MFITrack: Multi-Frame Integration Strategy for Enhanced Motion-Centric Single Object Tracking

1st Pochun Chen
School of Electronic and Computer Engineering, Peking University
Shenzhen, China
bjchen@stu.pku.edu.cn

2rd Nan Zhang
School of Electronic and Computer Engineering, Peking University
Shenzhen, China
zhangnan@stu.pku.edu.cn

3nd Guoqing Liu
Minieye Inc.
Shenzhen, China
guoqing@minieye.cc

4th Ge Li✉
School of Electronic and Computer Engineering, Peking University
Shenzhen, China
geli@ece.pku.edu.cn

2024 IEEE International Conference on Multimedia and Expo (ICME) | 979-8-3503-9015-5/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICME57554.2024.10687986

***Abstract*—3D Single Object Tracking (SOT) in LiDAR point clouds is essential for applications like autonomous driving and surveillance, requiring precise tracking for safety and efficiency. Existing methods often face challenges in adapting to changes in target appearance and capturing motion details in complex environments. To overcome these challenges, in this paper, we introduce a novel three-stage tracker employing a multi-frame integration strategy, dubbed MFITrack. It comprises three stages: multi-frame probabilistic data selection, multi-frame motion prediction and bounding box optimization. MFITrack effectively integrates data from multiple frames, enhancing the extraction of motion information and understanding of the target's trajectory and behavior. Tested on KITTI and Nuscenes datasets, MFITrack demonstrates superior performance over existing methods, showcasing robustness and precision in varied tracking scenarios, particularly in the more complex Nuscenes dataset.**

***Index Terms*—Single Object Tracking, LiDAR Point Clouds, Multi-frame Integration Strategy**

I. Introduction

In the field of computer vision, 3D Single Object Tracking (SOT) has gained prominence in autonomous driving and surveillance [1]–[3]. This is particularly true with the advancements in technologies like LiDAR [4]–[6], where the precise tracking of vehicles and pedestrians is critical.

Siamese network-based methods [7]–[9] have shown significant success in 3D Single Object Tracking. For instance, SC3D [10] compares targets from previous frames with candidate patches in the current frame, which is effective but time-consuming. BAT [11] employs an innovative and efficient box-aware feature fusion module, leveraging the capabilities of BoxCloud to achieve reliable functional matching and embedding. Yet, these methods mainly rely on the target's appearance, neglecting crucial contextual information, which is a drawback in environments with significant target appearance changes and sparse 3D point clouds. Methods [12], [13] that calculate motion information based on two frames, have shown performance improvements. However, these approaches often severely depend on the directly preceding frame, which leads to a lack of subtle motion details, such as inertia, especially during instances of abrupt motion changes. Consequently, they struggle to fully grasp the intricacies of object motion and lack stability in situations where frames are missing.

To deal with these problems, we introduce a novel multi-frame integration strategy, inspired by the Temporal Anti-Aliasing (TAA) [14] algorithm commonly used in computer graphics. We realized it in the design of a three-stage tracker named **MFITrack**. The core of MFITrack lies in its ability to effectively assimilate data across several frames, which can better integrate contextual and motion information to enhance tracking robustness and accuracy in diverse conditions, thereby creating a more comprehensive understanding of the target's trajectory and behavior. The tracker is structured into three distinct stages (as depicted in Fig. 1). In **Stage I**, we dynamically integrate data from different time points, laying the foundation for a context-rich tracking process. **Stage II** is dedicated to the initial extraction and fusion of the object's motion information, setting the stage for precise motion analysis. In **Stage III**, the focus shifts to refining the tracking accuracy through advanced bounding box optimization, enhancing the tracker's responsiveness to changes in object position and shape. In summary, the key contributions of our study include:

- We propose a multi-frame integration strategy to comprehensively capture motion information in 3D Single Object Tracking (SOT).
- We develop MFITrack, an innovative three-stage tracker. Its capacity to efficiently utilize motion data from multiple frames significantly improves tracking consistency and precision.
- Our model is extensively tested on two major datasets, KITTI [15] and Nuscenes [16]. It demonstrates exceptional

✉ Ge Li is the corresponding author (geli@ece.pku.edu.cn). This work was supported in part by Natural Science Foundation of China (No. 62172021), in part by Shenzhen Science and Technology Program (KQTD20180411143338837).

performance, especially on Nuscenes, which has a larger scale and environmental complexity.

