

Implicit and Efficient Point Cloud Completion for 3D Single Object Tracking

Pan Wang¹, Liangliang Ren², Shengkai Wu², Jinrong Yang^{1✉}, En Yu¹, Hangcheng Yu¹, Xiaoping Li¹

Abstract—The point cloud based 3D single object tracking (3DSOT) has drawn increasing attention. Lots of breakthroughs have been made, but we also reveal two severe issues. By an extensive analysis, we find the prediction manner of current approaches is non-robust, i.e., exposing a misalignment gap between prediction score and actually localization accuracy. Another issue is the sparse point returns will damage the feature matching procedure of the SOT task. Based on these insights, we introduce two novel modules, i.e., Adaptive Refine Prediction (ARP) and Target Knowledge Transfer (TKT), to tackle them, respectively. To this end, we first design a strong pipeline to extract discriminative features and conduct the matching procedure with the attention mechanism. Then, ARP module is proposed to tackle the misalignment issue by aggregating all predicted candidates with valuable clues. Finally, TKT module is designed to effectively overcome incomplete point cloud due to sparse and occlusion issues. We call our overall framework PCET. By conducting extensive experiments on the KITTI and Waymo Open Dataset, our model achieves state-of-the-art performance while maintaining a lower computational consumption.

I. INTRODUCTION

The point cloud based 3D single object tracking (3DSOT) is an important part of 3D perception scenes, which could be applied in many applications such as 3D environment perception, motion prediction, trajectory prediction in autonomous driving, and intelligent robotics. Although the 3DSOT task has made promising progress, current advanced works [1], [2], [3], [4], [5] still encounter performance bottlenecks since they are bounded by the sparse and occlusion point returns.

The 3DSOT methods can be split into two streams, i.e., matching-based and motion-centric. Inspired by 2D object tracking methods [6], [7], [8], [9], the stream of matching-based methods transfer template and search proposal features to the same embedding space, and then predict the target states according to the feature similarity. The pioneer method SC3D [1] matches the proposal and target by measuring the feature’s cosine similarity, but it fails to train the model in an end-to-end manner and even suffers from computations cost bottleneck. To make up for the computations overhead, P2B [2] first executes permutation-invariant feature augmentation to embed target clues from the template into search area seeds and then employs Hough voting mechanism [10] to directly predicts the target position in an end-to-end framework. To further forecast accurate position, BAT [3]

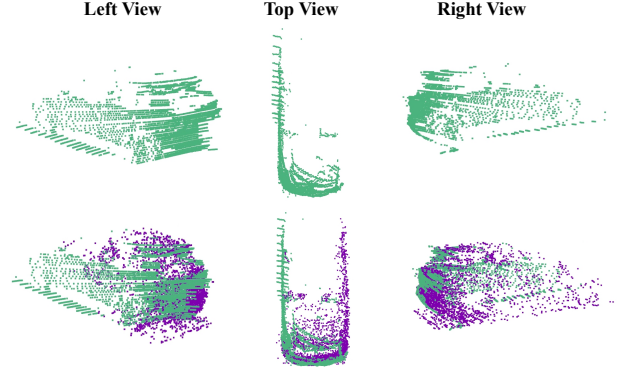


Fig. 1. A visualization of the target point cloud under three different observations. **Top:** Original target only occupies the left partly due to occlusion, which leads to an inaccurate position prediction. **Bottom:** By enhancing with Point Cloud Completion, the right part could be occupied.

enriches the matching feature with a more informative and robust representation by the virtue of point-to-box relation, which is capable of counteracting severe sparse and incomplete shapes effectively. With the help of power transformer operations, PTT [4] and PTTR [5] unleash the power of attention mechanism to capture long-range dependencies for enhancing the valuable feature of template and search areas.

Different from the above straightforward matching-based methods, another motion-centric stream method like MM-Track [11] focuses on motion clues and employs motion-assisted shape completion to refine the target box. By virtue of shape completion in multi frames, it achieves impressive performance. However, the processing of shape completion in a raw point cloud will bring about extra memory consumption and computational burden.

Based on the above discussions, we find potential weaknesses: *The shape completion matters for a template but current works fail to carry out it efficiently and effectively.* As shown in Fig. 1, using the incomplete and occlusive template point cloud to conduct tracking tasks is suboptimal since it is short of valuable tracking clues. After constructing a completed point cloud template, it is reasonable and robust to search for an identity object. The experimental results in Sec. IV will agree with our analysis. In this paper, we propose an efficient and effective Target Knowledge Transfer (TKT) module to conduct implicit shape completion in a compact latent space instead of an explicit counterpart in the raw point cloud. To this end, we first employ an attention mechanism to aggregate valuable information to template features for better absorbing rich knowledge. Afterward, a knowledge transfer module is introduced to transfer valuable

✉ Corresponding author

¹Pan Wang, Jinrong Yang, En Yu, Hangcheng Yu, Xiaoping Li are with Huazhong University of Science and Technology, Wuhan, 430074, China. {panwang725, yangjinrong, yuen, hcy, lixiaoping}@hust.edu.cn

²Liangliang Ren, Shengkai Wu are with the CVTE Research, Guangzhou, 510530, China. {renliangliang, wushengkai}@cvte.com

knowledge from the informative point cloud to template features. Without directly processing the raw point cloud, our framework brings about negligible computation overhead in the inference stage.

Besides, we reveal that the existing works all roughly select the top-1 score of the prediction object as the final result, which leads to serious performance degradation. Because the prediction is not robust, the best prediction box may not match the best score, leading to an imbalanced correlation. We elaborate on the details of the phenomenon in Sec. III. To alleviate the dilemma, we introduce a robust Adaptive Refine Prediction (ARP) method, which considers all prediction candidates' valuable information and employs an adaptive mechanism to predict the final result.

