Poly-MOT: A Polyhedral Framework For 3D Multi-Object Tracking

Xiaoyu Li[†], Tao Xie[†], Dedong Liu[†], Jinghan Gao, Kun Dai, Zhiqiang Jiang, Lijun Zhao, Ke Wang

Abstract— 3D Multi-object tracking (MOT) empowers mobile robots to accomplish well-informed motion planning and navigation tasks by providing motion trajectories of surrounding objects. However, existing 3D MOT methods typically employ a single similarity metric and physical model to perform data association and state estimation for all objects. With large-scale modern datasets and real scenes, there are a variety of object categories that commonly exhibit distinctive geometric properties and motion patterns. In this way, such distinctions would enable various object categories to behave differently under the same standard, resulting in erroneous matches between trajectories and detections, and jeopardizing the reliability of downstream tasks (navigation, etc.). Towards this end, we propose Poly-MOT, an efficient 3D MOT method based on the Tracking-By-Detection framework that enables the tracker to choose the most appropriate tracking criteria for each object category. Specifically, Poly-MOT leverages different motion models for various object categories to characterize distinct types of motion accurately. We also introduce the constraint of the rigid structure of objects into a specific motion model to accurately describe the highly nonlinear motion of the object. Additionally, we introduce a two-stage data association strategy to ensure that objects can find the optimal similarity metric from three custom metrics for their categories and reduce missing matches. On the NuScenes dataset, our proposed method achieves state-of-the-art performance with 75.4% AMOTA. The code is available at https://github.com/lixiaoyu2000/Poly-MOT.

I. INTRODUCTION

Multi-Object Tracking (MOT) is a critical component of environment perception systems in autonomous robots. It provides valuable information on the motion of tracked objects over time, enabling robots to predict the future motion patterns of surrounding objects effectively. Compared with 2D MOT [1], [2], [25], 3D MOT [3] offers more explicit and convenient spatial information about objects, culminating in more reliable and accurate tracking. Typically, current 3D MOT techniques can be divided into "Tracking-By-Detection" (TBD) [4], [5] and "Joint Detection and Tracking" (JDT) [6]–[8]. Due to the data-driven nature of JDT, it is generally less precise and robust than TBD, and consequently, the majority of 3D MOT approaches adhere to the TBD architecture.

In the most previous works [3], [4], [9], KITTI [10] and MOT15 [11] are employed to evaluate algorithm performance. Under these platforms, trackers are usually required to track only a single category of objects. Therefore, these works simply use a single linear motion model and similarity

(a) State Estimation For *Car*   (b) State Estimation For *Motorcycle*

Fig. 1. **Trajectory state estimation of our proposed (*CTRA* and *Bicycle* Model) and existing (*CA Model*) motion model on *Car* and *Motorcycle*.** For trackers equipped with different motion models, we truncate the tracking process from the same timestamp, which means that the trajectory can receive detection updates before this timestamp. In contrast, the trajectory can only use historical information to predict the future state after this timestamp. (a) *CTRA Model* exhibits a significantly higher prediction accuracy for *Car* than other models. This is particularly useful for recovering historical mismatch trajectories when objects are occluded, or detectors miss detections. (b) *Bicycle Model* is more suitable for *Motorcycle* due to different object categories exhibiting distinct motion patterns.

metric for state prediction and construct the cost matrix between trajectories and detections. However, with the advent of large-scale datasets such as NuScenes [12] and changeable real scenes, a long-ignored yet fundamental fact must be carefully considered: *there are multiple object categories in real scenes, and objects of different categories often exhibit various geometric features and motion patterns*. A single prediction and matching criterion is unsuitable for distinct object categories, which distorts the affinities between trajectories and detections, resulting in false matches and compromising the stability of subsequent tasks (navigation, prediction, etc.).

