that the physical dimensions of rigid objects captured by LiDAR in outdoor scenes usually remain constant, the size of the BBox remains consistent with the initial bounding box P_1 in the tracking process, eliminating the need to infer the target's dimensions (l, w, h).

Given the point cloud P_{t-1} and BBox B_{t-1} from the previous frame (timestamp $t > 1$), along with the current frame's point cloud P_t as our input, we denote the tracking process as a function F:

$$F : \mathbb{R}^{N_t \times 3} \times \mathbb{R}^{N_{t-1} \times 3} \times \mathbb{R}^7 \longmapsto \mathbb{R}^4 \tag{1}$$

$$F\left(P_{t-1}, P_t, B_{t-1}\right) \mapsto (x, y, z, \theta) \tag{2}$$

B. Foreground Segmentation

Unlike appearance-based matching methods, 1024 initial points are randomly sampled from the t frame and t-1 frame. These points are then transformed from the world coordinate system to the object coordinate system. Subsequently, we utilize PointNet++ and the proposed CSE module to segment the foreground points from local and global perspectives. PointNet++ samples the input point cloud, avoiding unnecessary rendering and reducing training time. Specifically, it learns hierarchical features by stacking multiple learning stages, capturing local geometric information, and gradually expanding the receptive field through repeated operations. In each stage, the farthest point sampling (FPS) algorithm is employed to resample points. For each sampled point, the k-nearest neighbors algorithm is used to select neighboring points, and the max pooling aggregation function is applied to capture the local structure. The entire process can be expressed as:

$$g_i = H(\Phi(f_{i,j}) \mid j = 1, \ldots, K) \tag{3}$$

where $H(\cdot)$ represents aggregate functions, usually using maximum pooling operations. $\Phi(\cdot)$ denotes the local feature extraction function, and the common method is to use a multi-layer perceptron to extract features. $f_{i,j}$ is the feature of the j-th neighborhood point of the i-th sampling point.

However, due to the dataset being sourced from real-world road scenes, there are inevitably interference objects similar to the target inevitably exist in the background, and the information lost due to the PointNet++ down-sampling strategy can impede subsequent tracking. To address this issue, we propose a simple and efficient convolutional segmentation module (CSE) instead of using subsampling strategies to aggregate spatial information. Figure 3 depicts the detailed

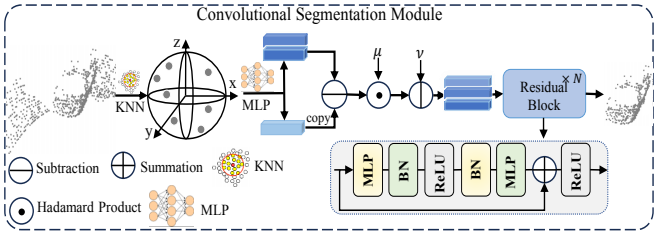

Fig. 3: Overall architecture of the convolutional segmentation module.

structure of the CSE module. Our CSE module can be represented as:

$$g_i = \Phi_\beta(H(\Phi_\alpha(f_{i,j}, \mid j = 1, \cdots, K)) \tag{4}$$

where $\Phi_\alpha(\cdot)$ and $\Phi_\beta(\cdot)$ represent residual blocks. $\Phi_\alpha(\cdot)$ is designed for learning shared weights from a local region, while $\Phi_\beta(\cdot)$ is used to extract deep aggregated features.

Specifically, the mapping function is constructed as a series of residual blocks. Our residual blocks consist of fully connected (FC) layers, normalization, and activation layers. Following the approach of PointNet, the aggregation function $H(\cdot)$ is considered as a max-pooling operation. Equation 4 describes one stage of CSE. For hierarchical and deep networks, this operation is repeated recursively to obtain the deep-geometric features of the target. MLP is naturally invariant for permutations, and the CSE module only uses MLP, so the CSE module is particularly suitable for the characteristics of point clouds. This module can easily be extended to dozens of layers by introducing residual connections, thereby generating deep feature representations. Furthermore, even with the introduction of additional layers, efficient performance can be maintained because we do not have complex extractors and our main operation is highly optimized feedforward MLP.