We conduct comprehensive experiments on the KITTI dataset [12] and Waymo Open Dataset [13], showing significant improvements in the Success metric while still maintaining a 32.8 FPS inference speed. In summary, the contributions of this work are as three-fold as follows:

- We find the shape completion matters for a template but inevitably falls into inference speed bottleneck. Thus we introduce the attention-based TKT module to implicitly and efficiently complete the template feature.
- We analyze the imbalance problem between prediction score and localization accuracy and propose an ARP method for mitigating the negative effects and improving its robustness.
- With novel ARP and TKT methods, our PCET achieves the best tracking results in the Success metric and decreases the 84% increase of forwarding time consumption compared to the naive merged points method.

II. RELATED WORK

A. 3D Point Cloud Single Object Tracking

3D SOT task aims to search the identical object with accuracy in 3D location, size, and rotation. One most common scheme is a matching-based method, which conducts the matching procedure between template and search features by measuring feature similarity. The first 3D point cloud single object tracker was proposed in SC3D [1], which generates the target proposals and matches the proposals to the target by cosine similarity. P2B [2] is the first end-to-end 3DSOT model, which generates the predicted target localization by Hough Voting in VoteNet [10]. BAT [3] encodes more rich information with valuable points within the box, it achieves better performance by embedding the box feature to target and search proposals. PTT [4] and PTTR [5] employ an attention mechanism to structure the matching procedure between target and proposals, which leverages implicit similarity operation for better matching. Although the above matching manners achieve remarkable performance, they still fail to tackle the phenomenon of sparse point returns. Therefore, the stream of motion-centric methods try to utilize point clouds in multi frames. By merging the point clouds within identical target regions, MM-Track [11] alleviates the sparse issue and structures a strong template

feature for matching. However, it exposes an inference speed bottleneck since it needs to cost a heavy overhead to extract robust template features again. In this paper, we introduce an efficient implicit point cloud completion method, TKT, which only brings about slight overhead.

B. Attention Mechanism

Attention mechanism [14] is extensively applied in NLP [15], vision [16], [17], and point cloud [18] tasks. Attention module is adept at capturing long-range feature information by measuring similarity. Due to the discreteness of the point cloud, the attention mechanism is very suitable for the 3D modal. Different from the origin attention module, PCT [19] found an offset operation better in enriching features. Point Transformer [20] designed a Point Transformer layer that could keep the permutation invariant of a point cloud. In this paper, we leverage the self-attention module to structure discriminative features and carry out implicit similarity measurement for better result prediction.

C. Point Cloud Completion

Due to occlusion, and sensor quality, the point cloud is often sparse. Therefore, it often lacks intact information, which would be disastrous for several downstream tasks. A natural solution is to conduct point cloud completion to make up for more complete information [21], [22], [23], [24], [25], [26]. AtlasNet [23] utilizes the powerful 3D convolution to process raw point cloud. However, these methods lose important detailed information and suffer from heavy computational consumption. PointNet++ [27] and its variants could capture hierarchical detailed features from a point cloud, which inspires some researchers to apply it in downstream tasks. PCN [28] applies the encoder-decoder framework to process raw data and then maps the 2D points onto a 3D surface gradually by using the FoldingNet [29]. Delving into the 3D SOT task, MM-Track [11] employs the technology to structure a strong template feature for matching. In this paper, we only use the point cloud combinations at the training stage. Then we design a novel TKT module to effectively transfer the valuable information from the feature of the merged point cloud to template feature, which implicitly and efficiently carries out the point cloud completion procedure and only produces slight overhead at the inference stage.

III. METHODOLOGY

We propose an attention-based 3D single object tracker enhanced by implicit point cloud completion dubbed PCET, which includes four parts: 1) Feature extraction, 2) Feature Augmentation and Matching, 3) Adaptive Refine Prediction, and 4) Target Knowledge Transfer. The overall architecture is shown in Fig. 2.

A. Feature Extraction

Following most previous methods [4], [5], we utilize the common PointNet++ [27] to extract feature. Given a single point cloud frame, we sample 1024, and 512 key points

