HyGFNet: Hybrid Geometry-Flow Learning Network for 3D Single Object Tracking

Yubo Cui, Zheng Fang*, *Member, IEEE*, Zhiheng Li, Shuo Li, Yu Lin

***Abstract*—3D single object tracking (SOT) which attempts to accurately locate the target object in the current frame, has made significant advancements over past years. However, most previous works built upon the Siamese architecture usually focus on the learning and matching of geometry information, while neglecting the motion information of the target. Consequently, those methods face challenges when distinguishing the target object from similar distractors. To overcome this limitation, we enhance the architecture by incorporating flow estimation, presenting a novel multi-frame hybrid geometry-flow learning network for 3D SOT. The proposed framework exploits both geometry and flow information from historical frames and further integrates the learned information into the current frame to improve the target object localization. Specifically, we propose a geometry matching branch and a flow estimation branch. The geometry matching branch first captures the geometry feature of each frame and then aligns these features through multi-frame spatial-temporal matching, leading to a geometry-aware feature for the target object. Meanwhile, in the flow estimation branch, a multi-frame flow-aware enhancement module is proposed to explicitly capture flow information across frames, leading to a flow-aware feature for the target object. Finally, the geometry-aware and flow-aware features are fused with the original feature to predict the position of the target object. Extensive experiments conducted on the KITTI and nuScenes datasets validate the effectiveness and show the competitive performance of our method. The code will be open soon.**

***Index Terms*—3D single object tracking (SOT), Flow estimation, Siamese network.**

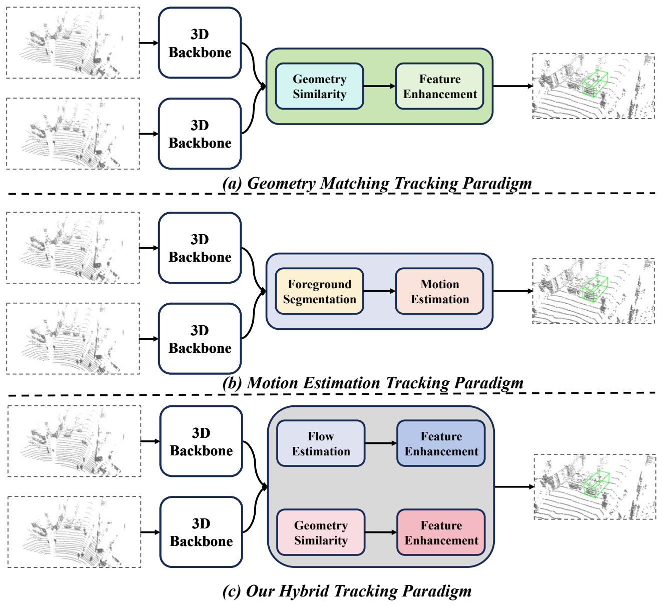

Fig. 1. **Comparison of the current 3D single object tracking frameworks.** (a) Geometry matching paradigm usually adopts similarity-based matching and fusion. (b) Motion estimation paradigm predicts an object-level box transformation. (c) Our hybrid paradigm includes both geometry learning and flow estimation.

I. INTRODUCTION

3D single object tracking, as an important task in 3D computer vision, has attracted increasing attention in recent years. Given the 3D bounding box of the target object in the initial frame, the task needs to estimate the position and state of the target in each subsequent frame. Previous works [1]–[7] have largely inherited the paradigm of visual tracking, employing the Siamese architecture to track target object. As shown in Figure. 1 (a), this paradigm first extracts features for the template point cloud and the current point cloud, then matches and fuses the two learned features for the prediction. However, unlike using ordered and dense images as inputs in visual tracking, 3D SOT needs to handle sparse and unordered point cloud inputs [8]. Therefore, there are some challenges in directly applying the matching paradigm from visual tracking to 3D SOT. For example, because the geometric structures of objects from the same category are usually similar in point cloud representation, it is difficult to locate the target object solely relying on the matching paradigm under the scenes with similar distractors.

To address this challenge, previous methods [5]–[7], [9], [10] introduced more complex geometry learning and matching modules to learn higher-dimensional information to improve the accuracy of geometric information matching. Specially, V2B [6] proposes template feature embedding to encode the search area by learning the similarity of the global shape and local geometric structures, and further learns the shape-aware feature to characterize the geometric structures of the target better. Moreover, SMAT [9] proposes a multi-scale attention-based encoder to capture the global similarity. STNet [10] proposes an iterative coarse-to-fine correlation to augment the similarity feature with attention. PTTR [7] proposes an attention-based feature matching to match search and template features and generates a coarse prediction based on the matched feature. Although these methods alleviated

This work was supported in part by the National Natural Science Foundation of China under Grants 62073066 and U20A20197, in part by the Fundamental Research Funds for the Central Universities under Grant N2226001, and in part by 111 Project under Grant B16009. (Corresponding author: Zheng Fang)

All authors are with the Faculty of Robot Science and Engineering, Northeastern University, Shenyang, China. Yubo Cui and Zheng Fang are also with the National Frontiers Science Center for Industrial Intelligence and Systems Optimization, Shenyang 110819, China, and also with the The Key Laboratory of Data Analytics and Optimization for Smart Industry (Northeastern University), Ministry of Education Shenyang 110819, China.

the problem of losing the target in some complex tracking scenarios, they are still limited by the geometric matching paradigm.

Differently, M^2-Tracker [11] proposes a two-stage motion estimation paradigm to the 3D SOT task. As shown in Figure. 1 (b), in the first stage, it segments the foreground points in the current frame by using the prior from the previous frame, and then estimate a coarse rigid body transformation to a coarse prediction. In the second stage, it merges the foreground points from current and previous frames to refine the coarse prediction to get the final prediction. Compared to the matching paradigm [1]–[7], this motion paradigm inherently avoids the challenge of distinguishing the target from similar distractors, thus achieving superior tracking performance. However, the object-level motion estimation is relatively coarse for small-scale motions. Additionally, the motion estimation paradigm is prone to inaccuracies over larger tracking intervals.

As mentioned above, on one hand, the matching paradigm could locate the target object based on similarity when the extracted features contain enough information, but it is sensitive to interference from similar objects and is unstable in sparse scenes. On the other hand, the motion paradigm can locate the target based on motion information, but imprecise motion estimation in scenarios with small-scale motion or substantial time intervals may also lead to target loss. Consequently, can we combine the advantages of matching and motion paradigms for 3D SOT?

In this paper, to solve this problem, we propose a multi-frame hybrid paradigm to combine the advantages of both geometry and motion paradigms. We propose **Hy**brid **G**eometry-**F**low Learning **Net**work (HyGFNet) to learn geometry information and estimate point-level flow for matching and motion estimation respectively, as illustrated in Figure. 1(c). The proposed HyGFNet takes multi-frame point clouds as input and consists of two branches, a geometry matching branch and a flow estimation branch. In the geometry matching branch, we first learn the geometry structure information of each historical frame by learning a Bird's-Eye View (BEV) bounding box mask, then we use the learned geometry information to enhance current representation by a proposed attention-based spatial-temporal decoder, resulting in a geometry-aware current feature. In the flow estimation branch, different from M^2-Tracker which only predicts object-level motion, we aim to predict more accurate point-level flow between the current frame and each historical frame. Consequently, our point-level flow can enhance point representation and emphasize the motion of target more effectively than M^2-Tracker. After that, similar to the geometry matching branch, each historical frame feature augmented with the flow feature, is further utilized to fuse with the current feature, yielding the flow-aware feature. Finally, the geometry-aware feature, the flow-aware feature, and the original current frame feature are fused to predict the current target object.

In summary, our contributions are as follows:

- We propose HyGFNet, a new hybrid paradigm for 3D SOT, which leverages both the advantages of geometry and flow information to track the target.
- We propose a geometry-aware masking module to learn geometry information at each historical frame, and introduce a flow-aware enhancement module to learn the flow information between the current frame and each historical frame. Finally, a spatial-temporal decoder is proposed to aggregate all geometry and flow information from multi-frame features respectively.
- The proposed HyGFNet achieves promising performances in KITTI and nuScenes datasets. Ablation studies also verify the effectiveness of each proposed module. We will release the source code of the method to the community.

The rest of this paper is organized as follows. In Section. II, we discuss the related work. Section. III describes the detailed structure of our HyGFNet. In Section. IV, we compare our methods with previous methods in KITTI and nuScenes datasets, and conduct ablation studies to explore the effectiveness of each proposed module. Finally, we conclude in Section. V.

