MLCFNet: Multi-Level Context Fusion Network for 3D Object Tracking

Yongchang Zhang[1,2], Hanbing Niu[3], Yue Guo[1], Wenhao He[1,2]
[1]*Institute of Automation, Chinese Academy of Sciences, Beijing, China*
[2]*School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China*
[3]*University of Electronic Science and Technology of China, Chengdu, China*
{zhangyongchang2020, guoyue2013, wenhao.he}@ia.ac.cn, niuhanbing2021@std.uestc.edu.cn

***Abstract*—Due to exhaustive proposal generations, conventional 3D object tracking methods based on template matching are time-consuming. Some recent works that leverage template clues to directly obtain 3D boxes are more efficient, but they don't take full advantage of the template context. In this work, we propose a novel Multi-Level Context Fusion Network (MLCFNet) to track objects robustly. Our main idea is to fuse template context in multiple levels (point, local, and global features) into the search area and utilize the joint information to predict the final box. Specifically, a 3D Siamese Network firstly extracts multi-level features in the search area and template. Then, to promote the guidance of the template, a Context Fusion Network fuses these features into the search area and generates guided points. Finally, these points are used to regress potential object centers and cluster 3D object proposals. Experiments on KITTI and nuScenes tracking datasets demonstrate that MLCFNet outperforms other state-of-the-art methods by a large margin.**

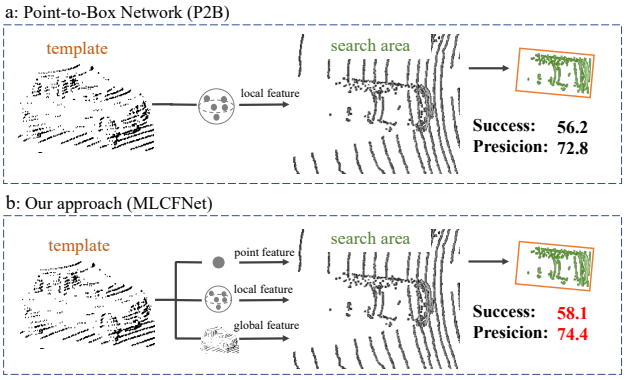

Fig. 1. Comparison between the state-of-the-art method [7] and ours.

I. Introduction

Object tracking, a key field in robotics and computer vision, involves many applications, such as self-driving cars and mobile robotics. Normally, the tracking process is that once an object is initially detected, the robot tracks the one for a while. The robot determines its tracking policy by sensing its surroundings. Modules for object detection [1], [2] and path recognition [3]–[6] instruct the robot where and how to run.

Traditionally, 2D RGB cameras, with the advantages of extracting rich illumination information, are widely equipped in existing tracking systems. But visual tracking methods may fail when 2D RGB visual information is degraded with illumination change. In addition, 2D images lack the accurate space information which is essential for object following and obstacle avoidance in practice.

Nowadays, 3D LIDAR systems are widely used in autonomous vehicles and intelligent robots. Compared to 2D RGB cameras, LIDAR sensors directly sense geometric structures and generate point clouds more accurately. Furthermore, 3D LIDAR sensors are insensitive to illuminations, so they provide reliable point clouds in a larger range of the real-world scene.

Towards the 3D object tracking, we focus on using point clouds only and propose a end-to-end 3D object tracking framework, Multi-Level Context Fusion Network, to improve the performance of the state-of-the-art Point-to-Box Network (P2B) [7]. Different from P2B, our work fuses multi-level context (point, local, and global features) from the template into the search area to guide 3D object tracking. As shown in Fig. 2, we firstly feed the template and the search area into the shared backbone to obtain points and local features of key points, respectively. Then, we use Context Fusion Network to fuse multi-level features into the search area and generate guided points. The multi-level context from the template mainly contain four components: 1) point features: 3D coordinates to retain spatial geometric information; 2) local features: point-wise features from the backbone to obtain the local contexts; 3) similarity: the cosine distance between the template and the search area to mine resembling patterns and reveal the local tracking clue; 4) global features: global features by max-pooling to obtain the global tracking guidance. Finally, we send these guided points to the Vote-Proposal Net to predict the final 3D box.

Experiments on KITTI [8] and nuScenes [9] tracking dataset demonstrate that our method significantly outperforms other state-of-the-art methods and has better performance under the same experimental conditions.