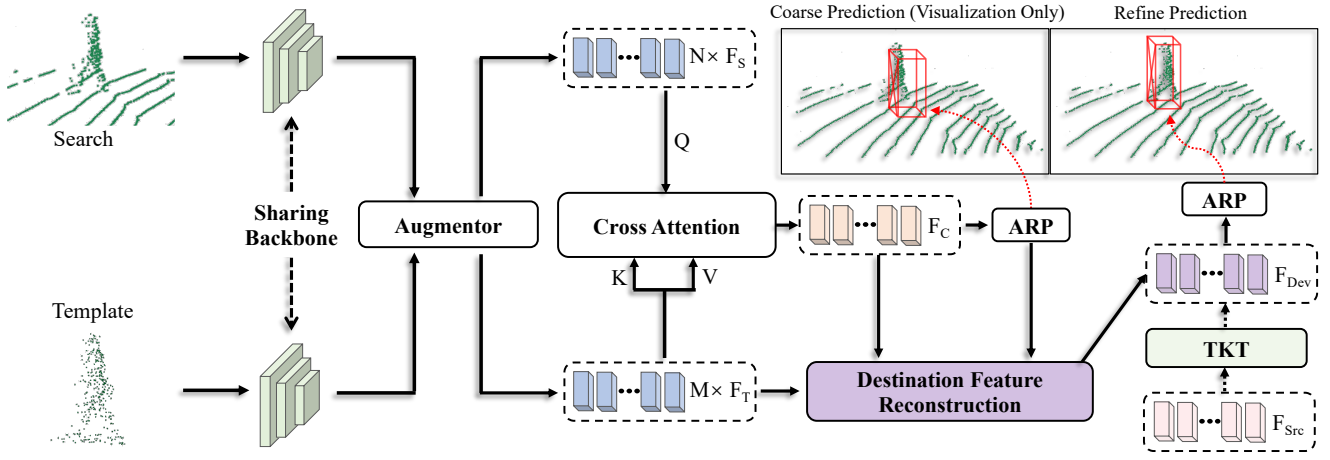


Fig. 2. **The architecture of our proposed PCET.** The network consists of four modules: feature extraction, attention-based feature augmentation and matching, Adaptive Refine Prediction module (ARP), and target knowledge transfer module (TKT). We use the common PointNet++ as sharing backbone for feature extraction. The Augmentor indicates a self-attention module, which enhances the template F_T features and search features F_S , respectively. The cross-Attention module generates the correlation features F_C , which implicitly structures the matching relationship between F_T and F_S . The ARP module is adopted to robustly forecast a coarse prediction or refined prediction. TKT module is used to transfer valuable information from the merged point features (source features F_{Src}) to the target template features (destination features F_{Dev}). Especially, the Destination Feature Reconstruction module is designed to generate F_{Dev} , which will encode more information.

for search area and template, respectively. In particular, we leverage the re-sampling strategy to fill the scale of point as facing sparse points returns (*i.e.*, the number of points is less than the requirement). To generate template and search seeds, we employ the farthest point sampling (FPS) strategy [27] to select M and N point features and then conduct set abstraction (SA) layer [27] to fetch final features. The template and search areas are encoded as compact feature sets $F_T \in \mathbb{R}^{M \times C}$ and $F_S \in \mathbb{R}^{N \times C}$, where C is the dimension of each feature.

B. Feature Augmentation and Matching

Searching the optimal corresponding template and search features is the main target for SOT task. The cruxes of it are *generating the more discriminative features* and *executing a better matching mechanism*. Recent works often adopt cosine similarity [1], [2], [3] to establish the matching relationship for template and search features. Inspire by [30], [14], [5], we instead leverage a versatile attention mechanism to implement the above two keys. To this end, we first introduce an augmentor to enhance both template and search features themselves, which employs a self-attention module to enrich each feature representation and focuses more on the discriminative factors. This is significantly important for subsequent matching. Afterward, we further introduce a cross-attention based augmentation for enhancing template features with target-specific clues, which enables it to implicitly build alignment for matching and to aggregate effective information for the subsequent prediction.

We adopt the attention mechanism (*i.e.*, self-attention and cross-attention) as in [14] for mentioned two keys. Specifically, given the features of template or search as $F = \{f_1, \dots, f_n\}, f_n \in \mathbb{R}^{1 \times C}$, linear projection layers are used to generate the vectors query Q , key K and value V . Then the cosine distances between Q and K are calculated following

by normalization with Softmax operation, which forms the weight map. Different from common attention modules, our cross-attention module utilizes the offset operation to generate the Q followed by [5]. Finally, Q needs to be transformed by a simple linear layer and ReLU operation. The attention module could be formulated as:

$$W = \bar{Q} \cdot \bar{K}^T, \bar{Q} = \frac{W_q Q}{\|W_q Q\|_2}, \bar{K} = \frac{W_k K}{\|W_k K\|_2}, \quad (1)$$

where the W indicates the output attention weights, W_q, W_k, W_v are the projection layer of “Query”, “Key”, “Value”, respectively.

$$\text{Attn}(Q, K, V) = \phi(Q - \text{softmax}(W) \cdot (W_v V)), \quad (2)$$

where the Q, K , and V represent the previous “Query”, “Key”, and “Value” respectively, W indicates the attention weight map, W_v is the projection layer of “Value”, ϕ means the linear layer and ReLU operation.

The feature augmentation and matching parts are shown in Fig. 2, we first employ self-attention for target and search features enhancement, it helps generate more discriminative features for matching. Then we conduct an implicit matching procedure by applying cross attention. Especially, we produce the “Query” vector from the search features F_S and the “Key”, and the “Value” vector from the template features F_T . By applying a 3 MLP layers, we use the ARP module (it will be introduced next section) to predict the coarse result.

C. Adaptive Refine Prediction (ARP)

Contemporary methods most regard the prediction with the maximum predicted score as the best result. However, we find that it exposes a misalignment issue between predicted scores and localization accuracy. As shown in Fig. 3, the prediction result with the maximum score is not the one

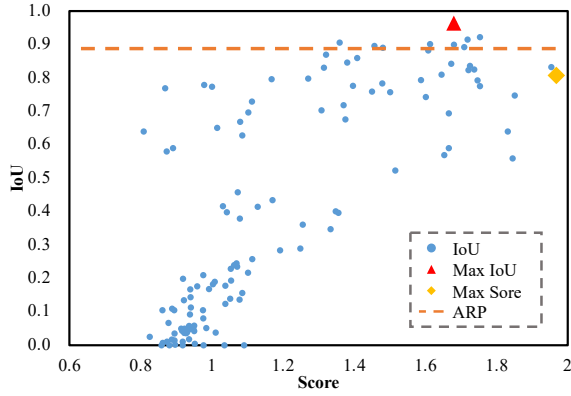


Fig. 3. The imbalance between prediction score and localization accuracy. We visualize some predictions, the horizontal axis represents the predicted scores, vertical axis means the IoU between predicted boxes and ground truth. Clearly, the max predicted score can't correspond to the max IoU result. The orange vertical line represents the result after conducting ARP.

with the highest localization accuracy. Thus, selecting the maximum score candidate will bring about a sub-optimal result, making it fall into a performance bottleneck. We argue that all predictions may carry valuable information for target since they are all supervised to forecast identical objects. From this perspective, we aim to gather all the predicted clues, and generate only one robust result. To this end, we introduce the ARP module to realize the motivation.