$$f_{i,j}^{'} = \mu \odot \frac{f_{i,j}^{'} - f_i}{\theta + \delta} + \nu \tag{5}$$

$$\theta = \sqrt{\frac{1}{k \times n \times d} \sum_{i=1}^{n} \sum_{j=1}^{k} (f_{i,j} - f_i)^2} \tag{6}$$

where $f_{i,j}^{'} \mid_{j=1,\cdots,k} \in \mathbb{R}^{k \times d}$ be the grouped local neighbors of $f_i \in \mathbb{R}^d$ containing k points, and each neighbor point $f_{i,j}$ is a d-dimensional vector. $\mu \in \mathbb{R}^d$ and $\nu \in \mathbb{R}^d$ are learnable parameters. $\odot$ indicates Hadamard production, and $\delta = 1e^{-5}$ is a small number for numerical stability. Note that θ is a scalar that describes the feature deviation across all local groups and channels. By this process, the local points are converted into a normal distribution while preserving the original geometric characteristics.

It is noteworthy our segmentation operation collaborates with the original PointNet++ and the CSE module. We consider that the features of the target can be divided into local geometric features and macroscopic geometric features. The proposed CSE module, based on a simple and efficient MLP

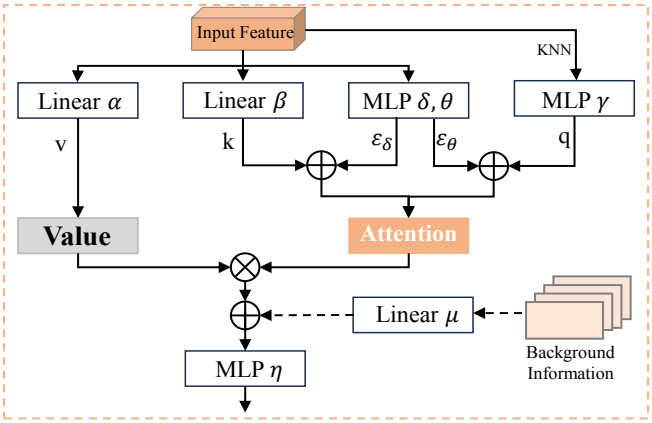

Fig. 4: The structure of the proposed Global Embedding Module.

and aggregation operation, avoids subsampling strategies to attain macroscopic features of the tracked object. Decoupling the segmentation features of the target has enabled a more precise segmentation of foreground points. Finally, the segmentation process of our model can be represented as:

$$F_P = f_p(x), F_C = f_c(x, x^*) \tag{7}$$

$$F_{out} = \alpha F_P + \beta F_C \tag{8}$$

where $x \in \mathbb{R}^{3\times N\times B}$, F_P denotes PointNet++, F_C denotes CSE modules. Component ablation experiments show that our design has been well validated.

C. Global Embedding Module

The structure of the global embed module is shown in Figure 4, which consists of a local interleaved attention and background information embed section. The purpose of cross-focus is to further focus on the important features of the target. Specifically, for each point, we use the attention mechanism to summarize the target local structure by visiting adjacency points. We found that considering the entire scenario makes it easier to identify and locate targets. Therefore, we retain the segmented target background information and optimize the target positioning through the residual connection. In addition, we set a threshold for the insertion of background information to improve the robustness of the network, and the embedding branch of background information in the GEM module will only take effect if the initial input point is below this value.

Firstly, the fusion features and the feature after the KNN operation are used as the input of the GEM module to summarize the context information. Then q is obtained using the non-linear transformation of the multilayer perceptron (MLP). Cross-attention is used to compare the gap between two branches enhances the learning ability of the network. The process of generating Q, K, and V is represented as follows:

$$\begin{aligned} q &= MLP(f_{in}) \\ v = Linear(f_{in}),\ & k = Linear(f_{in}) \end{aligned} \tag{9}$$

In addition, since the sparsity of point clouds can make target recognition more difficult, we enhance queries and key vectors with learnable location embeddings to incorporate location relationships. The process can be expressed as:

$$W_{ik} = \{\mathcal{K}(q_i, k) + \varepsilon_\theta, k_i + \varepsilon_\delta\} \tag{10}$$

$$h = Softmax(W_{ik})v_i \tag{11}$$

where $\mathcal{K}$ represents the KNN algorithm. After the attention mechanism is complete, we add background information to optimize the goal proposal. It can be expressed as:

$$f_{out} = MLP\{Linear(f_{back}) + h\} \tag{12}$$

where f_{back} represents background information, f_{out} is the output of the module.