II. Related Work

In the field of 3D Single Object Tracking, previous works [10]–[12], [17] primarily fall into two categories: those based on appearance matching, typically employing Siamese networks and those founded on motion paradigms.

A. Siamese Network-based Methods

In the earlier stages of 3D Single-Object Tracking (SOT), most models, exemplified by references [18]–[20], predominantly utilize RGB-D imaging and depth estimation techniques for tracking purposes. Grounded in 2D methodologies, these approaches often exhibit limited tracking performance due to their dependence on image properties. SC3D [10] introduces a Siamese tracker that encodes model and candidate shapes into a compact latent representation, marking the first use of Siamese networks in 3D point cloud tracking. However, this method is significantly time-consuming due to intricate data processing. P2B [17] integrates target information from the template into the search area, establishing an end-to-end approach for point-based target proposal and confirmation, achieving a commendable balance between performance and efficiency that set a benchmark for subsequent studies. MLVSNet [8] utilizes Hough voting [21] on multi-level features to increase vote centers and preserve more useful information, differing from the usual single-level feature voting approach. In a separate development, BAT [11] develops an efficient box-aware feature fusion module, enhancing feature matching and embedding through BoxCloud. PTT [22] integrates the Transformer into the P2B [17] architecture to refine point features, enhancing the precision of feature representation in tracking. LTTR [9] and PTTR [23] introduce the use of distinct correlational operations with 3D transformers to facilitate feature interaction, enhancing the efficacy of the tracking process. GLT-T [1] utilizes global and local self-attention for detailed object encoding, enhancing 3D proposal quality and refining the voting mechanism. While these methods have achieved promising results, they predominantly focus on matching by cropping the target from the previous frame using a given bounding box. This approach often neglects the object's motion tendencies and the contextual information within the data. Consequently, just like the initial limitations observed with RGB-D methods, the changes in object appearance from various angles during motion in 3D tracking lead to inaccuracies in matching targets, ultimately diminishing the effectiveness of the tracking network. Unlike these methods, our method ensures the object's motion information is utilized, enabling more precise tracking.

B. Motion-based Methods

LiDAR point clouds are often incomplete and textureless [24], which poses challenges for efficient appearance matching. Moreover, Siamese network-based methods largely overlooks key motion cues between targets. Addressing this, M2-Track introduces a motion center paradigm that utilizes contextual information. It models motion between two frames by calculating motion vectors between point clouds, enabling predictive capabilities. BEVTrack [13] presents a novel distribution-aware regression strategy for tracking, which constructs the likelihood function with the learned underlying distributions adapted to targets possessing diverse attributes. Nevertheless, the sole reliance of each subsequent frame on its predecessor may lead to incomplete extraction of motion information, reducing tracking precision. Different from these methods, our approach extracts more comprehensive motion information without relying solely on the immediate preceding frame.

III. Method

A. Overview

Building upon the motion paradigm for single-object tracking introduced in M2-Track [12], we introduce MFITrack, designed to provide improved tracking performance through a more comprehensive extraction of the target's motion information. The core objective in single-object tracking is to ascertain a target's position within dynamic scene sequences, especially in point clouds-based environments. The overall architecture of the model is illustrated in Fig. 1. The input to our model includes the scene point clouds at the current timestamp t, along with the historical frames of the scene point clouds and the bounding box of the target point clouds. The output corresponds to the bounding box of the target point clouds at the current timestamp t. In representing point clouds, we use $\mathcal{P}_t \in \mathbb{R}^{N_t \times 3}$, where N_t indicates the number of points in the point clouds at timestamp t, and the 3 channels represent the xyz coordinates of each point. The initial state is given by the first frame's bounding box, parameterized by its central coordinates (x, y, z), orientation (heading angle θ), and dimensions (width, length, height). Considering the limited deformations in non-rigid objects across frames, the size and dimensions of the bounding box are assumed to be constant.

MFITrack consists of three stages: multi-frame probabilistic selecting, multi-frame motion prediction and box optimization. The manifestation of the multi-frame integration strategy across these three stages will be elaborated in Section III-B.

B. Multi-frame Integration Strategy

Our Multi-frame Integration Strategy is sequentially manifested across three stages of MFITrack.