To the best of our knowledge, only a few recent works [4], [5] have optimized the MOT problem in multi-category settings. These methods prevent correlation between different categories by masking [5] or removing [4] invalid costs in the cost matrix calculated under the same standard. However, these methods can not tackle the issue of accurate tracking in multi-category settings fundamentally due to the *inaccuracy of the cost matrix* induced by *unreliable prediction* and *irrationality metric*. On the one hand[1], as shown in Fig. 1, due to the distinct and nonlinear motion patterns of different object categories, utilizing the same linear motion model for state prediction will result in an unreliable estimation of motion. Moreover, due to variances in geometric features,

[1]In Fig. 1, *CA* denotes *Constant Acceleration*, *CTRA* denotes *Constant Turn Rate and Acceleration*

TABLE I

Category	Similarity Metric	AMOTA↑	IDS↓
Ped	$gIoU_{3d}$	81.7	175
	$gIoU_{bev}$	81.2	203
	d_{eucl}	81.0	220
Bus	$gIoU_{3d}$	88.1	2
	$gIoU_{bev}$	88.2	2
	d_{eucl}	87.5	1

different object categories are susceptible to various similarity metrics and correlation thresholds. As presented in Table I, we conduct a simple and intuitive experiment confirming that a single similarity metric cannot perform well in all object categories. Thus, precise yet reliable motion prediction and affinity calculation for various object categories is a vital step toward deploying 3D MOT methods in real scenes.

To this end, we introduce Poly-MOT, a polyhedral framework for 3D MOT under multi object category scenes following the TBD framework. Specifically, to ensure accurate motion prediction in such scenes, we introduce geometry constraints to the motion model and establish multiple motion models (*CTRA* and *Bicycle* model) based on the distinct features of each object category. For accurate object matching, we design three similarity metrics and then introduce categorical data association, in which the tracker selects the optimal similarity metric for different categories to achieve accurate affinity calculation. We also employ a technique that combines Non-Maximum Suppression (NMS) and Score Filter (SF) to preprocess detections at each frame to eliminate the gap between detection task and tracking task. Finally, we additionally employ a combined count-based and confidence-based strategy so that Poly-MOT can handle the lifecycle of trajectories with various matching statuses.

Poly-MOT is learning-free and not data-driven, using only detection results as input and achieving state-of-the-art performance and manageable real-time performance without substantial computational resources, as shown in Tables II and III. Thanks to the TBD framework, Poly-MOT achieves stable tracking performance with multiple 3D detectors (CenterPoint [8], etc.). **With 75.4% AMOTA, our technique achieves state-of-the-art performance on the NuScenes test set.** We anticipate that Poly-MOT can provide an effective 3D MOT baseline algorithm for the community. The primary contributions of this work are as follows:

- We propose Poly-MOT, an efficient 3D MOT approach for multiple object category scenes based on the TBD framework.
- We introduce geometry constraints to the motion model and establish multiple motion models (*CTRA* and *Bicycle* model) according to the distinct features of different object categories, enabling capture motion pattern differences between categories.
- We design three custom similarity metrics and a novel two-stage data association strategy to ensure that various objects can identify the optimal similarity metric for their categories, thus reducing missing matches.

II. RELATED WORK

3D Multi-Object Tracking. Weng [3] pioneers the application of the TBD framework to the 3D MOT method, using Linear Kalman Filter and 3D IOU to build an advance and fast 3D MOT system. The TBD framework divides the tracker into four steps: (1) Receiving and preprocessing the 3D detection, (2) Predicting motion for active trajectories, (3) Correlating and matching trajectory with detection, (4) Managing the lifecycle of all state trajectories. Simple-Track [9] applies simple techniques to analyze and improve each of these four parts, resulting in impressive tracking performance. EagerMOT [4] takes the lead in employing result-level fusion to integrate 2D and 3D detections, improving the robustness of tracker to false negatives from different sensor modalities. In addition to TBD, the JDT framework processes tracking and detection tasks in a single Neural Network(NN). Feature alignment between multiple modalities is an important yet difficult point of JDT.

Data Association in 3D MOT. Data association is the core of MOT, as it is accomplished by calculating a cost matrix between trajectories and detections with a similarity metric and then applying a matching algorithm to obtain the final associations. Geometry-based and appearance-based are two common types of similarity metrics. The former leverages location and motion information to boost the performance under occlusion, and common metrics include 3D IOU [3], 3D GIOU [5], [9]. Appearance-based metrics, which utilize appearance information, can achieve more robust results in cases of large distance movement or low frame rate, as demonstrated in several studies [5], [7], [13]. Multi-modal 3D MOT methods typically use multi-level correlation [4], [5] (applying multiple metrics to match objects multiple times) to fuse different modalities and improve performance. Poly-MOT demonstrates the benefits of multi-level correlation in reducing FN matches in LiDAR-only methods. Hungarian algorithm [3], [5] and greedy algorithm [8] are commonly used to solve the cost matrix. A concern is that existing methods use a single similarity metric for all object categories, despite the differences in geometric and appearance features among them. In contrast, Poly-MOT enables the tracker to select the optimal metric for each category based on its characteristics.