II. Related Work

A. 2D Single Object Tracking

Early approaches in visual SOT usually relied on correlation filters [12], [13] to find the target object. However, these methods face limitations as they heavily depend on template matching and usually use hand-craft features with insufficient information, leading to limited performance for fast-moving object tracking. With the advent of convolutional neural network (CNN) and deep learning, many works [14]–[21] adopt the Siamese CNN network architecture to match the template and search features, largely improving the overall tracking performance. Recently, the Transformer architecture [22] has gained widespread application in visual SOT due to its global receptive field and enhanced similarity modeling capabilities. Therefore, many works tried to use the attention mechanism in visual SOT. TransTrack [23] replaces the correlation operation with the attention mechanism to model long-distance feature association. TransMeet [24] introduces transformer in the video clip to learn spatial-temporal information and achieves better performance. Stark [25] also proposes a transformer-based tracking framework to capture the long-range dependency in both spatial and temporal spaces, eliminating the need for hyperparameter-sensitive post-processing. Additionally, MixFormer [26] proposes iterative mixed attention, unifying feature extraction and target integration and leading to a neat and compact tracking pipeline. Different from the above models that directly predict the target, ToMP [27] proposes a transformer-based target model prediction network to estimate the weights of the target model. Moreover, inspired by MAE [28], DropMAE [29] proposes an adaptive spatial-attention dropout method to facilitate temporal correspondence learning in self-supervised pre-training on videos. Despite the success of the matching paradigm in visual SOT, the significant difference between the structures of the point cloud and the image poses a considerable challenge in leveraging the matching paradigm for 3D SOT.

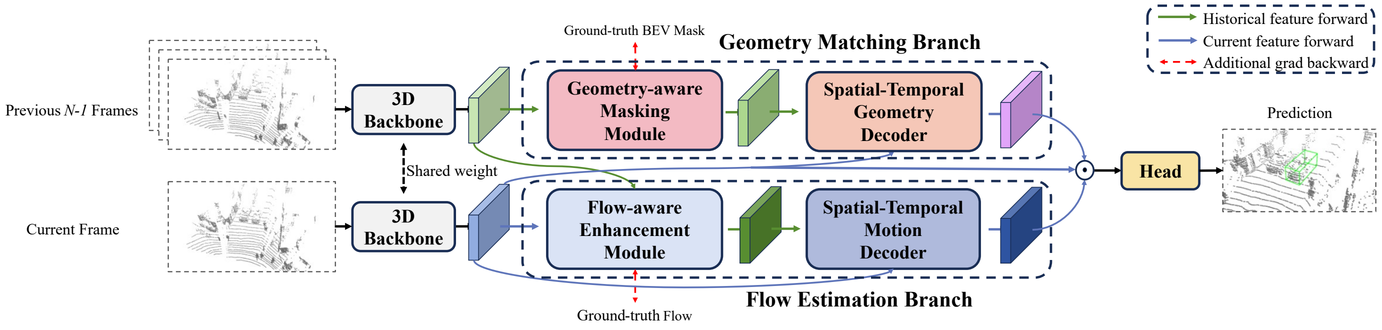

Fig. 2. **Overview of our proposed HyGFNet.** The network consists of two branches named Geometry Matching Branch (GMB) and Flow Estimation Branch (FEB) respectively. In the GMB, the proposed geometry-aware masking module first enhances the geometry information at each historical frame feature. Then, the proposed spatial-temporal geometry decoder integrates the multi-frame geometry information into the current feature. In the FEB, the flow-aware enhancement module obtains the flow-aware feature for each historical frame based on flow estimation. The enhanced feature is further used to fuse with the current frame by the proposed spatial-temporal motion decoder. Finally, we fuse the geometry-aware feature, flow-aware feature and the original feature and use a center-manner head to predict the box. $\odot$ means feature concatenation.

B. 3D Single Object Tracking

Similar to 2D SOT, most previous 3D SOT methods [2], [3], [5], [6] also depend on geometry matching between the template and search point cloud to track the target object. P2B [2] computes the point-wise cosine similarity map to augment the feature and uses VoteNet [30] to regress the 3D bounding box of the target. Based on that, BAT [3] enhances the point features with the box prior information to better capture the point-wise similarity. V2B [6] converts the unorder point features into dense BEV features and predicts the box in a dense manner. Inspired by the attention mechanism, PTT [4], LTTR [5], SMAT [9], PTTR [7], and STNet [10] also leverage the Transformer to capture the global geometry information within or between the point clouds. Benefiting from the outstanding similarity modeling and global perception of attention mechanisms, these methods have achieved impressive tracking performances. However, limited by the sparseness of point clouds, the geometry matching paradigm still encounters challenges in sparse and complex scenes. To address this problem, M^2-Tracker [11] proposes a motion-based framework to estimate the rigid box transformation between the previous frame and the current frame, leading to a more robust and accurate performance. The method first segments the foreground points in the current frame and then estimates a coarse bounding box. A refinement module is further used to refine the box to get a more accurate prediction. Nevertheless, in scenarios with large time intervals, this approach may also fail to track the target object accurately due to inaccurate motion estimation. Therefore, we can integrate the above two paradigms to leverage their advantages to address the challenges in the tracking process. However, how to integrate the two different paradigms still remains under-explored.

C. Scene Flow Estimation on 3D Point Clouds

3D scene flow was first introduced by Vedula *et al.* [31] and aims to directly estimate dense rigid motion fields in 3D LiDAR scans. Previous works usually estimate scene flow by using optical flow and depth information from RGB-D data [32]–[34]. Nowadays, with the advent of deep learning and LiDAR sensors, many works estimate the scene flow from raw point clouds with deep neural networks. FlowNet3D [35] first proposes an end-to-end framework to estimate the scene flow based on PointNet++ [36]. They introduce a flow embedding layer to aggregate geometric similarities and spatial relations for point motion encoding. HPLFlow-Net [37] proposes several modules to restore structural information from point clouds and fuse information. FESTA [38] proposes a spatial-temporal attention mech anism to estimate 3D scene flow from point clouds. Bi-PointFlowNet [39] proposes a bi-directional flow embedding module along forward and backward directions to enhance the estimate performance. Although these methods have achieved significant performances, they introduce an excessive number of parameters, making the networks overly complex. Therefore, FLOT [40] utilizes optimal transport tools to estimate scene flow and achieves competitive performance with less parameters. SPFlowNet [41] proposes an iterative superpoint-based framework that could simultaneously optimize the flow-guided superpoint generation and superpoint-guided flow refinement. Inspired by these works, we utilize the flow information for the 3D SOT task.

III. METHODOLOGY

A. Problem Definition

Given the 3D bounding box $\mathcal{B}_0 = (x_0, y_0, z_0, w, h, l, \theta_0) \in \mathbb{R}^7$ of the specific target in the initial frame, 3D SOT aims to locate the target object in subsequent frames and predict its target state $\mathcal{B}_t = (x_t, y_t, z_t, \theta_t) \in \mathbb{R}^4$ at t frame, where (x, y, z) is the bounding box center, and θ is the heading angle. Meanwhile, as we usually assume that the size of the target object (w, h, l) remains constant throughout the tracking process [2], [3], [9], [11], we directly use the size value given in the first frame.

Suppose we have N consecutive frames point cloud at t frame ranging from $\mathcal{P}_{t-N}$ to $\mathcal{P}_t$. Meanwhile, the previously $N-1$ predicted target boxes $\{\mathcal{B}_i\}_{t-N}^{t-1}$ could be preserved for utilization. For previous geometry matching paradigm, the SOT task could be formulated as follows:

$$\mathcal{F}(\{\mathcal{B}_i\}_{i=t-N}^{t-1}, \{\mathcal{P}_j\}_{j=t-N}^{t}) \mapsto \mathcal{B}_t \tag{1}$$

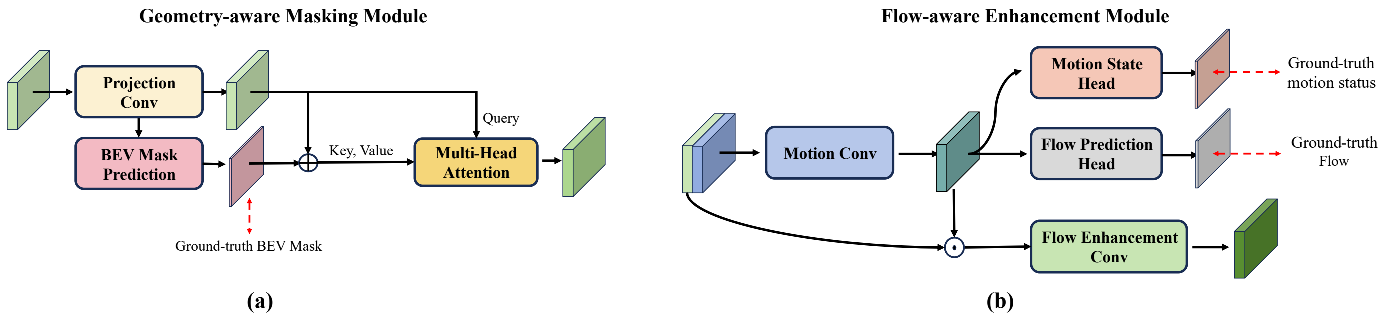

Fig. 3. **(a) Detailed structure of Geometry-aware Masking Module.** For each historical frame feature, we first predict a BEV box mask, and use the predicted mask to enhance the feature with multi-head attention to output a geometry-aware feature. **(b) Detailed structure of Flow-aware Enhancement Module.** For each historical frame feature, we first concatenate it with the current frame feature to generate a flow feature, followed by the motion state and flow prediction heads. Finally, the flow feature is fused with the historical frame feature to output the flow-aware features. $\odot$, $\oplus$ means feature concatenation and feature adding respectively.