Stage I: Multi-Frame Probabilistic Selection/Sampling. To encode point clouds for spatio-temporal learning modules for tracking tasks, we adopt a method akin to M2Track [12], extending the point clouds into a 5D format. We have introduced two dimensions, t and s_i, to represent time and confidence score respectively. s_i serves to guide the position of the target in the point clouds of a given frame at the current time t. The s_i values are determined by the following formula:

$$s_i = \begin{cases} 0.5 & \text{if } p_i \text{ in } \mathcal{P}_t \\ 0.2/k & \text{if } p_i \text{ in } \mathcal{P}_{t-k} \text{ and } p_i \text{ not in } \mathcal{B}_{t-k} \\ 1 - 0.2 \times k & \text{if } p_i \text{ in } \mathcal{P}_{t-k} \text{ and } p_i \text{ in } \mathcal{B}_{t-k} \end{cases} \quad (1)$$

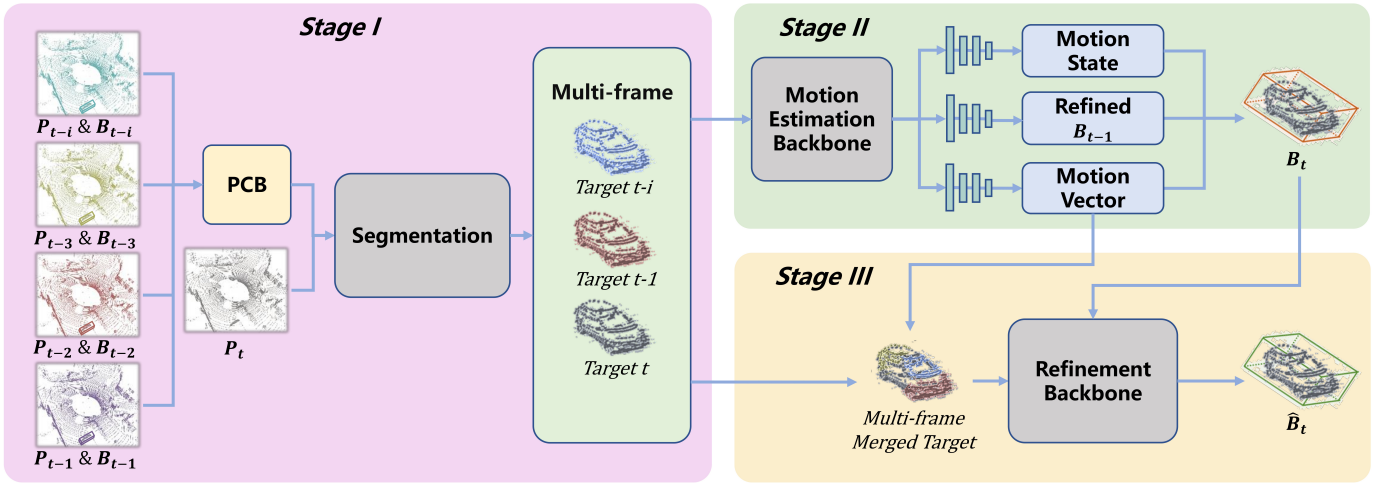

Fig. 1: **This is the overall architecture of MFITrack.** Given a series of consecutive point clouds and potential target bounding boxes (BBoxes), the process begins at **Stage I**, where point clouds are extended to 5D with the introduction of time and confidence score dimensions $\mathcal{P}_t = \{p_i = (x_i, y_i, z_i, t, s_i)\}_{i=1}^{N_t}$. These are then stored in the Point Clouds Bank (PCB). Frames are then selectively retrieved using a specific probabilistic strategy, segmenting targets at different timestamps. Subsequently, at **Stage II**, random sampling of different frames is conducted to compute and merge motion vectors, corresponding to the computation outlined in Eq. 2 and Eq. 3. This stage also produces a refined BBox $\mathcal{B}_{t-1}$ for the preceding timestamp and generates a Motion State, indicating whether the object is currently in motion. Then a rigid body transformation is applied to estimate the target $\mathcal{B}_t$ at timestamp t. Finally, in **Stage III**, a shape completion strategy is employed to regress and optimize the final bounding box $\hat{\mathcal{B}}_t$, which improves the tracking accuracy.