Motion module in 3D MOT. The motion module predicts the state of active trajectories, maintaining temporal consistency with detection. Motion prediction techniques can be divided into learning-based and filter-based methods. The former usually uses NN to predict the inter-frame displacement. CenterPoint [8] uses a center-based detector to output 3D detections and predicts the displacement of objects between frames by adding a regression branch. Filter-based methods use real-world physical models for state transitions, exhibit better robustness and real-time performance, and are widely adopted by most methods. Kalman Filter is a widely used method. Most Filter-based methods typically use *CA* [5] or *Constant Velocity (CV)* [3], [8], [9] model as the motion model. However, these models assume that the movements

Fig. 2. **The Pipeline Of Our Proposed Method At Frame** t. (I) Previous active trajectory T_{t-1} is divided into T_{t-1}^{CTRA} and T_{t-1}^{BIC} according to the different motion patterns. State predictions for T_{t-1}^{CTRA} and T_{t-1}^{BIC} are then made based on the distinct and nonlinear motion models using the EKF. (II) Raw detections output by 3D detector are subjected to NMS and SF to reduce false positives to obtain D_t. (III) The prediction states $T_{t,t-1}^{CTRA}$, $T_{t,t-1}^{BIC}$, and D_t are input to the Class Filter to classify the category. The first association is implemented within each category using the optimal similarity metric and a category-specific threshold. For unmatched trajectories T_{t-1}^{1u} and unmatched detections D_t^{1u}, the second association is implemented with a distinct metric than before and a strict threshold. The final matched pairs DT_t^m are used to update the corresponding trajectory. (IV) D_t^u are initialized as new active trajectories. Part of T_{t-1}^u are discarded based on the *count-based* strategy, while others are added to the active trajectories again after the confidence score decays. Still active trajectories will be output to the result file. Eventually, all active trajectories T_t will be passed to the next frame $t + 1$.

of objects on each coordinate axis are independent, ignoring nonlinear motion patterns constrained by geometry and differences in motion patterns across categories. Therefore, to ensure accurate prediction in multi-category scenes, we introduce geometry constraints and establish multiple models based on distinct features of each category.

III. METHOD

Poly-MOT can be divided into four parts: the pre-processing module, multi-category trajectory motion module, multi-category data association module, and trajectory management module, as shown in Fig. 2.

A. 3D Detector and Pre-processing Module

Existing 3D detectors [8], [19], [27] generate numerous low-confidence bounding boxes to ensure high recall, but applying these detections directly to update trajectories can result in severe ID switches (IDS). To tackle this issue, raw detections D_t' must be preprocessed to reduce false-positive matches. We apply Non-Maximum Suppression (NMS) to D_t' at each frame to remove bboxes with high similarity, improving precision without significant loss of recall. Nevertheless, each frame of the large-scale dataset (Waymo [14], NuScenes [12], etc.) and real scenes usually contains a large number of objects while the number of D_t' is significant. Directly applying NMS to D_t' would lead to substantial computational overhead, as illustrated in Table V. Before NMS, we apply a filtering process called Score Filter (SF) to remove detections D_t' with confidence scores less than θ_{SF}. SF can efficiently remove apparent false-positive detections, improving the inference speed of the algorithm. After preprocessing, we obtain D_t, which includes the center of geometry position (x, y, z), 3D size (width, length, height) (w, l, h), heading angle θ, and velocity (v_x, v_y) on the ground plane. Note that whether velocity information is included or not depends on the dataset.

B. Multi-Category Trajectory Motion Module

Most previous methods [5], [9] employ a uniform *CA* or *CV* model to predict the trajectories of all objects, whereas they fail to capture the highly nonlinear motion features of objects and ignore the differences in motion patterns across categories. To address this issue, we propose Multi-Category Trajectory Motion Module that utilizes different motion models (*CTRA Model* and *Bicycle Model*) for various object categories to characterize distinct types of motion accurately. In addition, we also introduce the constraint of the rigid structure of objects into a specific model to accurately describe the highly nonlinear motion of the object. Notably, our motion models are formulated in the East(x)- North(y)-Up(z) coordinate system, which follows the right-hand rule.