The contributions of this paper can be summarized as follows:

1) We propose a novel network that utilizes the multi-level template context (point, local, and global features) to achieve 3D object tracking.
2) Compared with expensive search and rough template matching, we directly obtain the final box by point-wise

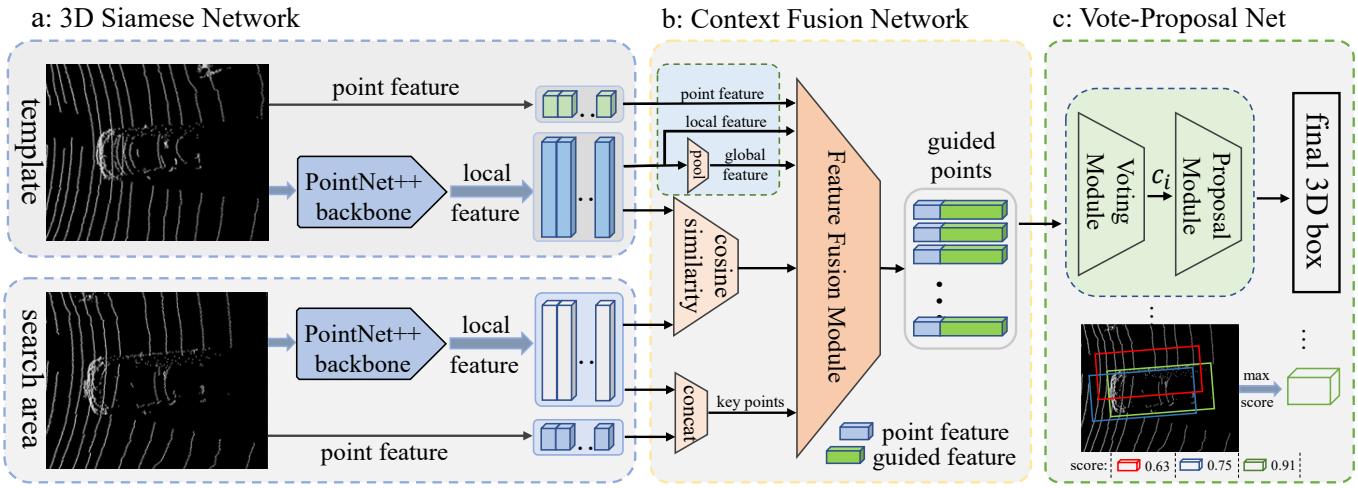

Fig. 2. The main structure of MLCFNet. MLCFNet has three parts: a) 3D Siamese Network, b) Context Fusion Network and c) Vote-Proposal Net. The backbone applies modified PointNet++ to extract the local features in template and search area. With the multi-level context fusion, Context Fusion Network utilizes the point, local and global features from template and similarity to instruct the search area to generate guided features and points. Then the Vote-Proposal Net regresses potential object centers and clusters 3D object proposals.

voting in guided search areas.

3) Extensive experiments demonstrate MLCFNet outperforms other state-of-the-art methods by a large margin.

II. Related Work

A. 2D Object Tracking in Images

Object tracking focuses on visual objects across consecutive frames such as people [10], vehicles [11], and visual attributes [12]. Early works relied on correlation filtering [13]–[15], which utilized the correlation between the template and the search area to precisely localize the object. Many feature extraction methods such as Histogram of Oriented Gradient (HOG) [16] and Scale-Invariant Feature Transform (SIFT) [17] were applied to improve the performance of the tracking. However, such manual features are sensitive to illumination changes, occlusions, and complex backgrounds, and current methods based on deep CNN and Siamese Network [18]–[20] alleviate this problem. Generally, Siamese Network has two branches for the template and the search area with shared weights to measure their similarity in an implicitly embedding space. Afterward, [21] unites the region proposal network and Siamese Network to improve both speed and accuracy. However, the above methods driven by 2D CNN are inapplicable to point clouds. Therefore, MLCFNet extends the siamese object tracking paradigm to the 3D field using an effective 3D object proposal.

B. 3D Object Detection in Point Clouds

Compared to 2D visual tracking, 3D object detection provides plenty of ideas to promote the development of 3D objecting tracking. Various methods, especially PointNet [22], PointNet++ [23], and VoxelNet [24] were proposed to gain discriminative features from sparse 3D point clouds. To address disorder, sparsity, and rotation variance in point clouds, [22] and [23] utilized Max-Pooling to obtain the most critical feature vector in all the point clouds. [24] grouped points into voxels and used 3D CNN that learned features of voxels to generate 3D boxes. Afterward, PointRCNN [25] extended 2D RCNN [26] to 3D point clouds and built 3D RPN to localize 3D bounding boxes, and it frequently appears in 3D object detection and tracking. Then Qi et al. got inspiration from Hough Voting [27] and proposed VoteNet [28], which utilized key seeds to acquire the center of proposals and predict the size of the 3D bounding box. All these methods effectively promote the developments of 3D object detection and provide new ideas for 3D object tracking. In this paper, MLCFNet makes use of PointNet++ [23] as the siamese tracking paradigm and obtains object proposals with VoteNet [28], which simplifies our method and improve the tracking efficiency.