Intuitively, the boxes with better prediction qualities should contribute more to the final result. Based on this insight, we propose to predict the *logits distance* between the predicted box center and ground-truth box center, which implicitly measures the prediction quality. We leverage farthest point sample operation [27], [10] to select fixed N candidate point features, and aggregate neighbor point's feature. Then, a set abstraction layer [27] is adopted to further encode the candidate features. Different from other SOT methods to only predict four offsets of center (x, y, z) and rotation θ , and classification score. As shown in Fig. 4, we first get the sum of the logits of prediction score (feature map before softmax) and logits distance. Then we apply a small MLP layer followed by Softmax operation to project them to the final weights. Finally, the weights are aggregated with four offsets by multiplication operation to calculate the final result. The ARP mechanism will adaptively aggregate all prediction results by virtue of both logits of score and distance. The formula of ARP procedure is:

$$R_{1 \times 4} = \sum Softmax(MLP(score + dis)) \cdot off_{N \times 4} \quad (5)$$

where $off_{N \times 4}$ represents the originally predicted offsets, the $R_{1 \times 4}$ indicates the refined results after weighting, and the dis means prediction distance. During training, it is natural to conduct supervision for logits distance [31], but we find that an unsupervised manner performs better, which actually simplifies the training process.

D. Target Knowledge Transfer (TKT)

Although the above cross attention augmentation (Sec. III-B) can enhance the template features, it still lacks more

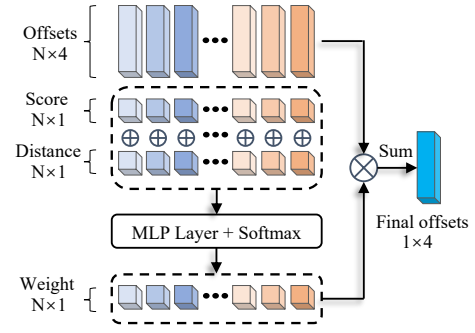


Fig. 4. The architecture of Adaptive Refine Prediction. The ARP module contains a small MLP network, which generates relational weights by combining the predicted score and distance. The final offsets are produced by weighting original offsets.

complete information due to the sparse point returns. As shown in Fig. 1, it reflects that merging point clouds by multiple frames shows more detailed tracking clues (e.g., shape and pose). Therefore, it naturally motivates us to build template features with more intact points for matching procedures during tracking. To do this, a naive way is to merge previous and current (coarse prediction) point cloud like [11]. This template's construction pattern is effective, but it is prone to encounter inference speed bottleneck since it needs to encode the merging template again. To fill this gap, we introduce an efficient module, i.e., TKT, to transfer the valuable knowledge from more intact template features to refine template features during training. It is worth noting that it only brings about negligible latency during inference process.

1) *Source Feature for TKT*: To structure source template with complete information for TKT, we first concatenate the point cloud cropping by ground truth 3D boxes of current frame and refined prediction 3D box of previous frame. The point cloud from previous frame is conducive to compensating for the sparse situation with few temporal biases. Technically, given the previous frame's predicted 3D box $B_p(x_p, y_p, z_p, \theta_p)$ and the current ground truth 3D box $B_c(x_c, y_c, z_c, \theta_c)$, the naive point clouds are cropped to fix points (i.e., 256×3) P_p and P_c , respectively. Specially, we randomly sample 256 points when a total number of points are larger than 256. Inversely, we re-sample the point number to 256 when less than 256. Due to the temporal gap, the previous point cloud will be aligned to the current state by translation and rotation operations. Finally, both of them are concatenated as $P_m(512 \times 3)$. Then, we employ PointNet++ network to encode it and adopt the same operations (i.e., FPS operation and SA layers) of Sec. III-A to generate the source features $F_{src}(N \times C)$ of TKT, where C is the number channels of features.

2) *Destination Feature for TKT*: To design a destination features with better learning potential during the TKT procedure, we argue that it should encode more information. We consider several clues, i.e., matching features F_c after attention (Sec. III-B), coarse prediction, and implicit weight prediction. To this end, we introduce Destination Feature

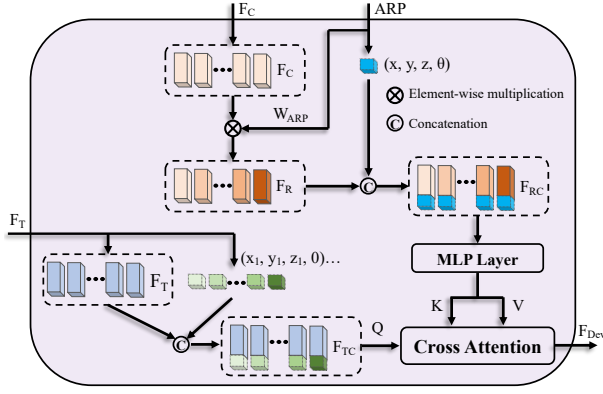


Fig. 5. Destination Feature Reconstruction module.

Reconstruction (DFR) module to do. As shown in Fig. 5, we reuse ARP weights to re-weight features F_C by element-wise multiplication, and then repeat the coarse prediction centers (x, y, z) and rotation θ to concatenate the re-weight features F_R as F_{RC} . To further enhance the destination features, the cross attention operation is conducted to interact with template features F_T . Specially, we also append corresponding center coordinates $(x, y, z, 0)$ to template features F_T as F_{TC} , where 0 is set to keep the same dimensions as F_R . Different from matching procedure, it generates the “Query” vector from the template features F_{TC} and the “Key” and “Value” vector from the relational features F_{RC} . By utilizing the cross-attention module, the destination feature is further enhanced. The formulation is

$$F_{Dev} = \text{Attn}(F_{TC}, F_{RC}, F_{RC}), \quad (3)$$

where F_{Dev} is the destination feature.