D. Loss Function

Our model is trained alongside other sub-networks in the M^2-Track framework, following the framework's design principles to construct the loss function. The loss function comprises mainly two components: classification loss and regression loss. In the classification loss, we incorporate L_{seg}, a cross-entropy loss for target segmentation, and L_{cls_mot}, a cross-entropy loss for motion state classification. Therefore, the overall loss function is defined as:

$$L = \lambda_1 L_{seg} + \lambda_2 L_{cls\_mot} + L_{reg\_all} \tag{13}$$

For the regression loss, we adopt Huber loss to quantify the relative error between the model predictions and the ground truth target boxes.$L_{reg\_all} = L_{reg\_motion} + L_{reg\_prev} + L_{reg1} + L_{reg2}$ The regression loss, denoted as L_{reg_all}, is a linear combination of multiple terms, specifically including L_{reg_mot}, used for continuous motion state regression, L_{reg_prev}, employed for the regression of target boxes between the previous timestamp predictions and ground truth, and L_{reg1}/L_{reg2}, utilized for target box regression in stages I and II, respectively. We set the parameters as follow: $\lambda_1 = \lambda_2 = 0.1$.

IV. Experiments

A. Experimental Settings

1) Datasets: We employ the KITTI dataset [15] along with the NuScenes dataset [16] for both training and evaluating our proposed tracker. KITTI comprises 21 scenes of autonomous driving and encompasses 8 object categories. To address the absence of ground truth in the test set, we adopt a similar partitioning strategy as P2B [6]: sequences 0-16 constitute the training set, 17-18 serve as the validation set, and 19-20 are allocated to the test set. Regarding the NuScenes dataset, it consists of 1000 autonomous driving scenes and covers 23 object categories. We specifically report results on key frames due to the lack of labels in other frames.

2) Metrics: We employ One-Pass Evaluation (OPE) to comprehensively assess success and precision. In the experiments below, we provide detailed reports on the success and precision of each model. For a given predicted bounding

TABLE I: EXTENSIVE COMPARISONS WITH SOTA TRACKERS ON KITTI [15] DATASET.

	Category	Car	Pedestrian	Van	Cyclist	Mean
	Frame Number	*6424*	*6088*	*1248*	*308*	*14068*
Success	SC3D [5]	41.3	18.2	40.4	41.5	31.2
	P2B [6]	56.2	28.7	40.8	32.1	42.4
	MLVSNet [34]	56.0	34.1	52.0	34.3	45.7
	BAT [9]	65.4	45.7	52.4	33.7	55.0
	PTT [7]	67.8	44.9	43.6	37.2	55.1
	C2FT [3]	67.0	48.6	53.4	38.0	57.2
	V2B [19]	70.5	48.3	50.1	40.8	58.4
	PTTR [20]	65.2	50.9	52.5	65.1	58.4
	CMT [21]	70.5	49.1	54.1	55.1	59.4
	STNet [23]	72.1	77.2	58.0	73.5	61.3
	M^2-Track [14]	65.5	61.5	53.8	73.2	62.9
	STTrack [25]	66.5	60.4	50.5	75.3	62.6
	Ours	67.6	61.6	60.0	77.9	64.6
Precision	SC3D [5]	72.8	49.6	48.4	44.7	60.0
	P2B [6]	72.8	49.6	48.4	44.7	60.0
	MLVSNet [34]	74.0	61.1	61.4	44.5	66.6
	BAT [9]	78.9	74.5	67.0	45.4	75.2
	PTT [7]	81.8	72.0	52.5	47.3	74.2
	C2FT [3]	80.4	75.6	66.1	48.7	76.4
	V2B [19]	81.3	73.5	58.0	49.7	75.2
	PTTR [20]	77.4	81.6	61.8	90.5	77.8
	CMT [21]	81.9	75.5	64.1	82.4	77.6
	STNet [23]	84.0	77.2	70.6	93.7	80.1
	M^2-Track [14]	80.8	88.2	70.7	93.5	83.4
	STTrack [25]	79.9	89.4	63.6	93.9	82.9
	Ours	83.1	88.5	76.2	94.4	85.1

TABLE II: EXTENSIVE COMPARISONS WITH SOTA TRACKERS ON NUSCENES [16] DATASET.