C. 3D Object Tracking in Point Clouds

Different from 2D bounding boxes in images, the 3D object tracking methods localize objects in the 3D world using the geometry contained in 3D bounding boxes. Some early works tackled the 3D tracking problem using the projection of LIDAR point clouds, such as the bird's-eye views [29],or foreground images [30], which input multiple images to the state-of-the-art 2D object tracking networks to generate object proposals. However, the above tracking methods lost fine-grained shape information by projecting point clouds in bird's-eye view or failed if visual features degraded. Since the above issue limited some real-world applications, SC3D [31] and P2B [7] addresses the above concerns from a pure geometric perspective. SC3D is the first work that applies the 3D Siamese tracker to point clouds rather than images, and P2B is a novel point-to-box network for 3D object tracking, which can be end-to-end trained. Both achieved state-of-the-art results on 3D object tracking with only point clouds. However, they are

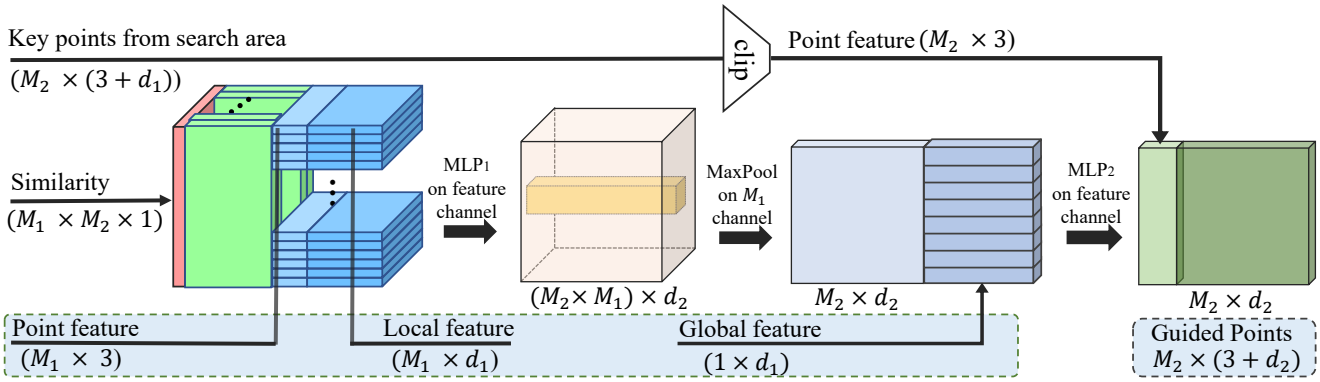

Fig. 3. Feature Fusion Module. Key points from the search area, point features, local features and the similarities are fused to obtain the output features. The global features are concatenated with the output features by the extension operation to generate guided points.

not taking full advantage of multi-level features extracted from Feature Extraction Network. In this paper, we propose a new strategy of using the multi-level context of the template to guide 3D object tracking in search areas, which demonstrates the advantages in the field of 3D object tracking.

III. Methodology

In 3D object tracking, MLCFNet tracks the object (defined by a template) in the search area frame by frame. In particular, to localize the potential object, multi-level context (point features, local features, and global features) of the template are fused into the search area to instruct the network.

A. Overview

As shown in Fig. 2, MLCFNet contains three main parts: 3D Siamese Network, Context Fusion Network, and Vote-Proposal Net. We feed the template and the search area into a shared backbone and obtain their local features, respectively. In the fusion module, we calculate the similarity and use multi-level features from the template instruct the search area to output the guided points. After that, we project these guided points from the search area to potential object centers via Voting Module. And each potential object center clusters its neighbors for a 3D object proposal. Finally, we define the object proposal with the highest score as the final bounding box.

B. 3D Siamese Network

To achieve excellent tracking performance, we need to extract stable and distinctive features from 3D point clouds. PointNet++ extracts local features effectively using a multi-level feature extraction structure. Therefore, our backbones consist of the shared PointNet++ (but not restricted to it) as our Siamese Network (see Fig. 2. a). The input of one backbone is the point clouds from the template, and the other is the point clouds in the search area. For a frame at time t, the template contains the ground truth of the object in the first frame and the predicted results in $t-1$ frames. The search area in the current frame is determined by previous results.