3) *Knowledge Transfer*: To better transfer information from F_{Src} to F_{Dev} , we first adopt the KL-Divergence to measure the gap between source and destination features. Inspired by knowledge distillation learning [32], we aim to minimize the distribution gap between destination and source features. Therefore, the optimal target is to minimize the KL-Divergence between two features:

$$D_{KL}(F_{Dev}(x) || F_{Src}(x)) = \sum_{x \in X} F_{Dev}(x) \log \left(\frac{F_{Dev}(x)}{F_{Src}(x)} \right), \quad (4)$$

where $F_{Dev}(x)$ represents the probability distribution of the refined feature and $F_{Src}(x)$ is the merged point’s feature distribution. TKT module enables reconstructed features to learn more valuable information like intact point clues with slight latency since it needs not to extract the merged point cloud again.

E. Optimization

During the training stage, our PCET is optimized in three stages: training coarse prediction, training source feature of TKT, and refined prediction with TKT procedure. All the predictions contain a regression component Y_{reg} , a classification component Y_{cls} . Y_{reg} consists of the predicted offsets $x, y,$

z and an angle offset θ . Our classification loss L_{cls} is defined by binary cross-entropy, and regression loss L_{reg} is computed by mean square error. Especially, for the TKT module, we use the KL-divergence D_{KL} to measure the difference between the target feature and the predicted feature. In a word, the optimal object is formulated as follow in three stages:

$$L_{coarse} = L_{cls}(Y_{cls}^c, L_{cls}^{gt}) + L_{reg}(Y_{reg}^c, L_{reg}^{gt}), \quad (6)$$

$$L_{source} = L_{cls}(Y_{cls}^s, L_{cls}^{gt}) + \lambda_1 L_{reg}(Y_{reg}^s, L_{reg}^{gt}), \quad (7)$$

$$L_{refine} = L_{cls}(Y_{cls}^r, L_{cls}^{gt}) + \lambda_2 L_{reg}(Y_{reg}^r, L_{reg}^{gt}) + \lambda_3 L_{KL}, \quad (8)$$

where $\lambda_1, \lambda_2,$ and λ_3 indicate the weight parameters. L_{KL} indicates the KL-Divergence loss in Eq. 4, F_{Dev}, F_{Src} represent destination feature and source feature, respectively. $L_{coarse}, L_{source},$ and L_{refine} correspond to the above three stages training procedures, respectively.

IV. EXPERIMENTS

A. Experiments Setups

1) *Dataset*: To show the superiority of our method, we evaluate the performance of our model on the KITTI [12] tracking dataset and Waymo Open Dataset (WOD) [13]. Following previous works [2], [3], [4], [5], we split KITTI tracking scenes into three type of tracklets, *i.e.*, scenes 0-16 for training, scenes 17-18 for validation, and scenes 19-20 for testing. For WOD, we follow [5] to generate a class balanced version. By verifying the performance of WOD, it shows the generalization of our method.

2) *Evaluation Metrics*: We use the One Pass Evaluation (OPE) [1] to evaluate the performance of models. It employs the Intersection over Union (IOU) to measure the *overlap* and adopts the distance between the centers of ground truth box and predicted box to measure the *error*. We report *Success* to measure the Area Under the Curve (AUC) with *overlap* threshold varying from 0 to 1, and *Precision* to measure the AUC with *error* threshold from 0 to 2 meters.

3) *Model Details*: For consistent comparisons, our model use the same setting as PTTR [5], 3 set-abstraction layers in PointNet++ [27]. We employ the common farthest sampling method for sampling key points, which is efficient enough for feature extraction. Especially, we share weights in the PointNet++ part for feature extraction, which could transform the original point cloud to the same feature space. In the coarse prediction and refinement stage, 3-layer MLP is used for classification and regression. In training and testing, our template is sampled from the previous prediction result, which could ensure the stability of the template.

4) *Training and Testing*: For balanced training loss, the $\lambda_1, \lambda_2, \lambda_3$ is set as 0.1, 0.05, and 1.0, respectively. We train the model in a three-stage way. Firstly, we train the Coarse Prediction procedure for 300 epochs with a batch size of 300 on 4 NVIDIA RTX3090 GPUs. We use the Adam optimizer

TABLE I

COMPARISON WITH OTHER METHODS ON THE KITTI DATASET. MEAN PRESENTS THE AVERAGE RESULT OF ALL CATEGORIES. BOLD AND UNDERLINE DENOTE THE BEST AND THE SECOND-BEST PERFORMANCE.