Meanwhile, for the motion paradigm [11], the SOT could be formulated as follows:

$$\mathcal{F}(\mathcal{B}_{t-1}, \{\mathcal{P}_j\}_{j=t-1}^{t}) \mapsto \mathcal{M}_t, \mathcal{B}_t \tag{2}$$

where $\mathcal{M}_t$ is the segmented foreground mask at the current t frame. Based on $\mathcal{M}_t$, the foreground point at the current frame could be classified and used to regress the box transformation.

Different from the above two paradigms, our method could be formulated as follows:

$$\mathcal{F}(\{\mathcal{P}_j\}_{j=t-N}^{t}) \mapsto \{\mathcal{M}_i\}_{i=t-N}^{t-1}, \mathcal{B}_t, \{\mathcal{S}_{j\to t}\}_{j=t-N}^{t-1} \tag{3}$$

where $\mathcal{S}_{j\to t}$ is the flow from frame j to the current t frame. Therefore, in comparison to the above two paradigms, we not only predict the foreground mask for all historical frames but also forecast the flow field from every historical frame to the current frame.

B. Overview

Following Equation. 3, we propose the hybrid paradigm framework, HyGFNet, to leverage both geometry and flow information from historical frames to locate the target object in the current frame. As illustrated in Figure. 2, given N consecutive frames point clouds ranging from $\mathcal{P}_{t-N}$ to $\mathcal{P}_t$, we follow [42] to extract basic point feature of each frame, denoted by $\mathcal{X}_i \in \mathbb{R}^{W\times H\times C}$ for i-th frame. Here W, H are the grid size of the feature map, and C is the basic feature channel. Meanwhile, the proposed HyGFNet consists of two branches, named geometry matching branch and flow estimation branch, to process geometry and flow features respectively. Then, we propose a Geometry-aware Masking Module and a Flow-aware Enhancement Module to learn geometry and flow information in the two branches respectively. Finally, an attention-based spatial-temporal decoder is proposed to fuse the multi-frame geometry-aware and flow-aware features to augment the current frame feature for prediction. The details are presented in the following sections.

C. BEV Feature Extraction

Compared to images, point clouds are usually sparse, especially at long distances, making feature extraction and prediction difficult [6]. Therefore, some previous works usually directly voxelize raw point clouds to voxels to extract dense voxel features [5], [9] or utilized PointNet to extract point features and then transform the point features into dense BEV features for dense predictions [6]. In this work, to avoid this problem, we follow previous works [5], [9], [43] to convert points to pillars and directly extract dense BEV features. Specially, we first project the raw point cloud to the BEV X-Y plane and extract the projected pillar features using a mini PointNet [44], getting a sparse 2D pseudo-image. Then, we utilize a U-Net-like 2D CNN-based network [45], [46], which is widely used in 3D object detection, to process the pseudo-image with stride $1\times$, $2\times$, and $4\times$ 2D convolution blocks. The multi-scale features are then concatenated and generate a $2\times$ BEV dense basic feature map for subsequent processing.

D. Geometry-aware Masking Module

We first propose Geometry-aware Masking Module (GMM) to fully utilize the geometry information at each frame. As shown in Figure. 3(a), for basic feature $\mathcal{X}$ of each frame, we first use two Conv2D blocks, each of which consists of *conv-bn-relu* and termed as Projection Conv, to encode its feature. Then, a Conv2D layer and sigmoid layer are used to predict the BEV foreground mask $\mathcal{M} \in \mathbb{R}^{W\times H\times 1}$. By training the features to predict BEV box masks, we can enhance the ability of features to capture the geometric information of the target. Moreover, the predicted mask is further used to fuse with the projected feature. Specially, we use the attention mechanism [22] for its good integration of global information. The projected feature is flattened to $\mathcal{X}' \in \mathbb{R}^{WH\times C_1}$ and serves as query, key and value feature, while the mask is added to key and value to focus on the geometry information of foreground points as follows:

$$\mathcal{Q} = \text{Proj}(\mathcal{X}') \tag{4}$$
$$\mathcal{K} = \mathcal{V} = \text{Proj}(\mathcal{X}') + \mathcal{M} \tag{5}$$
$$\mathcal{A} = \text{softmax}\Big(\frac{\mathcal{Q}\cdot\mathcal{K}^T}{\sqrt{d}}\Big) \tag{6}$$
$$\mathcal{X}'' = \mathcal{A}\cdot\mathcal{V} \tag{7}$$

where d is the feature channels. To train the network to predict the BEV mask, we generate the ground-truth BEV box mask based on the history 3D ground-truth bounding box. Specially, the history 3D bounding box is converted to 2D BEV box, and all pixels within the BEV box will be considered as foreground, while otherwise is the background:

$$\mathcal{M}_p = \begin{cases} 1, & \text{if } p \in \mathcal{B}_{BEV} \\ 0, & \text{else} \end{cases} \tag{8}$$

Meanwhile, in order to simulate the historical prediction error in the inference phase, we randomly add offsets and rotation to the historical ground-truth box in the training phase. Therefore, by predicting the BEV box mask and integrating it into the point feature, the GMM not only makes the feature capture the geometry information better, but also eliminates cumulative errors for historical predictions.

E. Flow-aware Enhancement Module

In parallel with the aforementioned GMM, we introduce the Flow-aware Enhancement Module (FEM) to enhance each historical frame feature. Different from the scene flow that is defined on each point, here we predict the flow on each grid point on the BEV X-Y plane. Meanwhile, because the flow is defined as the point-level motion between two frames, we need to utilize both the current frame feature and each past frame feature. Here we adopt a simple yet efficient approach that directly concatenates the two features. As shown in Figure. 3 (b), we first concatenate the i-th frame feature $\mathcal{X}_i$ and the current frame feature $\mathcal{X}_t$ along the feature channel, getting the basic flow feature $\mathcal{X}_{i\to t} \in \mathrm{R}^{W\times H\times 2C}$. Then, a series of 2D CNN blocks, termed as Motion Conv, are applied to the feature to learn the flow feature between the two frames. Followed by two motion prediction heads, we could obtain a dense BEV motion state and a dense BEV flow for each grid in the i-th historical frame feature. The motion state indicates whether each grid point in the BEV plane is in a moving or stationary state, while flow represents the BEV X-Y offset for each grid point. Finally, we add the flow feature to the basic i-th frame feature and use a flow enhancement conv which includes a 2D CNN block to process and enhance the fusion feature. The FEM could be formulated as follows:

$$\mathcal{X}_{i\to t} = f_1(\mathcal{X}_i \cdot \mathcal{X}_t) \tag{9}$$
$$\mathcal{S}_{i\to t} = h_1(\mathcal{X}_{i\to t}) \tag{10}$$
$$\mathcal{G}_{i\to t} = h_2(\mathcal{X}_{i\to t}) \tag{11}$$
$$\mathcal{X}'_{i\to t} = f_2(\mathcal{X}_i \cdot \mathcal{X}_{i\to t}) \tag{12}$$

where f_1, f_2, h_1, h_2 mean Motion Conv, Flow Enhancement Conv, Flow Head, and State Head respectively. $S_{i\to t}, G_{i\to t}$ mean the flow and the moving state BEV map respectively. $\bullet$ represents concatenation operation. By predicting the grid point-level motion states and flows, we can obtain more refined motion representations and capture more local flow information. By fusing the flow features used for prediction with the corresponding frame features, we obtain flow-aware features for subsequent spatial-temporal fusion.