	Category	Car	Pedestrian	Truck	Trailer	Bus	Mean
	Frame Number	*64159*	*33237*	*13587*	*3352*	*2953*	*117278*
Success	SC3D [5]	22.3	11.3	30.7	35.3	29.4	20.7
	P2B [6]	38.8	28.4	43.0	49.0	33.0	36.5
	BAT [9]	40.7	28.8	45.3	52.6	35.4	38.1
	M^2-Track [14]	55.9	32.1	57.4	57.6	51.4	49.2
	Ours	56.4	36.2	59.6	62.8	59.7	51.3
Precision	SC3D [5]	21.9	12.7	27.7	28.1	24.1	20.2
	P2B [6]	43.2	52.2	41.6	40.1	27.4	45.1
	BAT [9]	43.3	53.3	42.6	44.9	28.0	45.7
	M^2-Track [14]	65.1	60.9	59.5	58.3	51.4	62.7
	Ours	65.9	63.5	62.1	60.9	58.7	64.5

box (BBox) and its corresponding ground truth BBox, we define their Intersection over Union (IoU) as the overlap and the distance between their centers as the error. The success rate represents the percentage of frames where the overlap between the predicted and ground truth BBox exceeds a set threshold. Accuracy indicates the percentage of frames in which the distance between the predicted BBox's center and the ground truth BBox's center is less than a given threshold.

3) Training: The Adam optimizer is used to train our model with a batch size of 64, an initial learning rate of 0.001, and a decay of 10 times every 20 epochs. In practice, EST is found to converge to satisfactory results after approximately 60 periods. All the experiments are conducted using NVIDIA GTX 2080Ti GPUs.

B. Comparison with State-of-the-arts

1) Results on KITTI. As shown in table I, our EST also performs competitively on the KITTI dataset. Compared to the previous state-of-the-art M^2-Track, our method achieved an average success and average precision of over $1.7\%/1.7\%$, respectively. For Van, we achieved the most significant improvement, $6.2\%/5.5\%$ higher than M^2-Track. It is generally believed that the small volume of pedestrians and many interfering objects lead to inaccurate segmentation, which limits the tracking performance of the network. We strip background points, use segmentation fusion module for fine segmentation, and use GEM for robust tracking to achieve balanced performance in all categories. Meanwhile, in Figure 5, we show a comparison between our method and the visualization results of M^2-Track.

TABLE III: SPEED COMPARISON RESULTS OF DIFFERENT METHODS ON KITTI [15] DATASET.

Method	FPS	Success	Precision
P2B [6]	38.8	42.4	60.0
BAT [9]	40.9	55.0	75.2
STNet [23]	30.1	61.3	80.0
M^2-Track [14]	54.8	62.9	83.4
Ours	49.7	64.6	85.1

2) Results on NuScenes. In contrast to the KITTI dataset, which is known for its low density, the NuScenes dataset contains scenes with significant interference and drastic changes in appearance, making object localization extremely challenging. To verify the generalization ability of the proposed method, we applied models such as car and pedestrian trained on the KITTI dataset to the NuScenes dataset for evaluation. As shown in table II, our approach shows clear advantages in the NuScenes dataset. Compared to our baseline M^2-Track [14], EST shows significant improvement, with a 2.1%higher average success and 1.8% higher average precision. Specifically, M^2-Track is less accurate for medium and large objects (truck, trailer), and the prediction center is off target. We performed better on large objects (truck, bus), and the success of trailer and bus improved by 5.2% and 8.3%, respectively. In addition to big goals, our approach can still achieve higher performance for those smaller goals. For pedestrian projects, our approach also significantly improved performance, with Success increasing from 32.1 to 36.2 and precision increasing from 60.9 to 63.5. In Figure 6, we show the results visualized on the NuScenes dataset.

3) Robustness in Sparse Scenes: To evaluate robustness in sparse scenarios, we use the KITTI dataset for statistical analysis of the performance of P2B, PTT, M^2-Track, and our proposed EST tracker. Specifically, based on the number of foreground points of the template frame, we divided the tracking tracks of 120 vehicle categories in the test set into 6 intervals. As shown in Figure 7, our EST method shows excellent performance in sparse scenarios of less than 100 points, outperforming other leading trackers.

4) Inference Time: The running speed is a crucial indicator in the practical application of 3D trackers. Table III shows how we compare our EST with some representative methods, and we test the speed of the tracker using single NVIDIA 2080Ti, which takes about 20 ms to process a frame, achieving a real-time speed of 49.7 fps.