	Category	Car	Ped	Van	Cyclist	Mean
Success	SC3D [1]	41.3	18.2	40.4	41.5	35.4
	SC3D-RPN [33]	36.3	17.9	-	43.2	-
	FSiamese [34]	37.1	16.2	-	47.0	-
	P2B [2]	56.2	28.7	40.8	32.1	39.5
	3DSiamRPN [35]	58.2	35.2	45.6	36.1	43.8
	LTTR [36]	65.0	33.2	35.8	66.2	50.1
	BAT [3]	65.4	45.7	52.4	33.7	49.3
	PTT [4]	67.8	44.9	43.6	37.2	48.4
	V2B [37]	70.5	48.3	50.1	40.8	52.4
	PTTR [5]	65.2	50.9	52.5	65.1	58.4
	MM-Track [11]	65.5	61.5	<u>53.8</u>	<u>73.2</u>	<u>63.5</u>
	Ours	<u>68.7</u>	<u>56.9</u>	57.9	75.6	64.8
Precision	SC3D [1]	57.9	37.8	47.0	70.4	53.3
	SC3D-RPN [33]	51.0	47.8	-	81.2	-
	FSiamese [34]	50.6	32.2	-	77.2	-
	P2B [2]	72.8	49.6	48.4	44.7	53.9
	3DSiamRPN [35]	76.2	56.2	52.8	49.0	58.6
	LTTR [36]	77.1	56.8	45.6	89.9	67.4
	BAT [3]	78.9	74.5	<u>67.0</u>	45.4	66.5
	PTT [4]	81.8	72.0	52.5	47.3	63.4
	V2B [37]	<u>81.3</u>	73.5	58.0	49.7	65.6
	PTTR [5]	77.4	81.6	61.8	90.5	77.8
	MM-Track [11]	80.8	88.2	70.7	<u>93.5</u>	83.3
	Ours	80.1	<u>85.1</u>	66.1	93.7	<u>81.3</u>

TABLE II

COMPARISON WITH PREVIOUS METHODS ON THE WAYMO OPEN DATASET. SUCCESS / PRECISION ARE USED FOR EVALUATION, MEAN PRESENTS THE AVERAGE RESULT OF ALL CATEGORIES.

Method	Vehicle	Pedestrian	Cyclist	Mean
SC3D [1]	46.5 / 52.7	26.4 / 37.8	26.5 / 37.6	33.1 / 42.7
P2B [2]	55.7 / 62.2	35.3 / 54.9	30.7 / 44.5	40.6 / 53.9
PTTR [5]	58.7 / 65.2	49.0 / 69.1	43.3 / 60.4	50.3 / 64.9
Ours	61.2 / 67.4	50.8 / 70.0	47.9 / 66.0	53.3 / 67.8

and an initial learning rate of 0.001 which decreases by 2 every 50 epochs. Then, we train the merged point cloud to generate source feature of the TKT with another 100 epochs. Finally, we train the refine prediction with TKT procedure for 100 epochs with a learning rate of 0.0005. Due to Waymo Open Dataset's larger target numbers, we train the first stage in 400 epochs, and the second and last stage in 200 epochs, respectively.

B. Comparison with State-of-the-arts

1) *Results on KITTI*: As shown in Tab. I, we compare several advanced methods. Our method outperforms current schemes' Success metric by a large margin, whilst ranking the second performance in the Precision metric. Compared with attention based PTTR, our method shows better performance (i.e., either Success or Precision) with respect to all categories, revealing the superiority of our proposed modules. As for point cloud completion based MM-Track, our PCET performs better Success accuracy and performs

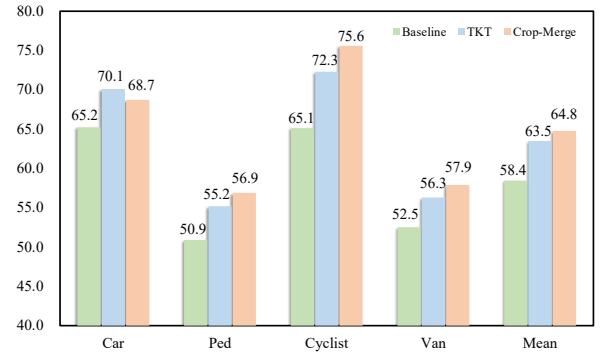


Fig. 6. The Success performance of TKT and Crop-Merge methods on KITTI scene.

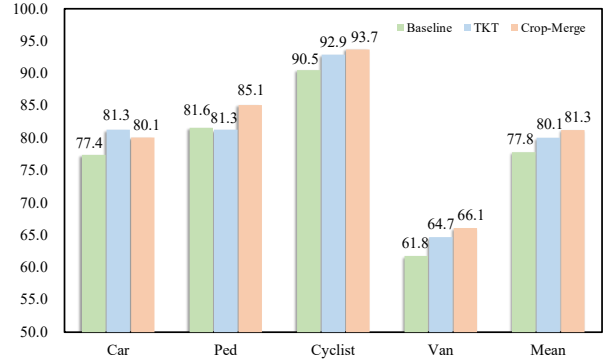


Fig. 7. The Precision performance of TKT and Crop-Merge methods on KITTI scene.

comparable Precision accuracy, while our TKT module is more efficient with lower inference latency.

2) *Results on Waymo Open Dataset*: As shown in Tab. II, our method also outperforms current approaches by a larger margin. For all categories, our scheme all ranks top-1 performance with respect to the Success and Precision metric, verifying the generalization and effectiveness.

C. Ablation Study

In this section, we conduct ablation experiments to verify the effectiveness of TKT and ARP modules. We report the detailed results of all categories on the KITTI dataset and WOD.

1) *Effectiveness of TKT*: As shown in Fig. 6 and Fig. 7, we report the performance of Crop-Merge point completion and our proposed TKT. The Crop-Merge manner shows significant improvements in Success and Precision accuracy, which reveals employing a more intact point cloud to structure template features is the key for the SOT tasks. The conclusion agrees with our analyse in Sec. I. As shown in Tab. III, although the Crop-Merge way shows remarkable performance, it will trigger cumbersome latency by 4.9 ms (see Tab. III). Our proposed TKT is designed to eliminate the dilemma. The results in Fig. 6 and Fig. 7 indicate that the TKT module can also improve performance by conducting knowledge transfer. Especially, the performance of the Car category surpasses the Crop-Merge way. Moreover, TKT only brings about slight latency, i.e., saves 84% of the time,