Feature Encoding on Point Clouds. Points in the template P_t (of size N_1) and those in the search area P_s (of size N_2) are inputs of the shared backbone to output key points (M_1 templates $T = \{t_i\}_{i=1}^{M_1}$ and M_2 search areas $S = \{s_i\}_{i=1}^{M_2}$), both are represented with local features ($f_l \in \mathbb{R}^{d_1}$). Due to the hierarchical feature learning architecture in PointNet++ [23], the key points T and S preserve local contexts within P_t and P_s. Meanwhile, according to indices of the key points, we obtain 3D coordinates of the raw point clouds as point features ($f_p \in \mathbb{R}^3$). Every key point is finally represented as $[f_p; f_l] \in \mathbb{R}^{3+d_1}$, in which f_p denotes the 3D position and f_l means the local feature.

C. Context Fusion Network

Context Fusion is a crucial part of our object tracking framework. Compared with the conventional matching method between the template and the search area, the proposed module uses the template features (point features, local features, and global features) to generate guided points (see Fig. 2. b). Context Fusion Network has two main parts: Cosine Similarity Module and Feature Fusion Module. Cosine Similarity Module calculates the similarity between local features of the template and those of the search area, then Context Fusion Network combines the similarity and all the features from the template, the search area. MaxPool operated on the local features of the template generates the global features.

Cosine Similarity Module. To find the relationship between the template and the search area, a natural approach is to compute the feature similarity Sim (of size $M_2 \times M_1$) between T (of size M_1) and S (of size M_2), $e.g.$, using cosine distance:

$$\mathrm{Sim}_{j,i} = \frac{f_{t_i}^{\mathrm{T}} \cdot f_{s_j}}{\|f_{t_i}\|_2 \cdot \|f_{s_j}\|_2}, \forall t_i \in T, s_j \in S \tag{1}$$

where f_{t_i}, f_{s_j} denote the local feature from the search area and that from the template. Note that the size of $Sim_{j,i}$ is $M_2 \times M_1$. $Sim_{j,:}$ means the similarity between s_j and all key points from template.

Feature Fusion Module. To make the template information instruct the search area, we design Feature Fusion Module (FFM) that adds the features from the template to the search area and satisfy the permutation invariance. Our FFM fuses

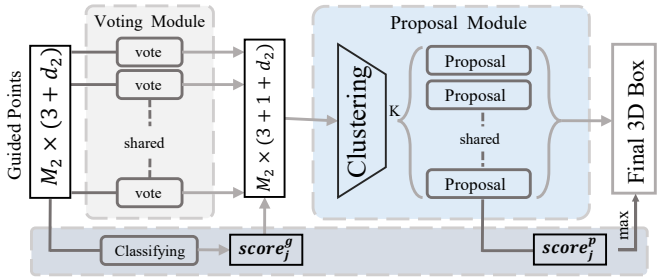

Fig. 4. Vote-Proposal Net that consists of Voting Module and Proposal Module. The guided points are the input of Voting Module and MLP to obtain centers of the proposal and scores of the guided points, respectively. Then the centers with the above scores are fed into Proposal Module to get proposals and their scores where the highest score corresponds to the final 3D box.

four features: the similarity, point features, local features, and global features, as shown in Fig. 3.

To ensure the dimensional consistency, we augment each $Sim_{:,i}$ (the similarity) with key points from the search area S and then add point features and local features from the template T to each $Sim_{j,:}$, yielding a tensor of size $M_2 \times M_1 \times (1+3+d_1+3+d_1)$. Then we feed this tensor into the MLP_1 to fuse the point and local features from the template to the search area and obtain new feature points of size $M_2 \times M_1 \times d_2$. By MaxPool on M_1 channels, we ensure the permutation invariance and acquire the candidate points (of size $M_2 \times d_2$), which combine point and local features from the template.

To make candidate points attached with global features, the local features from the template are fed into MaxPool on M_1 channels to generate the global features (of size $1 \times d_1$). Then candidate points with global features are sent to MLP_2 and obtain guided features $f_{g_j} \in \mathbb{R}^{d_2}$. And we clip the coordinates of s_j ($f_p^{s_j} \in \mathbb{R}^3$) as the point feature x_{g_j} and finally represent the guided point g_j (of size $M_2 \times (3+d_2)$) as $[x_{g_j}; f_{g_j}] \in \mathbb{R}^{3+d_2}$.