In this formulation, $\mathcal{P}_{t-k}$ and $\mathcal{B}_{t-k}$ respectively represent the point clouds and bounding box at the timestamp $t-k$. Consequently, at any given timestamp t, the point clouds are formulated as $\mathcal{P}_t = \{p_i = (x_i, y_i, z_i, t, s_i)\}_{i=1}^{N_t}$. It is crucial to recognize that in this formula, the s_i value calculation is based on $\mathcal{B}_{t-k}$, which originates from the tracking process's predicted outcomes at the timestamp $t-k$. This implies that each frame's mask inherently contains a level of uncertainty. In selecting the number k of historical frames, we opt for 5 frames as the historical input for our model. To effectively store and process these frames, a Point Clouds Bank (PCB) is established. When accessing the PCB for data retrieval, the probability of selecting a specific frame is modulated by its temporal closeness to the current frame, thereby prioritizing more recent frames. Subsequent to the retrieval of multiple frames from the PCB, these point clouds are segmented to distinguish targets from the scene. This process results in the creation of concatenated multi-frame target point clouds, which are then fed into Stage II for motion prediction and Stage III for further optimization.

Stage II: Multi-frame Motion Prediction. Upon acquiring the concatenated multi-frame point clouds, we progress to Stage II focused on computing motion vectors to predict preliminary bounding box. In this stage, pairwise motion vectors $\mathcal{M}_i \in \mathbb{R}^4$ between frames are calculated, encompassing spatial displacement $(\Delta x, \Delta y, \Delta z)$ and rotation angle $\Delta\theta$. We can formulate this calculation as a function $\mathcal{F}_1$:

$$\begin{aligned} &\mathcal{F}_1 : \mathbb{R}^{N_t \times C} \times (\mathbb{R}^{N_{t-i} \times C} \times \mathbb{R}^7)^k \mapsto \mathbb{R}^4, \\ &\mathcal{F}_1(\mathcal{P}_t, (\mathcal{P}_{t-1}, \mathcal{B}_{t-1})), \ldots, (\mathcal{P}_{t-i}, \mathcal{B}_{t-i})) \mapsto \mathcal{M}_1, \ldots, \mathcal{M}_i. \end{aligned} \tag{2}$$

Subsequently, these vectors are fused to generate the comprehensive motion vector M_t, as delineated in the fusion function $\mathcal{F}_2$. This process creates a unified motion representation at the current timestamp:

$$\mathcal{F}_2(\mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3, ..., \mathcal{M}_i) \mapsto \mathcal{M}_t. \tag{3}$$

After obtaining the motion vectors, as shown in Eq. 4, a **rigid body transformation** is applied to $\mathcal{B}_{t-1}$ to derive $\mathcal{B}_t$ at timestamp t. This step effectively utilizes the computed vectors to project the previous state into the current temporal context.

$$\mathcal{B}_t = \text{Transform}(\mathcal{B}_{t-1}, \mathcal{M}_t). \tag{4}$$

The computational process initiates with the encoding of the selected point clouds frames. We utilize a PointNet [25] backbone to generate an embedding, subsequently processed by three distinct MLPs. These MLPs compute three key outputs: the 6D motion parameters (comprising 2D motion logits and a 4D motion vector) and the 4D optimization vector, which is used to refine $\mathcal{B}_{t-1}$. As mentioned in Eq. 3, the calculation of the final motion vector M_t involves fusing all pairwise motion vectors. The weight assigned to each vector for point clouds extraction from the PCB, favoring vectors nearer to the current timestamp t. Nonetheless, a slight random noise is introduced to each weight to ensure dynamism. Consequently, the final motion vector M_t is derived through a straightforward vector

TABLE I: Comparison of MFITrack against state-of-the-arts on the KITTI Open Dataset. Mean shows the average result weighed by frame numbers. **Bold** denotes the best performance. Success/Precision are used for evaluation.

	Category Frame Number	Car 6424	Pedestrian 6088	Van 1248	Cyclists 308	Mean 14068
Success	SC3D [10]	41.3	18.2	40.4	41.5	31.2
	P2B [17]	56.2	28.7	40.8	32.1	42.4
	3DSiamRPN [7]	58.2	35.2	45.7	36.2	46.7
	LTTR [9]	65.0	33.2	35.8	66.2	48.7
	MLVSNet [8]	56.0	34.1	52.0	34.3	45.7
	BAT [11]	60.5	42.1	52.4	33.7	51.2
	PTT [22]	67.8	44.9	43.6	37.2	55.1
	V2B [28]	**70.5**	48.3	50.1	40.8	58.4
	PTTR [23]	65.2	50.9	52.5	65.1	57.9
	M2-Track [12]	65.5	61.5	53.8	73.2	62.9
	CAT [2]	66.6	51.6	53.1	67.0	58.9
	GLT-T [1]	68.2	52.4	52.6	68.9	60.1
	MFITrack(Ours)	67.6	**62.2**	**54.3**	**73.8**	**64.3**
Precision	SC3D [10]	57.9	37.8	47.0	70.4	48.5
	P2B [17]	72.8	49.6	48.4	44.7	60.0
	3DSiamRPN [7]	76.2	56.2	52.9	49.0	64.9
	LTTR [9]	77.1	56.8	45.6	89.9	65.8
	MLVSNet [8]	74.0	61.1	61.4	44.5	66.7
	BAT [11]	77.7	70.1	67.0	45.4	72.8
	PTT [22]	81.8	72.0	52.5	47.3	74.2
	V2B [28]	81.3	73.5	58.0	49.7	75.2
	PTTR [23]	77.4	81.6	61.8	90.5	78.1
	M2-Track [12]	80.8	88.2	70.7	93.5	83.4
	CAT [2]	81.8	77.7	69.8	90.1	79.1
	GLT-T [1]	**82.1**	78.8	62.9	92.1	79.3
	MFITrack(Ours)	81.9	**88.5**	**71.5**	**93.7**	**81.4**