To train the network to predict the dense motion state and flow, we need corresponding ground-truth flow labels to provide supervision. However, in the SOT task, we only have one 3D bounding box label for each batch, which is a sparser representation. To get the dense flow label, inspired by previous flow works [47], we generate the dense flow label based on the consecutive bounding boxes and point clouds. Given two frames of data along with their corresponding ground truth 3D bounding boxes $\mathcal{B}_i, \mathcal{B}_j$ and point clouds $\mathcal{P}_i, \mathcal{P}_j$, and suppose we need the flow label from i-th frame to j-th frame. We first compute the rigid transform $\mathcal{R}_{i\to j}$ and $\mathcal{T}_{i\to j}$ between $\mathcal{B}_i$ and $\mathcal{B}_j$, where $\mathcal{R}_{i\to j}$ and $\mathcal{T}_{i\to j}$ represent relative rotation and translation respectively. Then, we extract all foreground points $\mathcal{P}_i^f$ within the bounding boxes $\mathcal{B}_i$ from the point clouds $\mathcal{P}_i$. We apply the obtained translation $\mathcal{R}_{i\to j}$ and rotation $\mathcal{T}_{i\to j}$ to the cropped foreground points $\mathcal{P}_i^f$, and then use the transformed points subtract with $\mathcal{P}_i^f$, generating the dense flow label $\mathcal{S}_{i\to j}$. For the ground-truth motion state label $\mathcal{G}_{i\to j}$, we set a motion threshold α to classify each point into static or moving based on their flow. The generation process could be represented as follows:

$$\mathcal{R}_{i\to j}, \mathcal{T}_{i\to j} = \mathcal{B}_j - \mathcal{B}_i \tag{13}$$
$$\mathcal{P}_i^f = \mathrm{Crop}(\mathcal{B}_i, \mathcal{P}_i) \tag{14}$$
$$\mathcal{P}_{i\to j}^f = \mathcal{R}_{i\to j}[\mathcal{P}_i^f] + \mathcal{T}_{i\to j} \tag{15}$$
$$\mathcal{S}_{i\to j} = \mathcal{P}_{i\to j}^f - \mathcal{P}_i^f \tag{16}$$
$$\mathcal{G}_{i\to j} = \begin{cases} 1, & \text{if } \mathcal{S}_{i\to j} < \alpha \\ 0, & \text{else} \end{cases} \tag{17}$$

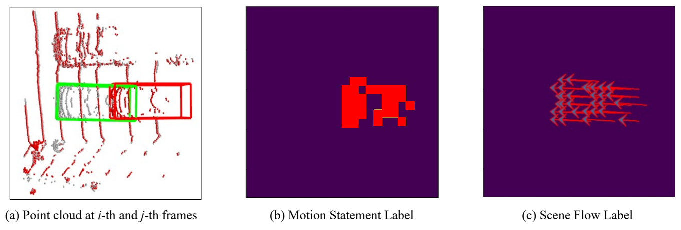
(a) Point cloud at i-th and j-th frames
(b) Motion Statement Label
(c) Scene Flow Label

Fig. 4. **Generated flow label.** (a) The two point cloud $\mathcal{P}_i$ and $\mathcal{P}_j$, shown in grey and red. Their corresponding 3D boxes $\mathcal{B}_i, \mathcal{B}_j$ are also shown in green and red respectively. (b) The generated motion state label, the red grids represent moving parts and the others are static. (c) The generated scene flow label.

As shown in Figure. 4, by processing the data with the above pipeline, we can obtain dense point-level flow labels from original sparse box-level labels, and we could train our module to predict the dense flow. We set α as 0.5 in our implementation.

F. Spatial-Temporal Decoder

After getting the geometry-aware feature and flow-aware feature for each historical frame, we would like to fuse these features with the current frame feature and thus propagate the geometry and flow information to the current for better prediction. Considering the global dependence of attention mechanism [22], we propose a transformer-based Spatial-Temporal Decoder (STD) to extract global information from all N frame features. The proposed STD is applied in both geometry and flow branches. As shown in Figure. 5, we first use a shared 2D CNN-based block (shared within each branch) to project each frame geometry-aware or flow-aware feature to a shared feature space for attention computation. Then, the

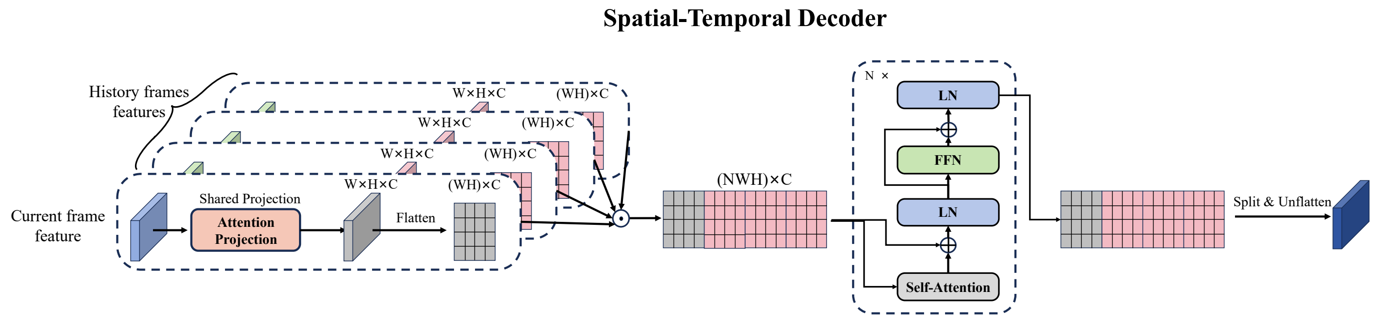

Fig. 5. **The detail of proposed spatial-temporal decoder (STD).** Firstly, each frame feature is projected to the same feature space by a shared 2D CNN block. Then, a spatial-temporal feature is constructed by concatenating the multi-frame features. N layers self-attention and feed-forward network (FFN) are following to extract global spatial-temporal information. Finally, the current frame feature is split and unflatten back for the prediction. $\odot$ means feature concatenation.

projected feature is flattened to the size of $(WH) \times C_2$, and all flatted N frame features are concatenated, constructing the spatial-temporal feature with size $(NHW) \times C_2$. The spatial-temporal feature is processed by the self-attention-based decoder to model the global geometry or flow information. Finally, the decoded feature is split into $N \times (HW) \times C_2$ features and the current frame feature is unflatted back into $W \times H \times C_2$. By using the STD in both branches, we obtain the final geometry-aware feature and flow-aware feature. We fuse the two features with the original basic feature $\mathcal{X}_t$ and input the fused current frame feature to the head to predict the current box.

Another approach is taking the current feature as query, the other feature as key and value, and using the cross-attention to fuse them. However, in cross-attention, there is no explicitly defined query-key relationship; instead, it requires an implicit calculation using $\mathrm{softmax}(QK^T)$. Differently, in self-attention, the concatenated multi-frame features inherently incorporate the explicit spatial-temporal information. Therefore, we can better model global spatio-temporal information. We will provide a more detailed comparison in the ablation study section.

By using the attention-based decoder to fuse the multi-frame features, we could learn spatial-temporal geometry and flow information for the target object. Meanwhile, we use the deformable attention [48] to replace the vanilla attention, avoiding too much computation cost and speeding up the training time.

G. Training Loss

We predict the target center and orientation with the fused feature in a center manner following CenterPoint [49]. Specially, we predict the heatmap, offset map, height map and orientation map with 2× downstride. Thus, the total head loss is formulated as follows:

$$L_{head} = \lambda_{cls} L_{cls} + \lambda_{offset} L_{offset} + \lambda_z L_z + \lambda_{yaw} L_{yaw} \quad (18)$$

where we use focal loss [50] as L_{cls} to train heatmap classification and adopt L1 loss as L_{offset}, L_z, L_{yaw} for the attribute regressions. $\lambda_{cls}, \lambda_{offset}, \lambda_z, \lambda_{yaw}$ are weights for these losses, respectively. Meanwhile, since we predict the historical BEV box mask and the BEV flow in the proposed GMB and FEB, we also need the following losses:

$$L_{branches} = \lambda_{mask} L_{mask} + \lambda_{flow} L_{flow} + \lambda_{state} L_{state} \quad (19)$$

We also use focal loss [50] as L_{state} to train motion state, L1 loss as L_{flow} and binary cross entropy as L_{mask}. Therefore, the overall training loss is:

$$L = L_{head} + L_{branches} \quad (20)$$

IV. EXPERIMENTS

A. Experimental Settings

Datasets. We train and evaluate our proposed HyGFNet on two commonly used large-scale datasets, KITTI tracking dataset [51] and nuScenes dataset [52]. KITTI tracking dataset includes more than 20,000 manually labeled 3D objects using Velodyne HDL-64E 3D lidar (10HZ). We follow previous works setting [2], [3], [5] to split this dataset as follows: scenes 0-16 for training, scenes 17-18 for validation, and scenes 19-20 for testing. nuScenes [52] is a larger dataset containing a total of 1,000 scenes with 20s duration in each. Each frame contains about 300,000 points and has 360-degree view annotations for various objects. Compared to KITTI dataset, nuScenes is more challenging for object tracking since it has more challenging scenes. Following the setting of previous works [3], [11], we also adopt the official 700 train scenes to train and the 150 val scenes to test.

Evaluation Metrics. Following previous works [2], [3], [11], we also use One Pass Evaluation (OPE) as the evaluation metric, including *Success* and *Precision*. Specially, the *Success* measures the 3D IoU between the predicted box and ground truth box, ranging from 0 to 1, while the *Precision* measures the accuracy between the predicted target center and ground-truth center, ranging from 0 to 2 meters. We report the two metrics for each category and then compute the two types of Mean performance. The F-Mean is computed based on the number of frames in each category and could be formulated as follows:

$$V^F_{mean} = \frac{\sum_{n=1}^{N} V_n \times F_n}{\sum_{n=1}^{N} F_n} \quad (21)$$

TABLE I
PERFORMANCE COMPARISON ON THE KITTI DATASET.