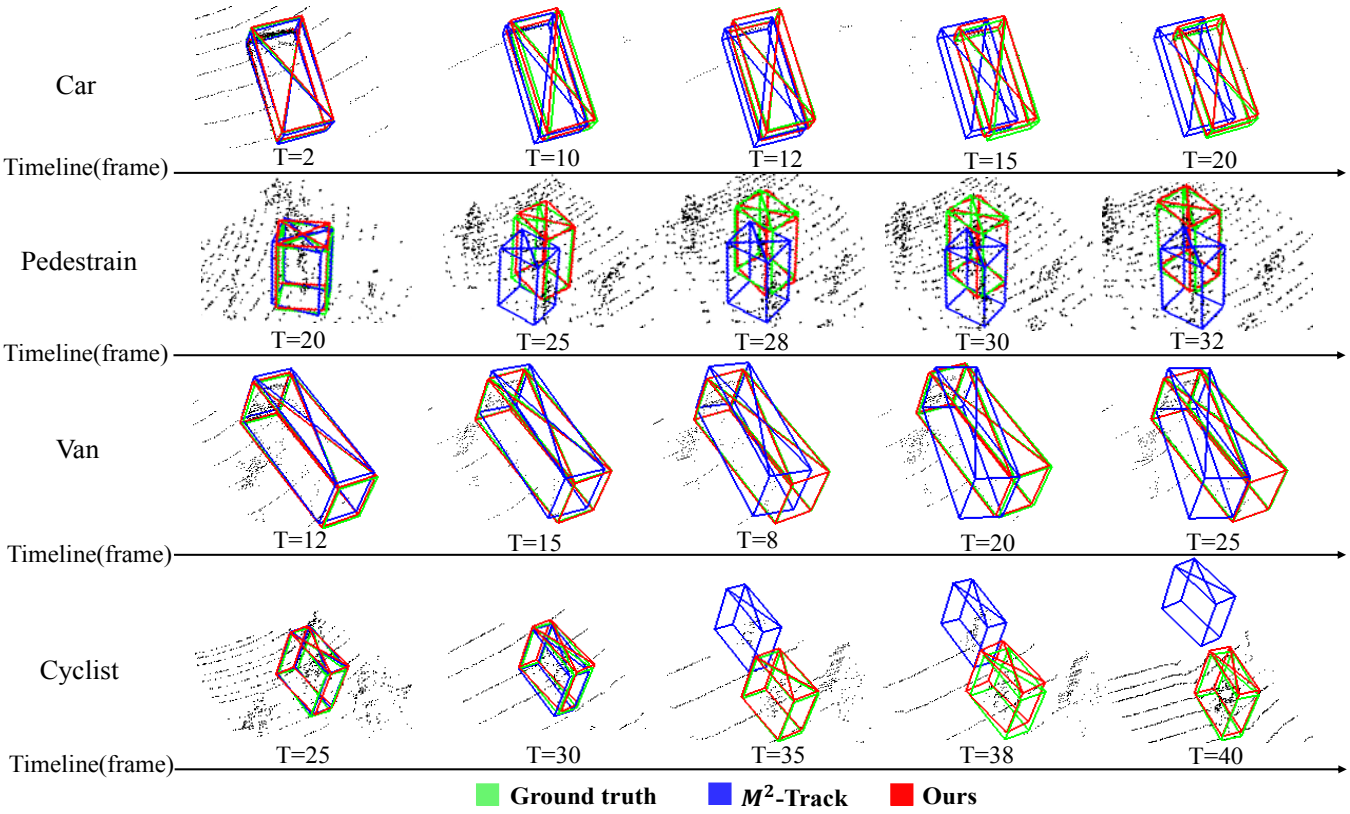

Fig. 5: Result comparison visualization with M^2-Track on the KITTI dataset.