Besides, there are other selections for context fusion: leaving out context fusion, leaving out the local features of T, or leaving out the global features of T. All of them are inferior, as detailed in the ablation study.

D. Vote-Proposal Net

Traditionally, embedded with template features, each guided point can be treated as a center to directly predict one proposal. However, the point clouds generated by LIDAR are distributed on the surface of the object, and generating object proposals from surface points will cause large errors. We follow the idea in VoteNet [28] to regress guided points into potential object centers via Hough voting in Voting Module and cluster neighboring centers to leverage the ensemble power and obtain object proposals in the Proposal Module, as shown in Fig. 2. c and Fig. 4.

Voting Module. Each guided point g_j with feature $[x_{g_j}; f_{g_j}] \in \mathbb{R}^{3+d_2}$ can roughly predict a potential object center via Voting Module. Following VoteNet [28], in Voting Module, we apply a Multi-Layer Perceptron (MLP) with fully connected layers, ReLU, and batch normalization to predict the Euclidean space offset $\Delta x_{g_j} \in \mathbb{R}^3$ from g_j to the ground truth object center and the feature offset $\Delta f_{g_j} \in \mathbb{R}_2^d$ from f_{g_j}. We represent the potential center c_j using the feature $[x_{c_j}; f_{c_j}] \in \mathbb{R}^{3+d_2}$. As a result, $x_{c_j} = x_{g_j} + \Delta x_{g_j}$ and $f_{c_j} = f_{g_j} + \Delta f_{g_j}$.

The predicted 3D offset $\Delta x_{g_j} \in \mathbb{R}^3$ is explicitly supervised with a regression loss.

$$L_{\text{reg}} = \frac{1}{G_{\text{obj}}} \sum_i \left\| \Delta x_{g_i} - \Delta x_{g_i}^* \right\| \cdot \mathbb{I}\left[g_i \text{ on object }\right] \quad (2)$$

where $\mathbb{I}\left[g_i \text{ on object }\right]$ indicates whether the guided point g_i is on the surface of the ground truth object, and G_{obj} is the total number of guided points on the object surface. $\Delta x_{g_i}^*$ is the ground truth displacement from the guided point position x_{g_i} to the bounding box center of the object. Note that we only train these guided points located on the surface of the ground truth object.

Proposal Module. Vote Module creates canonical centers C for an object. There are many center candidates, so we firstly cluster neighboring centers to leverage the ensemble power and obtain object proposals. Then we sample a subset of K clusters using farthest point sampling based on c_j in 3D Euclidean space to get c_k with $k = 1, ..., K$. For each c_k, we use ball query [23] to generate the cluster H_j with the radius d in 3D space: $H_k = \left\{c_k \mid \|c_k - c_j\|_2 < d\right\}$, where $k = 1, ..., K$. It is easy to integrate this clustering technique into an end-to-end pipeline, and this module works well in practice.

After clustering, we feed each H_k into the a PointNet-like module (MLP-MaxPool-MLP) and obtain the object proposal $p_k = [box_k^p; score_k^p]$.

$$[box_k^p; score_k^p] = \text{MLP}\left\{\max_{i=1,...,n} \left\{\text{MLP}\left(H_k\right)\right\}\right\} \quad (3)$$

where box_j^p has four parameters: offsets for the 3D position (x, y, z) and rotation in the X-Y plane, and $score_k^p$ denotes the proposal-wise object score.

In K proposals generated from Vote-Proposal Net, one with the highest score is the final tracking result.

Improved Proposal with Voting Score. In the search area, some guided points are not on the surface of the object and affect the final object proposal negatively. Therefore, we follow the idea from P2B [7], which assesses the guided point g_j with its object score to estimate its location and strengthen the learning of a multi-level context fusion network.

Parallel to Voting Module, we feed the guided feature in $\{g_j\}_{j=1}^n$ into the MLP to generate the score $score_j^g$ for each guided point g_j. We regard these guided points located on the surface of the ground truth object as positives and the extra as negatives. In the standard binary cross-entropy loss L_{cls}, $score_j^g$ is related to the location of g_j. Except for the location, L_{cls} can explicitly constrain the multi-level context fusion learning to guide object tracking in the search area.

Inheriting from $score_j^g$, we update the representation of the clustering center c_k with $[x_{c_k}; score_{c_k}^g; f_{c_k}] \in \mathbb{R}^{(3+1+d_2)}$. Sequentially, we update clusters with ball query and object

TABLE I
EXTENSIVE COMPARISONS WITH SC3D AND P2B ON **KITTI** DATASET.