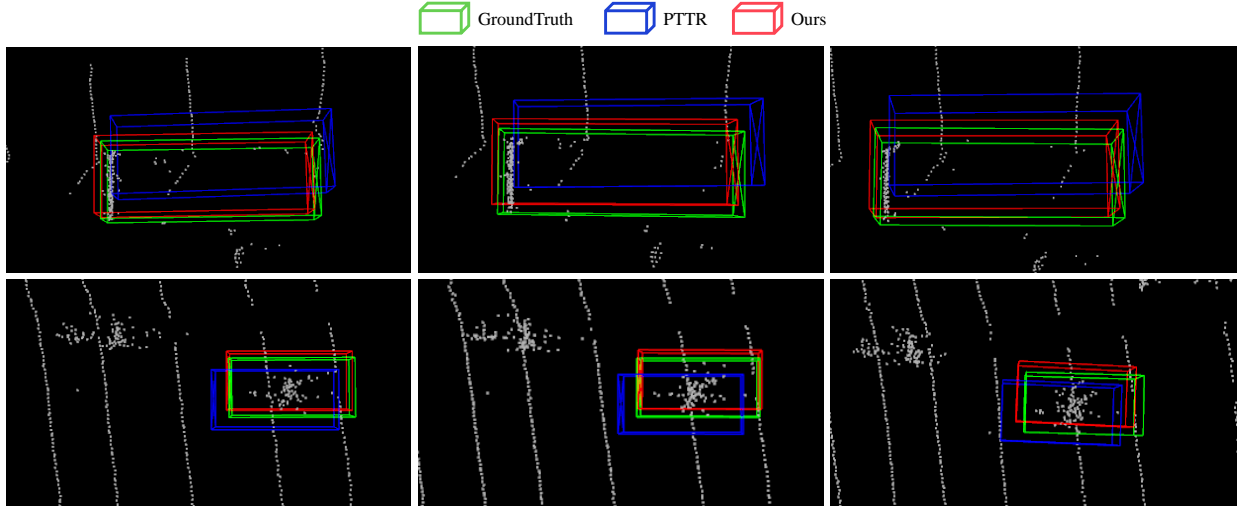


Fig. 8. **The visualization of tracking results on the KITTI dataset.** We compare our proposed PCET with PTTR on the Van (Top) and Cyclist (Bottom) categories. It's obvious that our PCET could achieve better tracking with an incomplete target than PTTR.

TABLE III

INFERENCE LATENCY. WE TEST THEM ON A NVIDIA 1080Ti GPU.

Method	Latency	Speed
Baseline	29.7 ms	33.7 FPS
Crop-Merge	34.6 ms	28.9 FPS
PCET	30.5 ms (84% \uparrow)	32.8 FPS (81% \uparrow)

TABLE IV

EFFECT OF TKT AND ARP MODULES ON WAYMO OPEN DATASET.

Method	Vehicle	Pedestrian	Cyclist	Mean
Ours	61.2 / 67.4	50.8 / 70.0	47.9 / 66.0	53.3 / 67.8
w/o ARP	59.5 / 65.7	49.7 / 69.3	44.6 / 63.4	51.3 / 66.1
Improvement	0.8 / 0.5	0.7 / 0.2	1.3 / 3.0	1.0 / 1.2
w/o TKT	60.4 / 66.4	50.3 / 69.8	46.3 / 64.8	52.3 / 67.0
Improvement	1.7 / 1.2	1.3 / 0.7	3.3 / 4.4	2.0 / 2.1

which is the key for real-time system [38], [39]. As shown in Fig. IV, the TKT module is also effective, which reflect the generalization in larger scale tracking scene.

2) *Effectiveness of ARP*: We report the performance of ARP in Tab. V, which verifies that it can significantly boost the Success and Precision accuracy in all categories by nearly 3.0% and 2.0%, respectively. As for the WOD scene, it also shows nearly 1.0% improvement, which reflects ARP can be applied to a wide range of scenarios. It also reveals that reasonably aggregating all predicted candidates with valuable information can improve the robustness of prediction, which actually alleviates the misalignment between prediction score and location accuracy.

D. Visualization result

To better demonstrate the effectiveness of our method, we report some tracking results on the KITTI dataset. Obviously, our model PCET could achieve a better localization accuracy

TABLE V

ABLATION STUDIES ON ADAPTIVE REFINE PREDICTION. SUCCESS / PRECISION ON THE KITTI DATASET IS USED FOR EVALUATION. BASELINE MEANS EVALUATING THE MAX SCORE PREDICTIONS.

Method	Car	Ped	Van	Cyclist	Mean
Baseline	65.3/77.3	50.2/81.6	52.1/61.9	65.6/90.3	58.3/77.8
ARP	67.8/78.9	53.2/83.1	54.0/64.5	69.6/92.2	61.2/79.7

as shown in Fig. 8. In the sparse and occlusion environment, PTTR [5] fails to carry out accurate localization. In the contrast, our PCET still achieves accurate tracking due to our proposed ARP and TKT modules.

V. CONCLUSIONS

In this paper, we analyze the importance of using a more complete point template to deal with sparse point returns. Furthermore, we find the misalignment between prediction score and localization accuracy since only selecting the top-1 score for final result. Based on these findings, we propose TKT and ARP modules to tackle them. Comprehensive experiments reflect that the TKT module can efficiently enrich template features with valuable information for better matching. Experiments also verify that the ARP module can aggregate all predictions and output more robust results. Compared with the current advanced methods, our PCET achieves state-of-the-art tracking performance, while outperforming on most of categories by a large margin.

ACKNOWLEDGMENT

The authors would like to thank Mr. Junlong Wu, Wenbo Zhou, Qiong Luo, and Yupeng Li for providing the GPUs, reviewing this work, and downloading datasets support. This work is done during Pan Wang's internship at CVTE Research, the authors would also like to thank the CVTE for research support.