summation. The role of 2D motion logits is to determine the object's mobility. If stationary, B_t can be directly obtained without computing the Motion Vector. As the 4D optimization vector primarily focuses on optimizing the previous frame's BBox, we do not fuse information from earlier frames.

Stage III: Box Optimization. To address the issue of partial target occlusion and enhance our target representation, we have implemented a shape completion strategy. Initially, we perform random sampling on the segmented targets from multiple frames, extracting portions of the target corresponding to each frame. These segments are then subjected to a rigid body transformation using the motion vectors corresponding to each frame, computed in Stage II, aligning them to the current timestamp t. Subsequently, these transformed segments are directly merged as supplements to the current target, enriching the point clouds of the current frame. We adopt the methodology outlined in [12], [26], [27] to transform the synthesized targets into the coordinate system of the bounding box output $\mathcal{B}_t$ by Stage II. Consequently, a PointNet refinement backbone [25] is used to regress and compute an optimized box $\hat{\mathcal{B}}_t$.

C. Loss Functions

In our framework, models are trained end-to-end by minimizing the following objective function:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{seg} + \lambda_2 \mathcal{L}_{logits} + \lambda_3 \mathcal{L}_{motion} + \lambda_4 \mathcal{L}_{box}, \quad (5)$$

TABLE II: Comparison among our MFITrack and the state-of-the-art methods on the Nuscenes datasets.

	Category Frame Number	Car 64,159	Pedestrian 33,227	Truck 13,587	Trailer 3,352	Bus 2,953	Mean 117,278
Success	SC3D [10]	22.31	11.29	30.67	35.28	29.35	20.70
	P2B [17]	38.81	28.39	42.95	48.96	32.95	36.48
	BAT [11]	40.73	28.83	45.34	52.59	35.44	38.10
	M2-Track [12]	55.85	32.10	57.36	57.61	51.39	49.23
	CAT [2]	43.34	30.68	47.64	57.90	43.30	40.67
	GLT-T [1]	-	-	52.7	57.6	44.6	-
	MFITrack(Ours)	**56.15**	**37.25**	**62.30**	**64.25**	**57.24**	**51.94**
Precision	SC3D [10]	21.93	12.65	27.73	28.12	24.08	20.20
	P2B [17]	43.18	52.24	41.59	40.05	27.41	45.08
	BAT [11]	43.29	53.32	42.58	44.89	28.01	45.71
	M2-Track [12]	65.09	60.92	59.54	58.26	51.44	62.73
	CAT [2]	49.41	56.67	48.10	55.31	41.42	51.28
	GLT-T [1]	-	-	51.4	52.0	40.7	-
	MFITrack(Ours)	**65.30**	**63.55**	**64.87**	**63.39**	**56.06**	**64.67**

where $\mathcal{L}_{seg}$ represents the segmentation task loss in Stage I and $\mathcal{L}_{logits}$ denotes the loss for motion classification in Stage II, both calculated using the standard cross-entropy loss method. $\mathcal{L}_{motion}$ corresponds to the loss incurred during the computation of relative motion vectors, and $\mathcal{L}_{box}$ is the loss between the predicted bounding box and the ground truth. The $\mathcal{L}_{motion}$ component is defined as:

$$\mathcal{L}_{motion} = \sum_{i=1}^{5} \mathcal{L}_{vector\_1}^{(i)} \times \mathcal{P}_i + \mathcal{L}_{vector\_2} + \mathcal{L}_{pre\_refine}. \quad (6)$$

Both $\mathcal{L}_{box}$ and each component of $\mathcal{L}_{motion}$ apply the Huber loss [29] between the predictions and the ground truth. For these calculations, the ground truth vectors are derived from the dimensions of the bounding box. The sum of frame selection probabilities $\mathcal{P}_i$ from the PCB equals one, integrating these probabilities into the loss calculation to reflect each frame's likelihood of inclusion.