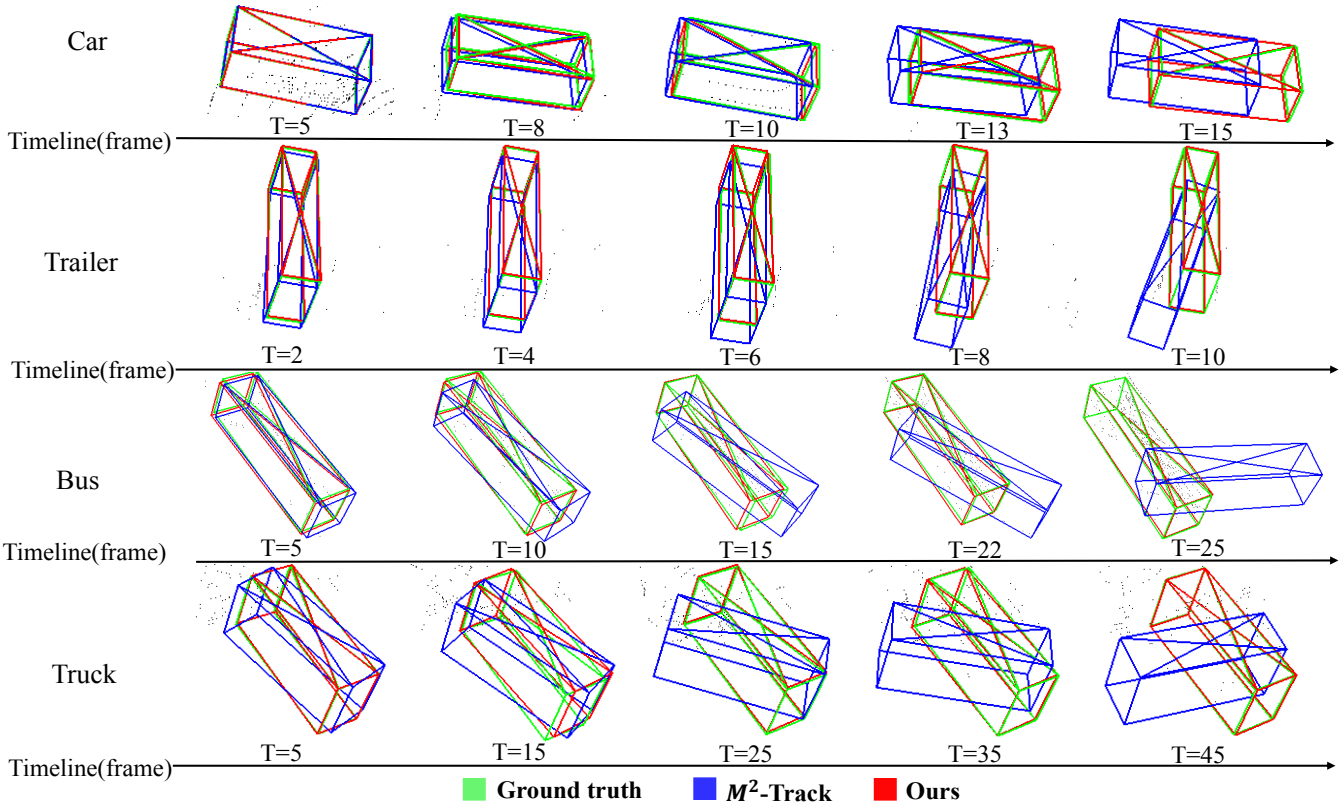

Fig. 6: Result comparison visualization with M^2-Track on the NuScenes dataset.

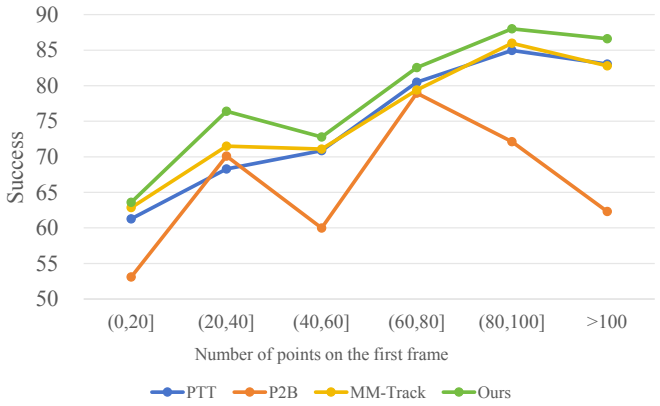

Fig. 7: Comparison with P2B [6], PTT [7], and M^2-Track [14] when the number of points on the car category in the first frame is different.

TABLE IV: ABLATION STUDIES ON DIFFERENT MODEL COMPONENTS OF ON KITTI [15] DATASET. WHERE F STANDS FOR CONVERGED SEGMENTATION NETWORK AND G STANDS FOR GLOBAL EMBEDDED MODULE

F	G	Car	Pedestrian	Van	Cyclist	All
		65.5/80.8	61.5/88.2	53.8/70.7	73.2/93.5	62.9/83.4
✓		66.9/81.3	**61.7**/88.0	54.7/74.8	**77.9**/94.1	63.8/83.9
	✓	66.6/82.0	59.3/88.3	59.7/**76.4**	77.5/94.3	63.1/84.5
✓	✓	**67.6/83.1**	61.6/**88.5**	**60.0**/76.2	**77.9/94.4**	**64.6/85.1**

C. Ablation Studies

1) Model Components: We designed experiments to verify the effectiveness of this module. Using the M^2-Track as a benchmark, we gradually equipped it with our design. The results are shown in table IV. We first replace PointNet++ with a fusion segmentation feature extraction network. The average success and precision are increased by 0.9/0.5, which indicates that it has strong feature representation ability. Then, after configuring GEM, its performance improved by 0.2/0.9. This shows that accurate segmentation and the use of spatial attention can effectively improve tracking performance.