	Method	**Car**	**Pedestrian**	**Van**	**Cyclist**	**Mean**
	Frame Number	6424	6088	1248	308	14068
Success	SC3D-EX [31]	21.2	8.1	25.2	11.1	15.7
	SC3D-KF [31]	41.3	18.2	40.4	**41.5**	31.2
	P2B [7]	56.2	28.7	40.8	32.1	42.4
	MLCFNet [**ours**]	**58.1**	**30.8**	**42.8**	34.2	**44.3**
Precision	SC3D-EX [31]	24.8	14.2	28.9	14.8	20.4
	SC3D-KF [31]	57.9	37.8	47.0	**70.4**	48.5
	P2B [7]	72.8	49.6	48.4	44.7	60.0
	MLCFNet [**ours**]	**74.4**	**51.3**	**49.9**	46.2	**61.5**

TABLE II
ADDITIONAL EXPERIMENT WITH SC3D AND P2B ON **NUSCENES** DATASET.

	Method	**Car**	**Pedestrian**	**Truck**	**Bicycle**	**Overall**
Success	SC3D-EX [31]	14.5	7.4	24.1	13.6	16.4
	SC3D-KF [31]	22.3	11.3	30.7	**35.4**	20,7
	P2B [7]	38.8	28.4	45.3	32.8	36.5
	MLCFNet [**ours**]	**39.4**	**30.2**	**47.0**	34.5	**39.3**
Precision	SC3D-EX [31]	12.3	7.9	15.2	15.3	14.1
	SC3D-KF [31]	21.9	12.7	27.7	**34.7**	20.2
	P2B [7]	43.2	52.2	41.6	32.5	45.1
	MLCFNet [**ours**]	**45.1**	**53.6**	**42.8**	32.5	**47.3**

TABLE III
COMPREHENSIVE COMPARISONS WITH SC3D AND P2B.

	Method	**P-Result**	**P-GT**	**C-GT**
Success	SC3D [31]	41.3	64.6	76.9
	P2B [7]	56.2	82.4	84.0
	MLCFNet [**ours**]	**58.1**	**84.7**	**86.4**
Precision	SC3D [31]	57.9	74.5	81.3
	P2B [7]	72.8	90.1	90.3
	MLCFNet [**ours**]	**74.4**	**92.3**	**92.9**

proposals with Equation (3). Experiments demonstrate that $score^{g}_{c_k}$ can help pick out potential object centers and obtain object proposals with higher quality.

$$L_{total} = L_{\text{reg}} + \lambda_1 L_{\text{cls}} + \lambda_2 L_{\text{obj}} + \lambda_3 L_{\text{box}} \tag{4}$$

where these losses are weighted with $\lambda_1 = 0.2$, $\lambda_2 = 1.5$, and $\lambda_3 = 0.2$.

E. Loss Function

The loss functions in MLCFNet consist of classification loss L_{cls}, regression loss L_{reg}, bounding box estimation loss L_{box}, and objectness loss L_{obj}. We train MLCFNet in an end-to-end manner and represent the total loss function as L_{total}.

We choose the binary cross-entropy loss for classification L_{cls} and objectness L_{obj}. For L_{obj}, the proposals whose centers within 0.3 m of the object center are positive samples, those far away from the object center (by more than 0.6 m) are negative samples, and others are abandoned. Meanwhile, similar to VoteNet [28] and P2B [7], the regression loss L_{reg} and box estimation loss L_{box} are supervised via the Huber (smooth-L1 [32]) loss.

IV. EXPERIMENTS

KITTI and NusSenes tracking datasets [8] in 3D LIDAR is applied for our experiments. Following the same settings of SC3D [31] and P2B [7] in dataset split, object generation, and evaluation for fair comparisons, we mainly focus on car tracking and ablation studies on KITTI. Similar to P2B, we also test MLCFNet in extensive experiments on trackings of other objects (Pedestrian, Van, and Cyclist).

A. Dataset Setting

In KITTI tracking dataset, 21 scenes and 8 types of objects are accessible, and we follow the data split setting in SC3D and P2B: scenes 0-16 for training, 17-18 for validation, and 19-20 for the test. NuScenes dataset contains 1000 scenes and the point clouds are captured by a 32-beam LiDAR. This dataset is split to 700, 150, and 150 scenes for training, validation and testing, respectively. Each instance of a car appearing in each scene is considered as a tracklet for 3D single object tracking. For each tracklet, the ground truth bounding box is given in the first frame, and our task is to locate the object in the following frames.