IV. Experiments

A. Settings

Datasets. Our MFITrack is extensively tested on two large-scale datasets, KITTI [15] and Nuscenes [16], to evaluate its performance. Comparative analyses are conducted against previous works to demonstrate its efficacy and improvements in tracking accuracy. KITTI [15] comprises a total of 21 training sequences and 29 test sequences. Following the method outlined in [10]–[12], [17], we partitioned its training portion into three segments: training, validation, and testing, with a distribution ratio of 17:2:2. The Nuscenes dataset presents a more challenging scenario, encompassing 1000 driving scenes, which are divided into 700/150/150 scenes for train/val/test and cover 23 object classes. Additionally, the Nuscenes dataset's dense object population makes its tracking more challenging. Following [1], [11], we train our MFITrack on training set, and evaluate the performance on validation set.

Evaluation Metrics. Adopting the approach in [1], [2], [11], [12], [17], we employ the One Pass Evaluation (OPE) [30]

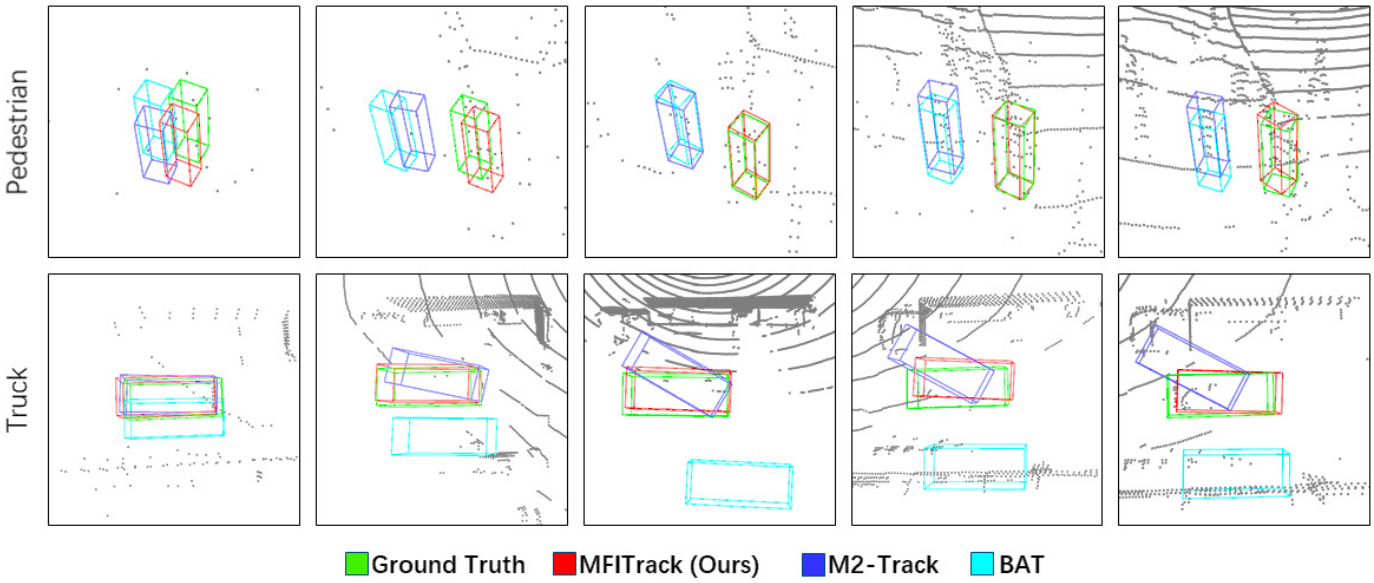

Fig. 2: **Visualization Results on Nuscenes.** The sequence is displayed from left to right following the order of time.

TABLE III: Ablation study for the Truck category on the Nuscenes dataset, testing the key components in **Stage I&II** and **Stage III** of MFITrack and the application of the multi-frame integration strategy to BAT [11]. The green segment represents the full performance of MFITrack.