2) CSE module parameter analysis: The precision of segmentation results is very important for the subsequent target positioning of the network. Therefore, under the existing CSE framework, we study the influence of different numbers of residual blocks on tracking performance. The results are shown in table V and we find that we get the best performance when the residual block in the CSE module is set to 3.

3) The choice of k on GEM: The value of k is a hyper-parameter of GEM. As presented in Sec III-C, we adopt the KNN to obtain position embedding. The experimental results for the Car category are indicated in Figure 8, and the best performance is achieved when $k = 16$.

TABLE V: THE INFLUENCE OF RESIDUAL BLOCK NUMBER ON PERFORMANCE.

Depth	Suc/Pre	Depth	Suc/Pre
0	62.6/82.1	1	63.0/82.9
2	63.9/83.5	3	**64.6/85.1**
4	63.3/83.0	5	62.8/82.7

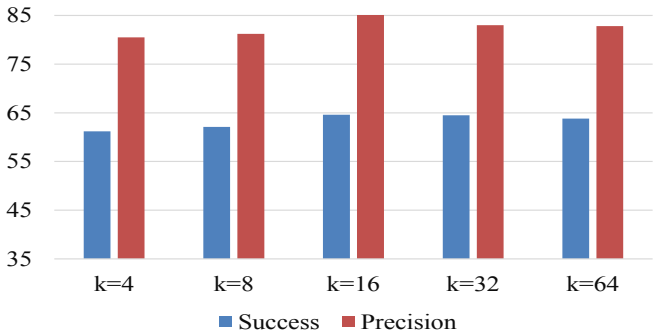

Fig. 8: Performance for different k values in the CSE.

V. CONCLUSION

In conclusion, we introduced an efficient 3D single object tracker (EST) addressing limitations in existing methods, such as M^2-Track, which overlooks segmentation errors in sparse point cloud scenarios. Our EST comprises a fusion segmentation module to compensate for feature loss due to downsampling, enhancing recognition in the foreground area and effectively segmenting point cloud features. Additionally, a global embedded module leverages residual networks and background information to refine target optimization. Extensive experiments on KITTI and NuScenes datasets demonstrate the competitive performance of our approach.

REFERENCES

[1] T. Yin, X. Zhou, and P. Krahenbuhl, "Center-based 3d object detection and tracking," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 11 784–11 793.

[2] M. Jiang, R. Li, Q. Liu, Y. Shi, and E. Tlelo-Cuautle, "High speed long-term visual object tracking algorithm for real robot systems," *Neurocomputing*, vol. 434, pp. 268–284, 2021.

[3] B. Fan, K. Wang, H. Zhang, and J. Tian, "Accurate 3d single object tracker with local-to-global feature refinement," *IEEE Robotics and Automation Letters*, 2022.

[4] Z. Fang, S. Zhou, Y. Cui, and S. Scherer, "3d-siamrpn: An end-to-end learning method for real-time 3d single object tracking using raw point cloud," *IEEE Sensors Journal*, vol. 21, no. 4, pp. 4995–5011, 2020.

[5] S. Giancola, J. Zarzar, and B. Ghanem, "Leveraging shape completion for 3d siamese tracking," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1359–1368.

[6] H. Qi, C. Feng, Z. Cao, F. Zhao, and Y. 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*, 2020, pp. 6329–6338.

[7] J. Shan, S. Zhou, Z. Fang, and Y. 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)*. IEEE, 2021, pp. 1310–1316.

[8] J. Zarzar, S. Giancola, and B. Ghanem, "Efficient bird eye view proposals for 3d siamese tracking," *arXiv preprint arXiv:1903.10168*, 2019.

[9] C. Zheng, X. Yan, J. Gao, W. Zhao, W. Zhang, Z. Li, and S. Cui, "Box-aware feature enhancement for single object tracking on point clouds," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 13 199–13 208.

[10] Y. Cui, J. Shan, Z. Gu, Z. Li, and Z. Fang, "Exploiting more information in sparse point cloud for 3d single object tracking," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 11 926–11 933, 2022.