	Method	Paradigm	Car *6,424*	Ped. *6,088*	Van *1,248*	Cyc. *308*	F-Mean *14,068*	C-Mean *14,068*
Success	SC3D [1]	Match	41.3	18.2	40.4	41.5	31.2	35.4
	P2B [2]		56.2	28.7	40.8	32.1	42.4	39.5
	PTT [4]		67.8	44.9	43.6	37.2	55.1	48.4
	BAT [3]		60.5	42.1	52.4	33.7	51.2	47.2
	LTTR [5]		65.0	33.2	35.8	66.2	48.7	50.1
	V2B [6]		70.5	48.3	50.1	40.8	58.4	52.4
	C2FT [53]		67.0	48.6	53.4	38.0	57.2	51.8
	MLSET [54]		69.7	50.7	55.2	41.0	59.6	54.2
	PTTR [7]		65.2	50.9	52.5	65.1	57.9	58.4
	DMT [55]		66.4	48.1	53.3	70.4	55.1	59.6
	SMAT [9]		71.9	52.1	41.4	61.2	60.4	56.7
	STNet [10]		72.1	49.9	58.0	73.5	61.3	63.4
	GLT-T [56]		68.2	52.4	52.6	68.9	60.1	60.5
	PCET [57]		68.7	56.9	57.9	75.6	**64.8**	64.8
	CAT [58]		66.6	51.6	53.1	67.0	58.9	59.6
	PTIT [59]		68.6	56.7	53.8	74.2	62.6	63.3
	CorpNet [60]		**73.6**	55.6	**58.7**	74.3	64.5	**65.6**
	OSP2B [61]		67.5	53.6	56.3	65.6	60.5	60.8
	M^2-Tracker [11]	Motion	65.5	**61.5**	53.8	73.2	62.9	63.5
	Ours	Hybird	68.0	60.0	56.7	**75.9**	63.7	65.2
Precision	SC3D [1]	Match	57.9	37.8	47.0	70.4	48.5	53.3
	P2B [2]		72.8	49.6	48.4	44.7	60.0	53.9
	PTT [4]		81.8	72.0	52.5	47.3	74.2	63.4
	BAT [3]		77.7	70.1	67.0	45.4	72.8	65.1
	LTTR [5]		77.1	56.8	45.6	89.9	65.8	67.4
	V2B [6]		81.3	73.5	58.0	49.7	75.2	65.6
	C2FT [53]		80.4	75.6	66.1	48.7	76.4	67.7
	MLSET [54]		81.0	80.0	64.8	49.7	78.4	68.9
	PTTR [7]		77.4	81.6	61.8	90.5	78.1	77.8
	DMT [55]		79.4	77.9	65.6	92.6	75.8	78.9
	SMAT [9]		82.4	81.5	53.2	87.3	79.5	76.1
	STNet [10]		84.0	77.2	70.6	93.7	80.1	81.4
	GLT-T [56]		82.1	78.8	62.9	92.1	79.3	79.0
	PCET [57]		80.1	85.1	66.1	93.7	81.3	81.3
	CAT [58]		81.8	77.7	69.8	90.1	79.1	79.9
	PTIT [59]		81.2	86.3	**70.7**	92.5	82.7	82.7
	CorpNet [60]		**84.1**	82.4	66.5	**94.2**	82.0	81.8
	OSP2B [61]		82.3	85.1	66.2	90.5	82.3	81.0
	M^2-Tracker [11]	Motion	80.8	**88.2**	**70.7**	93.5	83.4	83.3
	Ours	Hybird	79.3	86.6	67.1	93.9	**85.6**	**85.2**

TABLE II
PERFORMANCE COMPARISON BETWEEN OURS AND M^2-TRACKER ON THE MODIFIED KITTI.

	Category / Frame Number	Car *1328*	Pedestrian *1248*	Van *255*	Cyclist *65*	F-Mean *2896*	C-Mean *2896*
Success	M^2-Tracker [11]	40.4	19.9	16.4	16.6	28.9	26.4
	Ours	**58.1**	**28.1**	**28.9**	**46.0**	**42.3**	**40.3**
Precision	M^2-Tracker [11]	46.9	34.0	16.0	17.3	38.0	38.1
	Ours	**68.0**	**40.9**	**35.1**	**60.0**	**53.2**	**51.0**

TABLE III
PERFORMANCE COMPARISON ON THE NUSCENES DATASET. PED AND BIC IS AN ABBREVIATION FOR PEDESTRIAN AND BICYCLE. - MEANS NOT AVAILABLE.

	Method	Modality	Car *64,159*	Ped. *33,227*	Truck *13,587*	Bic. *2,292*	F-Mean *113,265*	C-Mean *113,265*
Success	SC3D [1]	Match	22.31	11.29	30.67	16.70	19.97	20.24
	PTT [4]		41.22	19.33	50.23	28.39	35.52	34.79
	P2B [2]		38.81	28.39	42.95	26.32	35.99	34.12
	BAT [3]		40.73	28.83	45.34	27.17	37.52	35.52
	PTTR [7]		58.61	45.09	44.74	-	-	-
	SMAT [9]		43.51	32.27	44.78	25.74	40.00	36.58
	CAT [58]		43.34	30.68	47.64	-	-	-
	PTIT [59]		52.50	-	51.60	-	-	-
	M^2-Tracker [11]	Motion	55.85	32.10	**57.36**	36.32	48.67	45.41
	Ours	Hybrid	**58.52**	**40.76**	55.32	**37.72**	**52.51**	**48.08**
Precision	SC3D [1]	Match	21.93	12.65	27.73	28.12	20.03	22.61
	PTT [4]		45.26	32.03	48.56	51.19	41.89	44.26
	P2B [2]		43.18	52.24	41.59	47.80	45.74	46.20
	BAT [3]		43.29	53.32	42.58	51.37	46.31	47.64
	PTTR [7]		51.89	29.90	45.30	-	-	-
	SMAT [9]		49.04	60.28	44.69	61.06	52.06	52.77
	CAT [58]		49.41	56.67	48.10	-	-	-
	PTIT [59]		61.90	-	52.30	-	-	-
	M^2-Tracker [11]	Motion	65.09	60.92	59.54	67.50	63.23	63.26
	Ours	Hybrid	**72.14**	**73.05**	**60.74**	**71.88**	**71.03**	**69.45**

where V_n and F_n mean the value and frames of each category, and N is the number of categories. The C-Mean is computed by averaging all categories as:

$$V^C_{mean} = \frac{\sum_{n=1}^{N} V_n}{N} \tag{22}$$

Implementation Details. For our pillar-based network, we need a fixed search region to crop the point cloud to input to the pillar backbone. Meanwhile, different types of objects have different sizes, thus we set different point cloud ranges for different objects. Specially, we set the point cloud range of [(-4.8, 4.8), (-4.8, 4.8), (-3, 3)] meters and voxel size of [0.075, 0.075, 6] meters for Car and Van. For cyclist, we set a point cloud range of [(-3.2, 3.2), (-3.2, 3.2), (-3, 1)] meters and a voxel size of [0.05, 0.05, 6] meters. For pedestrian, [(-1.6, 1.6), (-1.6, 1.6), (-3, 1)] meters and [0.025, 0.025, 4] meters are set for point cloud range and voxel size. For truck, we set [(-12.8, 12.8), (-12.8, 12.8), (-10, 10)] meters and [0.2, 0.2, 20] meters for point cloud range and voxel size. [(-12.8, 12.8), (-12.8, 12.8), (-10, 10)] meters and [0.2, 0.2, 20] meters for Truck. For network details, we use the dynamic pillar [42] and a U-Net-like 2D CNN network [46], which are widely used in 3D detection, as our 3D and 2D backbone. For our attention-based decoder, we follow the common setting in Vision Transformer [62], [63] and set heads as 8, layers as 6. Moreover, we train our model with adam optimizer with the initial learning rate of 0.003, weight decay of 0.01 for both datasets, and train 80 epochs in KITTI but 40 epochs in nuScenes. We use 4 NVIDIA 3090 GPUs and set 32 batch size in each in the training phase.