Multi-frame Sampling&Fusion	Multi-frame Completion	Baseline	Success	Precision
✓		MFITrack	60.57 ↓ 1.73	62.52 ↓ 2.35
✓	✓	MFITrack	60.17 ↓ 2.13	63.17 ↓ 1.70
✓	✓	MFITrack	62.30	64.87
		BAT	45.34	42.58
✓		BAT	50.37 ↑ 5.03	47.17 ↑ 4.59

for measuring *Success* and *Precision* metrics of trackers. This method considers overlap as the IoU between predicted and ground truth BBoxes and defines error as the distance between their centers. We calculate Success as the area under the curve (AUC) for IoU thresholds (0 to 1), and Precision as the AUC for center distance thresholds (up to 2 meters).

Implementation Details. Experiments are performed on NVIDIA 3090 GPUs. The MFITrack model is trained for 180 epochs on KITTI and 60 epochs on the larger Nuscenes dataset to balance computational demands.

B. Comparison with Existing Methods

Here, we present the results of MFITrack conducted on KITTI and Nuscenes, comparing them with previous methods.

Results on KITTI. From the Tab. I, it is evident that MFITrack surpasses all previous methods in terms of performance on four categories, exhibiting the best performance to date. In the car category, our tracking results (67.6/81.9) were slightly inferior to those of V2B (70.5/81.3) and GLT-T (68.2/82.1). However, our method demonstrated a notable improvement over these two methods in other categories, especially in Pedestrian and Van. We believe that the voxel approach used in V2B is particularly well-suited for tracking vehicles like cars that are relatively box-shaped.

Results on Nuscenes. As illustrated in Tab. II, our approach shows more pronounced improvements on the Nuscenes dataset compared to its performance enhancements over other methods on the KITTI dataset. It excels across every category, clearly surpassing previous approaches. This is attributed to the multi-frame integration strategy which is effective against the numerous distractors in Nuscenes. This overall performance suggests that our method is well-suited for complex environments and demonstrates robustness.

C. Ablation Studies

Tab. III showcases the ablation study on Nuscenes, analyzing the impact of multi-frame sampling and motion vector fusion in **Stage I&II** and multi-frame completion in **Stage III**. Not utilizing the integrated vector implies the use of the M2-Track [12] which computes vectors between two frames only. The row highlighted in green indicates MFITrack's full performance. The results indicate a performance drop when either module is removed, yet MFITrack still performs competitively, affirming the robustness of its architecture. We also apply a multi-frame integration strategy to the BAT [11] baseline, using methods from **Stage I&II** to randomly sample and fuse matched results, which significantly enhances performance and demonstrates the efficacy of our multi-frame strategy in Siamese-based approaches.

V. Conclusions

In our work, we revisited the 3D Single Object Tracking (SOT) task in LiDAR point clouds and explored ways to enhance the use of motion information and precision in complex datasets, addressing some of the limitations found in existing methods. To address these challenges, we introduced

MFITrack, a novel tracker employing a multi-frame integration strategy. This strategy effectively integrates data across multiple frames, significantly enhancing the understanding of the target's trajectory and behavior, especially in scenarios with many interfering factors. Tested on the KITTI and Nuscenes datasets, MFITrack demonstrated superior performance, proving its robustness and precision in various tracking scenarios. The success of MFITrack highlights the importance of considering multi-frame data and contextual information in 3D SOT, contributing to the ongoing development in this field and indicating a promising direction for future research in advanced tracking technologies.

REFERENCES

[1] Jiahao Nie, Zhiwei He, Yuxiang Yang, Mingyu Gao, and Jing Zhang, "Glt-t: Global-local transformer voting for 3d single object tracking in point clouds," in *AAAI*, 2023.

[2] Jiantao Gao, Xu Yan, Weibing Zhao, Zhen Lyu, Yinghong Liao, and Chaoda Zheng, "Spatio-temporal contextual learning for single object tracking on point clouds," *IEEE TNNLS*, 2023.

[3] Zitong Yi, Zhihang Tong, Yanyun Zhao, Zhicheng Zhao, and Fei Su, "A method of stable long-term single object tracking," in *IEEE International Conference on Multimedia Expo (ICME)*, 2021.

[4] Kanchan Bahirat and Balakrishnan Prabhakaran, "A study on lidar data forensics," in *IEEE International Conference on Multimedia Expo (ICME)*, 2017.

[5] Xiaolei Chen, Wenlong Liao, Bin Liu, Junchi Yan, and Tao He, "Opendenselane: A dense lidar-based dataset for hd map construction," in *IEEE International Conference on Multimedia Expo (ICME)*, 2022.