[11] Q. Wu, C. Sun, and J. Wang, "Multi-level structure-enhanced network for 3d single object tracking in sparse point clouds," *IEEE Robotics and Automation Letters*, vol. 8, no. 1, pp. 9–16, 2022.

[12] S. Feng, P. Liang, J. Gao, and E. Cheng, "Multi-correlation siamese transformer network with dense connection for 3d single object tracking," *IEEE Robotics and Automation Letters*, 2023.

[13] B. Fan, W. Zhou, Y. Yang, and J. Tian, "Integrating scaling strategy and central guided voting for 3d point cloud object tracking," *IEEE Robotics and Automation Letters*, 2024.

[14] C. Zheng, X. Yan, H. Zhang, B. Wang, S. Cheng, S. Cui, and Z. Li, "Beyond 3d siamese tracking: A motion-centric paradigm for 3d single object tracking in point clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 8111–8120.

[15] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE conference on computer vision and pattern recognition*. IEEE, 2012, pp. 3354–3361.

[16] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 621–11 631.

[17] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," *Advances in neural information processing systems*, vol. 30, 2017.

[18] C. R. Qi, O. Litany, K. He, and L. J. Guibas, "Deep hough voting for 3d object detection in point clouds," in *proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 9277–9286.

[19] L. Hui, L. Wang, M. Cheng, J. Xie, and J. Yang, "3d siamese voxel-to-bev tracker for sparse point clouds," *Advances in Neural Information Processing Systems*, vol. 34, pp. 28 714–28 727, 2021.

[20] C. Zhou, Z. Luo, Y. Luo, T. Liu, L. Pan, Z. Cai, H. Zhao, and S. Lu, "Pttr: Relational 3d point cloud object tracking with transformer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 8531–8540.

[21] Z. Guo, Y. Mao, W. Zhou, M. Wang, and H. Li, "Cmt: Context-matching-guided transformer for 3d tracking in point clouds," in *European Conference on Computer Vision*. Springer, 2022, pp. 95–111.

[22] M. Wang, T. Ma, X. Zuo, J. Lv, and Y. Liu, "Correlation pyramid network for 3d single object tracking," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 3215–3224.

[23] J. Han, H. Zheng, D. Z. Chen, and C. Wang, "Stnet: An end-to-end generative framework for synthesizing spatiotemporal super-resolution volumes," *IEEE Transactions on Visualization and Computer Graphics*, vol. 28, no. 1, pp. 270–280, 2021.

[24] T.-X. Xu, Y.-C. Guo, Y.-K. Lai, and S.-H. Zhang, "Cxtrack: Improving 3d point cloud tracking with contextual information," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 1084–1093.

[25] Y. Cui, Z. Li, and Z. Fang, "Sttracker: Spatio-temporal tracker for 3d single object tracking," *IEEE Robotics and Automation Letters*, 2023.

[26] C. R. Qi, H. Su, K. Mo, and L. 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*, 2017, pp. 652–660.

[27] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "Pointcnn: Convolution on x-transformed points," *Advances in neural information processing systems*, vol. 31, 2018.

[28] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. J. Guibas, "Kpconv: Flexible and deformable convolution for point clouds," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 6411–6420.

[29] M. Jiang, Y. Wu, T. Zhao, Z. Zhao, and C. Lu, "Pointsift: A sift-like network module for 3d point cloud semantic segmentation," *arXiv preprint arXiv:1807.00652*, 2018.

[30] X. Yan, C. Zheng, Z. Li, S. Wang, and S. Cui, "Pointasnl: Robust point clouds processing using nonlocal neural networks with adaptive sampling," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 5589–5598.

[31] S. Zheng, J. Pan, C. Lu, and G. Gupta, "Pointnorm: Dual normalization is all you need for point cloud analysis," in *2023 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2023, pp. 1–8.

[32] H. Ran, J. Liu, and C. Wang, "Surface representation for point clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 18 942–18 952.

[33] G. Qian, Y. Li, H. Peng, J. Mai, H. A. A. K. Hammoud, M. Elhoseiny, and B. Ghanem, "Pointnext: Revisiting pointnet++ with improved training and scaling strategies," *arXiv preprint arXiv:2206.04670*, 2022.

[34] Z. Wang, Q. Xie, Y.-K. Lai, J. Wu, K. Long, and J. Wang, "Mlvsnet: Multi-level voting siamese network for 3d visual tracking," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 3101–3110.