REFERENCES

- [1] 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*, pp. 1359–1368, 2019.
- [2] 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*, pp. 6329–6338, 2020.
- [3] 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*, pp. 13199–13208, 2021.
- [4] 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)*, pp. 1310–1316, IEEE, 2021.
- [5] C. Zhou, Z. Luo, Y. Luo, T. Liu, L. Pan, Z. Cai, H. Zhao, and S. Lu, “Pptr: Relational 3d point cloud object tracking with transformer,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8531–8540, 2022.
- [6] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr, “Fully-convolutional siamese networks for object tracking,” in *European conference on computer vision*, pp. 850–865, Springer, 2016.
- [7] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan, “Siamrpn++: Evolution of siamese visual tracking with very deep networks,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4282–4291, 2019.
- [8] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu, “High performance visual tracking with siamese region proposal network,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8971–8980, 2018.
- [9] R. Tao, E. Gavves, and A. W. Smeulders, “Siamese instance search for tracking,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1420–1429, 2016.
- [10] Z. Ding, X. Han, and M. Niethammer, “Votenet: A deep learning label fusion method for multi-atlas segmentation,” in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 202–210, Springer, 2019.
- [11] 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*, pp. 8111–8120, 2022.
- [12] 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*, pp. 3354–3361, IEEE, 2012.
- [13] P. Sun, H. Kretschmar, X. Dotiwalla, A. Choudhury, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. 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*, pp. 2446–2454, 2020.
- [14] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” 2017.
- [15] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018.
- [16] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al., “An image is worth 16x16 words: Transformers for image recognition at scale,” *arXiv preprint arXiv:2010.11929*, 2020.
- [17] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, “End-to-end object detection with transformers,” in *European conference on computer vision*, pp. 213–229, Springer, 2020.
- [18] L. Fan, Z. Pang, T. Zhang, Y.-X. Wang, H. Zhao, F. Wang, N. Wang, and Z. Zhang, “Embracing single stride 3d object detector with sparse transformer,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8458–8468, 2022.
- [19] M.-H. Guo, J.-X. Cai, Z.-N. Liu, T.-J. Mu, R. R. Martin, and S.-M. Hu, “Pct: Point cloud transformer,” *Computational Visual Media*, vol. 7, no. 2, pp. 187–199, 2021.
- [20] H. Zhao, L. Jiang, J. Jia, P. H. Torr, and V. Koltun, “Point transformer,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16259–16268, 2021.
- [21] H. Fan, H. Su, and L. J. Guibas, “A point set generation network for 3d object reconstruction from a single image,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 605–613, 2017.
- [22] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese, “3d-r2n2: A unified approach for single and multi-view 3d object reconstruction,” in *European conference on computer vision*, pp. 628–644, Springer, 2016.
- [23] T. Groueix, M. Fisher, V. G. Kim, B. Russell, and M. Aubry, “Atlasnet: A papier-mâché approach to learning 3d surface generation,” *computer vision and pattern recognition*, 2018.
- [24] Z. Huang, Y. Yu, J. Xu, F. Ni, and X. Le, “Pf-net: Point fractal network for 3d point cloud completion,” *computer vision and pattern recognition*, 2020.
- [25] G. Synnaeve, Q. Xu, J. Kahn, E. Grave, T. Likhomanenko, V. Pratap, A. Sriram, V. Liptchinsky, and R. Collobert, “End-to-end asr: from supervised to semi-supervised learning with modern architectures,” *arXiv: Computation and Language*, 2019.
- [26] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, “3d shapenets: A deep representation for volumetric shapes,” *computer vision and pattern recognition*, 2014.
- [27] 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.
- [28] W. Yuan, T. Khot, D. Held, C. Mertz, and M. Hebert, “Pcn: Point completion network,” in *2018 International Conference on 3D Vision (3DV)*, pp. 728–737, IEEE, 2018.
- [29] Y. Yang, C. Feng, Y. Shen, and D. Tian, “Foldingnet: Point cloud auto-encoder via deep grid deformation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 206–215, 2018.
- [30] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al., “An image is worth 16x16 words: Transformers for image recognition at scale,” *arXiv preprint arXiv:2010.11929*, 2020.
- [31] J. Yang, S. Wu, L. Gou, H. Yu, C. Lin, J. Wang, P. Wang, M. Li, and X. Li, “Scd: A stacked carton dataset for detection and segmentation,” *Sensors*, vol. 22, no. 10, p. 3617, 2022.
- [32] G. Hinton, O. Vinyals, J. Dean, et al., “Distilling the knowledge in a neural network,” *arXiv preprint arXiv:1503.02531*, vol. 2, no. 7, 2015.
- [33] J. Zarzar, S. Giancola, and B. Ghanem, “Efficient bird eye view proposals for 3d siamese tracking,” *arXiv: Computer Vision and Pattern Recognition*, 2019.
- [34] H. Zou, J. Cui, X. Kong, C. Zhang, Y. Liu, F. Wen, and W. Li, “F-siamese tracker: A frustum-based double siamese network for 3d single object tracking,” *intelligent robots and systems*, 2020.
- [35] 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*, 2021.
- [36] Y. Cui, Z. Fang, J. Shan, Z. Gu, and S. Zhou, “3d object tracking with transformer,” *arXiv preprint arXiv:2110.14921*, 2021.
- [37] 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. 28714–28727, 2021.
- [38] J. Yang, S. Liu, Z. Li, X. Li, and J. Sun, “Real-time object detection for streaming perception,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5385–5395, 2022.
- [39] J. Yang, S. Liu, Z. Li, X. Li, and J. Sun, “Streamyolo: Real-time object detection for streaming perception,” *arXiv preprint arXiv:2207.10433*, 2022.