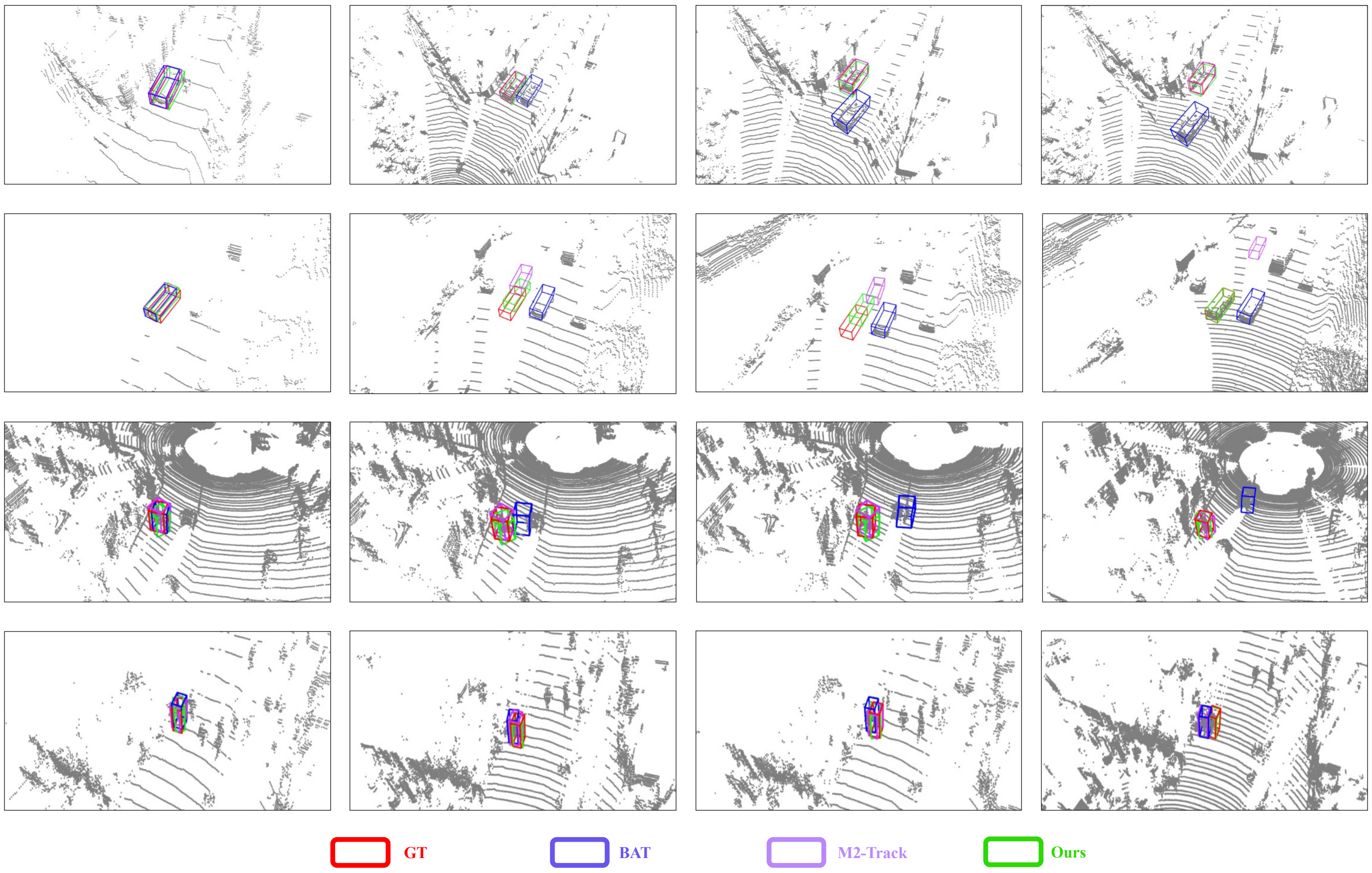

Fig. 6. **Visualization of tracking results on KITTI dataset.** From top to bottom, we compare our HyGFNet with BAT, M^2-Tracker and show the cases of Car and Pedestrian categories.

B. State-of-the-arts Comparison

Quantitative results on KITTI. As shown in Table. I, our HyGFNet achieves competitive tracking performance in the KITTI tracking dataset. Specially, our method outperforms the motion paradigm method M^2-Tracker [11] by 0.8%/1,7% in F-Mean/C-Mean Success respectively. Moreover, for previous match paradigm methods [3], [7], [9], [10], [57], [60], although they could achieve better tracking performance in the Car category [9], [10], [57], [60], they usually have limited tracking performance in the Pedestrian category. Compared to Car, Pedestrian targets usually have sparser point clouds and are more sensitive to interference from surrounding pedestrians. Therefore, it is challenging for match paradigm methods to track Pedestrian objects accurately. However, by utilizing the flow information from history frames, our method could achieve a better performance in Pedestrian tracking, surpassing PCET [57], which is the best match paradigm in the Pedestrian category, by 3.1% and 1.5% in Success and Precision respectively. The results indicate that by combining motion and match paradigms, we can achieve better and more balanced tracking results.

Moreover, to explore the tracking performance with larger relative motion, we compare our method with M^2-Tracker in a modified KITTI dataset. Specially, to build the dataset, we take one frame out of every 5 frames in KITTI dataset as a valid frame. Compared to the original KITTI dataset, the modified dataset has a larger relative motion between two consecutive frames. We train our method and M^2-Tracker in the modified dataset. We follow their public default setting[1] to train M^2-Tracker. As shown in Table. II, our method still achieves better performance, surpassing the M^2-Tracker both in Success and Precision. The results verify our discussion in Section. I that the motion paradigm methods usually have difficulty tracking the target object with larger motion. By using the geometry matching information, our method could alleviate this problem and track the target object better.

Quantitative results on nuScenes. As shown in Table. III, our method also achieves state-of-the-arts tracking performance in the nuScenes dataset. Specially, our method outperforms M^2-Tracker by 3.84% and 2.67% in F-Mean and C-Mean Success respectively. We believe that as analyzed in the previous section of the KITTI comparison, by combining the advantages of geometric matching and flow estimation paradigms, our method could achieve better tracking results. Meanwhile, compared to previous geometry matching paradigm methods [2]–[4], [9], [58], our method improves the tracking performance in all categories, especially in the Pedestrian category. Meanwhile, compared to the KITTI dataset, nuScenes dataset has much **more testing frames** (113265 vs 14068), **more testing scenes** (150 vs 2) and **sparser points** (32-beam vs 64-beam). The results also verify the effectiveness and the scalability of

[1]https://github.com/Ghostish/Open3DSOT/tree/main/cfgs

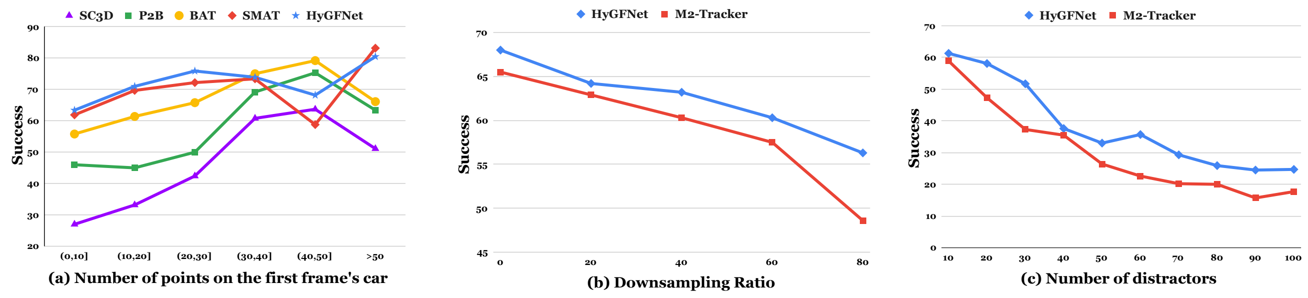

Fig. 7. Robustness Comparison. **(a) Effect of the number of points in the first frame.** In most cases, our HyGFNet outperforms other methods by a significant margin. **(b) Effect of different point cloud downsampling ratios.** Compared to M^2-Tracker, our method shows better performances under different downsampling ratios. **(c) Effect of different number of distractors.** Compared to M^2-Tracker, our method still achieves better performances when facing different numbers of distractors.

TABLE IV
TIME CONSUMPTION OF EACH MODULE IN OUR HYGFNET.

Module	Backbone	GMM	FEM
Time	17.16 ms	4.69 ms	4.01 ms
Module	G-STD	F-STD	Head
Time	8.76 ms	7.81 ms	1.79 ms

TABLE V
COMPARISON OF THE RUNNING SPEEDS.

Method	SC3D [1]	P2B [2]	PTT [4]	BAT [3]	LTTR [5]
FPS	1.8	45.5	40.0	57.0	22.3
Method	V2B [6]	C2FT [53]	MLSET [54]	PTTR [7]	DMT [55]
FPS	13.0	50.0	61.0	71.5	51.0
Method	OSP2B [61]	SMAT [9]	STNet [10]	GLT-T [56]	PCET [57]
FPS	34.0	17.6	35.0	30.0	50.0
Method	CAT [58]	PTIT [59]	CropNet [60]	M^2-Tracker [11]	Ours
FPS	45.3	32.8	36.0	57.0	22.6

our method.

Visualization results on KITTI. We also visualize some tracking results in KITTI tracking dataset in Figure. 6. As shown in the 1st and 4th rows, when there comes a similar distractor object, our method could continue to consistently track the target object, while the matching paradigm method, BAT [3], usually loses the target object. In the scene with a few points, such as the 2nd case, BAT and M^2-Tracker gradually lose the target object. However, although it is also very difficult for our method to track the target, we ultimately returned to the target object.