One Pass Evaluation (OPE) [12] is used as an evaluation metric for single object tracking, which contains *Success* and *Precision*. Specifically, we define IOU between the predicted box and the ground truth one as *Success*, then we estimate *Precision* by AUC about the 3D distance between centers of the above boxes from 0 to 2 m.

B. Comparisons

In MLCFNet, we fuse multiple features from the template to guide 3D object tracking. Similar to SC3D [31] and P2B [7],

TABLE IV
FEATURE FUSION METHODS.

Method	Success	Precision
without feature fusion	54.6	68.9
feature fusion without the local feature	56.6	70.3
feature fusion without the global feature	56.2	72.8
our default setting	**58.1**	**74.4**

TABLE V
TEMPLATE GENERATION METHODS

Method	Success	Precision
the first ground truth	46.9	60.3
the previous result	54.6	70.8
the first ground truth + the previous result	**58.1**	**74.4**
all the previous results	52.8	69.1

our method only use the geometric information from point clouds. Therefore, we mainly compare MLCFNet with the above two methods.

Results for 3D car tracking are listed in TABLE III. In frame T, the search area is output in three ways: the predicted result in frame $T-1$, the ground truth in frame $T-1$, and the ground truth in frame T. However, only using the previously predicted result for the search area meets the requirement of realistic application scenarios. The previous and current ground truths are unreasonable to be used, but they are considered to assess the discriminative power in SC3D by approximate exhaustive search and help approximately assess short-term tracking performances in P2B.

Compared with SC3D and P2B, MLCFNet has better performances in real-time 3D object tracking. In SC3D, the approximate exhaustive search and poor feature extraction limit the speed and accuracy. P2B addresses these weaknesses using VoteNet and greatly improves Success and Precision. Based on P2B, MLCFNet fuses multiple features from the template into the search area. The point and local features instruct the search area to match the template, while global features help points locate the object center.

Extra Comparisons. In addition to Car, results of MLCFNet on Pedestrian, Van, and Cyclist are in TABLE I and II. MLCFNet outperforms both P2B and SC3D. In KITTI dataset, our MLCFNet has a 2.1% improvement in Pedestrian and Van. The mean Success in all frames has improved 1.5%. While in nuScenes dataset, our MLCFNet outperforms P2B and SC3D with about 2% improvement on Precision and 2.5% in Success. Overall, these results demonstrate the advantages of multi-level context fusion for tracking.

C. Ablation Study

In this section, we mainly present the ablation study to highlight the importance of Context Fusion Network.

Importance of Context Fusion. We divide results into four different cases: without context fusion, context fusion without the local feature, context fusion without the global feature, and our default setting, as listed in TABLE IV.

Removing Feature Fusion Module degrades by about 3%, which indicates the importance of context fusion in MLCFNet. After removing the local features or the global features, results show the importance of the global features for *Success* and the local features for *Precision*. We consider that the local features enhance local matching capabilities to improve *Precision*, and the global features promote *Success* by adding global information of the object. In comparison, our default setting brings point, local, and global features from the template to yield a more "directed" object proposal generation.

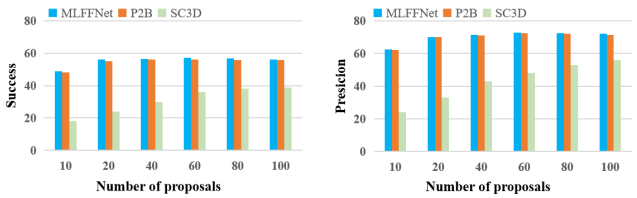

Fig. 5. Experiments on different number of proposals show that our method is compatible with a wide range of parameters.

Robustness on Different Number of Object Proposals. Similar to P2B and SC3D, we also test MLCFNet using the different number of proposals. As shown in Fig. 5, MLCFNet and P2B have stronger stabilities when the number of object proposals degrades dramatically, even when there are only 20 proposals. Without Voting Net, SC3D is sensitive to the number of object proposals using the expensive search strategy. To conclude, MLCFNet can generate object proposals of higher quality, which is crucial to improve efficiency for 3D object tracking.

Template Generation Methods. In addition to our method that generates the template within the ground truth in the first frame and the previous predicted result, we also test the generation method in SC3D and P2B, including the previous result, only the first ground truth, and all the previous results. The network using context fusion of the first ground truth and the previous result performs better than other settings, as listed in TABLE V.