[6] Wei Zhang, Youguang Yu, and Fuzheng Yang, "A novel grid-based geometry compression framework for spinning lidar point clouds," in *IEEE International Conference on Multimedia Expo (ICME)*, 2022.

[7] Zheng Fang, Sifan Zhou, Yubo Cui, and Sebastian Scherer, "3d-siamrpn: An end-to-end learning method for real-time 3d single object tracking using raw point cloud," *IEEE Sensors Journal*, 2020.

[8] Zhoutao Wang, Qian Xie, Yu-Kun Lai, Jing Wu, Kun Long, and Jun Wang, "Mlvsnet: Multi-level voting siamese network for 3d visual tracking," in *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[9] Yubo Cui, Zheng Fang, Jiayao Shan, Zuoxu Gu, and Sifan Zhou, "3d object tracking with transformer," *arXiv preprint arXiv:2110.14921*, 2021.

[10] Silvio Giancola, Jesus Zarzar, and Bernard Ghanem, "Leveraging shape completion for 3d siamese tracking," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[11] Chaoda Zheng, Xu Yan, Jiantao Gao, Weibing Zhao, Wei Zhang, Zhen Li, and Shuguang Cui, "Box-aware feature enhancement for single object tracking on point clouds," in *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[12] Chaoda Zheng, Xu Yan, Haiming Zhang, Baoyuan Wang, Shenghui Cheng, Shuguang Cui, and Zhen Li, "Beyond 3d siamese tracking: A motion-centric paradigm for 3d single object tracking in point clouds," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[13] Yuxiang Yang, Yingqi Deng, Jiahao Nie, and Jing Zhang, "Bevtrack: A simple baseline for point cloud tracking in bird's-eye-view," *arXiv preprint arXiv:2309.02185*, 2023.

[14] Jonathan Korein and Norman Badler, "Temporal anti-aliasing in computer generated animation," in *Proceedings of the 10th annual conference on Computer graphics and interactive techniques*, 1983.

[15] Andreas Geiger, Philip Lenz, and Raquel Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[16] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[17] Haozhe Qi, Chen Feng, Zhiguo Cao, Feng Zhao, and Yang Xiao, "P2b: Point-to-box network for 3d object tracking in point clouds," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[18] Alessandro Pieropan, Niklas Bergström, Masatoshi Ishikawa, and Hedvig Kjellström, "Robust 3d tracking of unknown objects," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2015.

[19] Alireza Asvadi, Cristiano Premebida, Paulo Peixoto, and Urbano Nunes, "3d lidar-based static and moving obstacle detection in driving environments: An approach based on voxels and multi-region ground planes," *Robotics and Autonomous Systems*, 2016.

[20] Ugur Kart, Alan Lukezic, Matej Kristan, Joni-Kristian Kamarainen, and Jiri Matas, "Object tracking by reconstruction with view-specific discriminative correlation filters," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[21] Charles R Qi, Or Litany, Kaiming He, and Leonidas J Guibas, "Deep hough voting for 3d object detection in point clouds," in *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[22] Jiayao Shan, Sifan Zhou, Zheng Fang, and Yubo Cui, "Ptt: Point-track-transformer module for 3d single object tracking in point clouds," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021.

[23] Changqing Zhou, Zhipeng Luo, Yueru Luo, Tianrui Liu, Liang Pan, Zhongang Cai, Haiyu Zhao, and Shijian Lu, "Pttr: Relational 3d point cloud object tracking with transformer," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[24] Xinhua Cheng, Nan Zhang, Jiwen Yu, Yinhuai Wang, Ge Li, and Jian Zhang, "Null-space diffusion sampling for zero-shot point cloud completion," in *International Joint Conference on Artificial Intelligence (IJCAI)*, 2023.

[25] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[26] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li, "Pointrcnn: 3d object proposal generation and detection from point cloud," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[27] Shaoshuai Shi, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li, "From points to parts: 3d object detection from point cloud with part-aware and part-aggregation network," *IEEE transactions on pattern analysis and machine intelligence*, 2020.

[28] Le Hui, Lingpeng Wang, Mingmei Cheng, Jin Xie, and Jian Yang, "3d siamese voxel-to-bev tracker for sparse point clouds," *Advances in Neural Information Processing Systems*, 2021.

[29] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in Neural Information Processing Systems*, 2015.

[30] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang, "Online object tracking: A benchmark," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition(CVPR)*, 2013.