Running speed. The time consumed by each module of the network is shown in detail in Tab. IV. Meanwhile, we also compare our method with other methods in running speed in Table. V. Our speed still could satisfy the real-time requirements, *i.e.* (>10 FPS). Meanwhile, compared to the other BEV-based methods, such as LTTR [5], V2B [6], SMAT [9], though we need to process more frame point clouds, our method still shows faster running speed.

C. Robustness Comparison

To further explore the robustness of our method, we compare our HyGFNet with other methods in some challenge scenes. We conduct these robustness comparison on the Car category of KITTI dataset.

Robustness to Sparseness. We follow previous works [1]–[3], [9] to compare the tracking performance under different numbers of points in the first frame. As shown in Figure. 7 (a), our method achieves better balanced and higher performance in most comparison sparse levels, especially in the targets with fewer than 30 points.

Robustness to Downsampling. We further compare our HyGFNet with M^2-Tracker with different point cloud downsampling ratios to explore the robustness to the sparsity of input point cloud. We downsample the point clouds with different sampling ratios and input the downsampled point clouds to the compared two models. As shown in Figure. 7 (b), our method achieves better performance under different sampling ratios. The result further confirms the robustness of our method to sparsity.

Robustness to Distractors. Finally, to explore the robustness of distractors, we add multiple objects of the same category with the target object (car) in the tracking scenes. As shown in Figure. 7 (c), we compare our HyGFNet with M^2-Tracker with different numbers of distractors. Although our method and M^2-Tracker both have a large performance decline, our HyGFNet still achieves better performance, verifying our robustness to distractors.

D. Ablation Study

To validate the effectiveness of our proposed modules, including GMM, FEN and STD, we conduct ablation studies on the Car category of KITTI dataset.

Model Components. We first conduct ablation studies on the main components to explore the effects of each module. As shown in Table. VI, we first build a baseline model (A1) that directly fuses the input multi-frame features with concatenating and 2D CNN blocks. We then add our proposed GMM to the baseline, leading to an improvement of 3.5% and 3.8% in Success and Precision respectively. Moreover, the proposed STD (A3) further improves the performance. For the flow estimation branch, the proposed FEM brings 4.8% and 7.1% improvements in Success and Precision, and the proposed STD added in the flow estimation branch (A5) also achieves higher performance. Meanwhile, we also notice

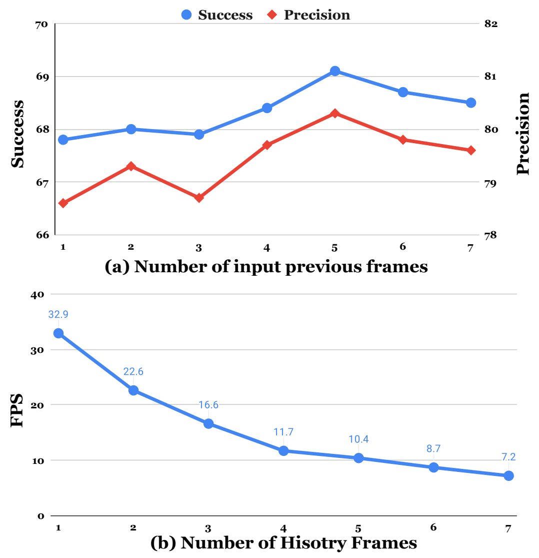

Fig. 8. **Performances with different number of previous frames.** We input different numbers of previous frames, from $t-1$ to $t-5$. Our method could achieve higher performance with more previous frames.

TABLE VI
ABLATION OF OUR COMPONENTS. G- AND M- MEAN IN GEOMETRY AND FLOW BRANCHES RESPECTIVELY.

	Geometry		Motion		Performance	
	GMM	G-STD	FEM	M-STD	3D Success	3D Precision
A1					57.1	67.9
A2	✓				60.6	71.7
A3	✓	✓			61.6	73.8
A4			✓		61.9	75.0
A5			✓	✓	64.7	75.3
A6	✓	✓	✓	✓	68.0	79.3

TABLE VII
ABLATION STUDIES OF THE PROPOSED GMM.

	GMM	3D Success	3D Precision
G1	w/o mask prediction	64.2	74.6
G2	dot Key Value	66.6	77.9
G3	Conv Fusion	64.6	74.3

that the FEM could bring higher improvements than GMM, and M-STD also achieves higher performance than G-STD. We believe that the results imply that motion plays a more important role than matching in 3D SOT, which is also coherent with the perspective of M^2-Tracker. Finally, by utilizing all proposed modules in both the geometry matching branch and the flow estimation branch, our method could achieve better tracking performance, verifying the effectiveness of our proposed modules.

Input Frames. Since our method utilizes multiple frames as our input, we also conduct an ablation study to explore the effect of input historical frames. As shown in Figure. 8, our method's performance gradually improves with an increase in the number of frames. Meanwhile, our method still achieves a good performance with only $t-1$ and t frames, showing the robustness to the number of frames. We also show the running speed of the proposed method under different input frames, shown in Figure. 8(b). Our method could achieve 32.9 FPS with only 1 history frame. However, inputting too many historical frames introduces too much computational complexity and longer training times. Therefore, we adopt the 2 historical input frames as our default setting.

Components in GMM. We compare different feature fusion methods within the proposed GMM shown in Table. VII. First, as shown in G1, by removing the BEV box mask prediction, our method has a decrease in performance by 3.8% and 4.7% in Success and Precision respectively. The result demonstrates the significance of historical BEV mask predictions. Meanwhile, by using the predicted mask multiplies with Key and Value, G2 achieves 66.6% and 77.9%. We believe that the dot multiplication makes all the predicted background features zero, breaking the geometry information of the target. Meanwhile, the prediction errors in the mask also eliminate information about correct foreground points. Finally, G3 replaces the multi-head attention with a 2D CNN block, leading to a performance decline of 3.4% and 5.0%. The result shows that for geometry information, it is better to use attention to extract global information rather than CNN to extract local information.

Components in FEM. We compare different flow enhancement strategies in Table. VIII. F1 only uses the flow feature and does not concatenate with the basic frame feature like Equation. 12, achieving 65.1% and 76.4% in Success and Precision respectively. The result shows the importance of the basic feature in FEM. F2 replaces the Conv-Fusion with attention-based fusion, leading to a 1.3% and 1.8% performance degeneration. We believe that the flow estimation focuses more on local information rather than global information, thus the attention mechanism with global perception may not be effective. Finally, removing the flow estimation prediction (F3) or motion statement prediction (F4) will result in a decrease in performance, indicating that they are indispensable.

Components in STD. Finally, we compare different spatial-temporal information fusion in Table. IX. S1 first replaces the proposed STD with 2D CNN blocks and achieves 66.0% and 75.5% in Success and Precision. The result indicates that simple convolutional fusion is not sufficient for multi-frame feature fusion. Moreover, to compare the self-attention with cross-attention in the proposed STD, we replace the cross-attention in the two branches STD (S2, S3, S4). As the results show, our "self-self" achieves the best performance. The results verify our previous hypothesis in Section III-F that self-attention performs better than cross-attention in extracting spatial-temporal information.

E. Limitation

Although our proposed HyGFNet achieves better performance, we still have some limitations. First, as Table. I and Table. III show, our method does not show better performance in large-size object tracking, such as Van in KITTI and Truck in nuScenes. As discussed in FSD [64], because points often

TABLE VIII
ABLATION STUDIES OF THE PROPOSED FEM.

	FEM	3D Success	3D Precision
F1	only motion	65.1	76.4
F2	Attention Fusion	66.7	77.5
F3	w/o Flow	64.8	77.5
F4	w/o State	64.7	74.3

TABLE IX
ABLATION STUDIES OF THE PROPOSED STD.

	STD			3D Success	3D Precision
	Conv	Attention			
S1	✓			66.0	75.5
S2		cross	cross	67.6	78.8
S3		self	cross	67.1	77.2
S4		cross	self	67.9	78.5
S5		self	self	68.0	79.3

reside on the surface of objects, and the center point of large objects is farther from the surface, the central point cloud is less dense, making it more challenging for center-based head [49] predictions. Second, for some extremely sparse or fast-moving scenes, it is also difficult for our method to track the target object accurately and stably.

V. CONCLUSION

In this paper, we introduce a novel hybrid paradigm method HyGFNet for 3D SOT task. Our proposed network consists of a geometry matching branch and a flow estimation branch. We first design a geometry-aware masking module to better learn geometry information at each historical frame. Then, we introduce a flow-aware enhancement module to enhance each frame feature with flow information which is used to predict flow and motion state. Finally, we propose a spatial-temporal decoder to fuse the multi-frame features and propagate the fused information to the current frame feature for better prediction. Extensive experiments show that our HyGFNet significantly outperforms previous matching and motion paradigm methods and achieves new state-of-the-arts tracking performance.

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*, 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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 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," 2021.

[4] J. Shan, S. Zhou, Z. Fang, and Y. Cui, "Ptt: Point-track-transformer module for 3d single object tracking in point clouds," 2021.