V. CONCLUSION

In this paper, we propose a Multi-Level Context Fusion Network (MLCFNet) for 3D object tracking. We fuse the multi-level object context (point, local and global features) into the search area to guide 3D object tracking, and we formulate a method that can be trained end-to-end using Voting Module and Proposal Module. With the multi-level features, MLCFNet can directly operate on the whole search area instead of redundant extracted 3D object boxes.

Extensive experiments on public dataset show that MLCFNet performs better in 3D single object tracking. In the future, we intend to explore how to track smaller objects with sparse points and extend MLCFNet into multiple object tracking, only using the point clouds.

ACKNOWLEDGEMENT

This work is supported by National Key R&D Program of China (2018YFB1306302, 2018YFB1306300, and 2018YFB1306500).

REFERENCES

[1] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3d object detection network for autonomous driving," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 6526–6534.

[2] A. Dai, C. R. Qi, and M. Nießner, "Shape completion using 3d-encoder-predictor cnns and shape synthesis," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 6545–6554.

[3] A. Lc, A. Mb, B. Ls, and A. Mw, "Lidar–camera fusion for road detection using fully convolutional neural networks - sciencedirect," *Robotics and Autonomous Systems*, vol. 111, pp. 125–131, 2019.

[4] H. Dahlkamp, A. Kaehler, D. Stavens, S. Thrun, and G. R. Bradski, "Self-supervised monocular road detection in desert terrain," in *Robotics: Science and Systems Ii, August, University of Pennsylvania, Philadelphia, Pennsylvania, Usa*, 2006.

[5] A. Martinovic, G. Glavas, M. Juribasic, D. Sutic, and Z. Kalafatic, "Real-time detection and recognition of traffic signs," in *Mipro, International Convention*, 2010.

[6] R. Timofte, K. Zimmermann, and L. Van Gool, "Multi-view traffic sign detection, recognition, and 3d localisation," in *2009 Workshop on Applications of Computer Vision (WACV)*, 2009, pp. 1–8.

[7] 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 Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 6328–6337.

[8] 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 (CVPR)*, 2012, pp. 3354–3361.

[9] 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 (CVPR)*, 2020, pp. 11 618–11 628.

[10] M. Luber, L. Spinello, and K. O. Arras, "People tracking in rgb-d data with on-line boosted target models," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011, pp. 3844–3849.

[11] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[12] M. Kristan, J. Matas, A. Leonardis, T. Vojíř, R. Pflugfelder, G. Fernández, G. Nebehay, F. Porikli, and L. Čehovin, "A novel performance evaluation methodology for single-target trackers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 11, pp. 2137–2155, 2016.

[13] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010, pp. 2544–2550.

[14] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, 2015, pp. 583–596.

[15] L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik, and P. H. S. Torr, "Staple: Complementary learners for real-time tracking," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 1401–1409.

[16] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, 2005, pp. 886–893 vol. 1.

[17] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[18] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, "Signature verification using a "siamese" time delay neural network," in *Advances in Neural Information Processing Systems*, 1993.

[19] R. Tao, E. Gavves, and A. W. M. Smeulders, "Siamese instance search for tracking," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 1420–1429.

[20] Z. Zhang and H. Peng, "Deeper and wider siamese networks for real-time visual tracking," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 4586–4595.

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

[22] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 77–85.

[23] 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*, 2017.

[24] Y. Zhou and O. Tuzel, "Voxelnet: End-to-end learning for point cloud based 3d object detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 4490–4499.

[25] S. Shi, X. Wang, and H. Li, "Pointrcnn: 3d object proposal generation and detection from point cloud," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 770–779.

[26] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014, pp. 580–587.

[27] B. Leibe, A. Leonardis, and B. Schiele, "Robust object detection with interleaved categorization and segmentation," *International Journal of Computer Vision*, vol. 77, no. 1-3, pp. 259–289, 2008.

[28] C. R. Qi, O. Litany, K. He, and L. Guibas, "Deep hough voting for 3d object detection in point clouds," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 9276–9285.

[29] W. Luo, B. Yang, and R. Urtasun, "Fast and furious: Real time end-to-end 3d detection, tracking and motion forecasting with a single convolutional net," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 3569–3577.

[30] B. Yang, W. Luo, and R. Urtasun, "Pixor: Real-time 3d object detection from point clouds," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 7652–7660.

[31] S. Giancola, J. Zarzar, and B. Ghanem, "Leveraging shape completion for 3d siamese tracking," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 1359–1368.

[32] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2017.