[5] Y. Cui, Z. Fang, J. Shan, Z. Gu, and S. Zhou, "3d object tracking with transformer," 2021.

[6] L. Hui, L. Wang, M. Cheng, J. Xie, and J. Yang, "3d siamese voxel-to-bev tracker for sparse point clouds," 2021.

[7] 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 *CVPR*, pp. 8531–8540, 2022.

[8] K. Wang, T. Zhou, X. Li, and F. Ren, "Performance and challenges of 3d object detection methods in complex scenes for autonomous driving," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 2, pp. 1699–1716, 2023.

[9] 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. 11926–11933, 2022.

[10] L. Hui, L. Wang, L. Tang, K. Lan, J. Xie, and J. Yang, "3d siamese transformer network for single object tracking on point clouds," *arXiv preprint arXiv:2207.11995*, 2022.

[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] L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik, and P. S. Torr, "Staple: Complementary learners for real-time tracking," in *CVPR*, pp. 1401–1409, 2016.

[13] P. Barcellos and J. Scharcanski, "Part-based object tracking using multiple adaptive correlation filters," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–10, 2021.

[14] D. Held, S. Thrun, and S. Savarese, "Learning to track at 100 fps with deep regression networks," in *European Conference on Computer Vision* (B. Leibe, J. Matas, N. Sebe, and M. Welling, eds.), pp. 749–765, 2016.

[15] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. S. Torr, "Fully-convolutional siamese networks for object tracking," in *European Conference on Computer Vision Workshops* (G. Hua and H. Jégou, eds.), pp. 850–865, 2016.

[16] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu, "High performance visual tracking with siamese region proposal network," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8971–8980, 2018.

[17] D. Guo, J. Wang, Y. Cui, Z. Wang, and S. Chen, "Siamcar: Siamese fully convolutional classification and regression for visual tracking," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6269–6277, 2020.

[18] Z. Chen, B. Zhong, G. Li, S. Zhang, and R. Ji, "Siamese box adaptive network for visual tracking," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6668–6677, 2020.

[19] G. Bhat, M. Danelljan, L. Van Gool, and R. Timofte, "Learning discriminative model prediction for tracking," in *IEEE/CVF International Conference on Computer Vision*, 2019.

[20] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg, "Atom: Accurate tracking by overlap maximization," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4655–4664, 2019.

[21] Y. Zhu, C. Li, J. Tang, and B. Luo, "Quality-aware feature aggregation network for robust rgbt tracking," *IEEE Transactions on Intelligent Vehicles*, vol. 6, no. 1, pp. 121–130, 2021.

[22] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, vol. 30, Curran Associates, Inc., 2017.

[23] X. Chen, B. Yan, J. Zhu, D. Wang, X. Yang, and H. Lu, "Transformer tracking," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8126–8135, 2021.

[24] N. Wang, W. Zhou, J. Wang, and H. Li, "Transformer meets tracker: Exploiting temporal context for robust visual tracking," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1571–1580, 2021.

[25] B. Yan, H. Peng, J. Fu, D. Wang, and H. Lu, "Learning spatio-temporal transformer for visual tracking," 2021.

[26] Y. Cui, C. Jiang, L. Wang, and G. Wu, "Mixformer: End-to-end tracking with iterative mixed attention," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13608–13618, 2022.

[27] C. Mayer, M. Danelljan, G. Bhat, M. Paul, D. P. Paudel, F. Yu, and L. Van Gool, "Transforming model prediction for tracking," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8731–8740, 2022.

[28] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16000–16009, 2022.

[29] Q. Wu, T. Yang, Z. Liu, B. Wu, Y. Shan, and A. B. Chan, "Dropmae: Masked autoencoders with spatial-attention dropout for tracking tasks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14561–14571, 2023.

[30] 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.

[31] S. Vedula, S. Baker, P. Rander, R. Collins, and T. Kanade, "Three-dimensional scene flow," in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, vol. 2, pp. 722–729, IEEE, 1999.

[32] S. Hadfield and R. Bowden, "Kinecting the dots: Particle based scene flow from depth sensors," in *2011 International Conference on Computer Vision*, pp. 2290–2295, IEEE, 2011.

[33] E. Herbst, X. Ren, and D. Fox, "Rgb-d flow: Dense 3-d motion estimation using color and depth," in *2013 IEEE international conference on robotics and automation*, pp. 2276–2282, IEEE, 2013.

[34] M. Jaimez, M. Souiai, J. Gonzalez-Jimenez, and D. Cremers, "A primal-dual framework for real-time dense rgb-d scene flow," in *2015 IEEE international conference on robotics and automation (ICRA)*, pp. 98–104, IEEE, 2015.

[35] X. Liu, C. R. Qi, and L. J. Guibas, "Flownet3d: Learning scene flow in 3d point clouds," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 529–537, 2019.

[36] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *Advances in Neural Information Processing Systems*, vol. 30, Curran Associates, Inc., 2017.

[37] X. Gu, Y. Wang, C. Wu, Y. J. Lee, and P. Wang, "Hplflownet: Hierarchical permutohedral lattice flownet for scene flow estimation on large-scale point clouds," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 3254–3263, 2019.

[38] H. Wang, J. Pang, M. A. Lodhi, Y. Tian, and D. Tian, "Festa: Flow estimation via spatial-temporal attention for scene point clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14173–14182, 2021.

[39] W. Cheng and J. H. Ko, "Bi-pointflownet: Bidirectional learning for point cloud based scene flow estimation," in *European Conference on Computer Vision*, pp. 108–124, Springer, 2022.

[40] G. Puy, A. Boulch, and R. Marlet, "Flot: Scene flow on point clouds guided by optimal transport," in *European conference on computer vision*, pp. 527–544, Springer, 2020.

[41] Y. Shen, L. Hui, J. Xie, and J. Yang, "Self-supervised 3d scene flow estimation guided by superpoints," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5271–5280, 2023.

[42] Y. Zhou, P. Sun, Y. Zhang, D. Anguelov, J. Gao, T. Ouyang, J. Guo, J. Ngiam, and V. Vasudevan, "End-to-end multi-view fusion for 3d object detection in lidar point clouds," in *Conference on Robot Learning*, pp. 923–932, PMLR, 2020.

[43] Z. Meng, X. Xia, R. Xu, W. Liu, and J. Ma, "Hydro-3d: Hybrid object detection and tracking for cooperative perception using 3d lidar," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 8, pp. 4069–4080, 2023.

[44] 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.

[45] Y. Yan, Y. Mao, and B. Li, "Second: Sparsely embedded convolutional detection," in *Sensors*, vol. 18, 2018.

[46] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

[47] P. Wu, S. Chen, and D. N. Metaxas, "Motionnet: Joint perception and motion prediction for autonomous driving based on bird's eye view maps," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11385–11395, 2020.

[48] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, "Deformable detr: Deformable transformers for end-to-end object detection," 2021.

[49] 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*, pp. 11784–11793, 2021.

[50] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017.

[51] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3354–3361, 2012.

[52] 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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[53] 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*, vol. 7, no. 4, pp. 12211–12218, 2022.

[54] 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, 2023.

[55] Y. Xia, Q. Wu, W. Li, A. B. Chan, and U. Stilla, "A lightweight and detector-free 3d single object tracker on point clouds," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 5, pp. 5543–5554, 2023.

[56] J. Nie, Z. He, Y. Yang, M. Gao, and J. Zhang, "Glt-t: Global-local transformer voting for 3d single object tracking in point clouds," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, pp. 1957–1965, 2023.

[57] P. Wang, L. Ren, S. Wu, J. Yang, E. Yu, H. Yu, and X. Li, "Implicit and efficient point cloud completion for 3d single object tracking," *IEEE Robotics and Automation Letters*, vol. 8, no. 4, pp. 1935–1942, 2023.

[58] J. Gao, X. Yan, W. Zhao, Z. Lyu, Y. Liao, and C. Zheng, "Spatio-temporal contextual learning for single object tracking on point clouds," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–13, 2023.

[59] J. Liu, Y. Wu, M. Gong, Q. Miao, W. Ma, and F. Xie, "Instance-guided point cloud single object tracking with inception transformer," *IEEE Transactions on Instrumentation and Measurement*, vol. 72, pp. 1–12, 2023.

[60] M. Wang, T. Ma, X. Zuo, J. Lv, and Y. Liu, "Correlation pyramid network for 3d single object tracking," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 3216–3225, 2023.

[61] J. Nie, Z. He, Y. Yang, Z. Bao, M. Gao, and J. Zhang, "Osp2b: One-stage point-to-box network for 3d siamese tracking," *arXiv preprint arXiv:2304.11584*, 2023.

[62] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2021.

[63] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10012–10022, 2021.

[64] L. Fan, F. Wang, N. Wang, and Z.-X. ZHANG, "Fully sparse 3d object detection," *Advances in Neural Information Processing Systems*, vol. 35, pp. 351–363, 2022.