CTRA Model. For the *CTRA model*, the turn rate ω and acceleration a of the object are considered constant. As shown in Fig. 3 (a), the heading angle and motion pattern of objects are tightly coupled in the *CTRA model*, which means the directions of the heading angle θ, velocity v, and acceleration a of the object are on the same straight line. *CTRA model* is suitable for *car-like* objects and $pedestrian$. We formulate the state of an object trajectory as a 10-dimensional vector $T^{CTRA} = [x, y, z, v, a, \theta, \omega, w, l, h]$ in the *CTRA model*, where (x, y, z) represent the location of the geometric center of objects in the 3D space, (w, l, h) represent the 3D size of objects.

Bicycle Model. For the *Bicycle model*, it maintains the rigid structure of objects and enables the velocity direction and heading angle of objects to vary, rendering it suitable for objects that behave like bicycles, as illustrated in Fig. 3 (b). Meanwhile, we assume that the steering angle and velocity of the object remain constant. The state of the trajectory is also represented by a 10-dimensional vector $T^{BIC} = [x', y', z, v, a, \theta, \delta, w, l, h]$, where (x', y') represents the location of the gravity center of the object on the ground, δ represents the steering angle of the object, the

remaining variables have the same meaning as the variables in T^{CTRA}.

Model Establishment and State Prediction. Due to the nonlinear property of the motion models, we leverage the Extended Kalman Filter (EKF) to estimate the trajectory state. The prediction process can be described by:

$$T_{t,t-1} = f(T_{t-1}), \ P_{t,t-1} = F_t P_{t-1} F_t^T + Q, \quad (1)$$

where T_{t-1} denotes T^{CTRA} or T^{BIC}, depending on the motion model of objects. P_{t-1} is the covariance matrix at the previous moment $t-1$. $T_{t,t-1}$ is the predict state of T_{t-1} at the current moment t. Q is the process noise, which has an artificially set value. $f(\cdot)$ is the state transition function that is established from the motion model, reflecting the changes of all state variables of the trajectory between two consecutive frames. F_t is the Jacobian matrix obtained through the partial derivative of $f(\cdot)$ with respect to T_{t-1}.

During the state transition of all motion models, the variables a, z, w, l, h are assumed to remain constant.

The object location transition process as components of $f(\cdot)$ can be formulated as:

$$\hat{x}_{t,t-1} = \hat{x}_{t-1} + \int_{(t-1)\sigma}^{t\sigma} v(\tau)cos(\eta(\tau))d\tau, \quad (2)$$

$$\hat{y}_{t,t-1} = \hat{y}_{t-1} + \int_{(t-1)\sigma}^{t\sigma} v(\tau)sin(\eta(\tau))d\tau, \quad (3)$$

where σ is the interval between two adjacent frames of the LiDAR scan. Depending on the choice of motion model, the geometric center (x, y) or gravity center (x', y') of the object can be represented uniformly by $(\hat{x}, \hat{y})$. To better illustrate the state transition process of variables over time in each motion model, we introduce the time interval Δt, which is defined as follows:

$$\Delta t = \tau - (t-1)\sigma. \quad (4)$$

Δt is the distance between the integral variable τ and the integral lower limit $(t-1)\sigma$ during the integration process. A tricky problem is that directly setting each variable in (2) and (3) to be time-varying would result in non-integrable outcomes. A key insight is to leverage various motion models to simplify the complex nonlinear motion of objects to varying degrees, while accurately capturing the distinct motion patterns of different object categories. The velocity transition function $v(\tau)$ is formulated as:

$$v(\tau) = \begin{cases} v_{t-1} + a\Delta t & if \ \ T = T^{CTRA} \\ v_{t-1} & if \ \ T = T^{BIC} \end{cases}, \quad (5)$$

Fig. 3 illustrates the angle η between the velocity of the object and the *X-axis* of the coordinate system, and its state transition process is described by:

$$\eta(\tau) = \begin{cases} \theta(\tau) & if \ \ T = T^{CTRA} \\ \theta(\tau) + \beta(\tau) & if \ \ T = T^{BIC} \end{cases}, \quad (6)$$

(a) CTRA Model　　　　(b) Bicycle Model

Fig. 3. **Representation of *CTRA Model* and *Bicycle Model* in 2D and 3D space.**

where β represents the slip angle between the velocity and heading of the object, which can be calculated from assumed constant steering angle δ according to:

$$\beta(\tau) = tan^{-1}(\frac{l_r}{\gamma l}tan(\delta(\tau))), \quad (7)$$

where γ is the ratio of the wheelbase to object length l. l_r denotes the distance between the gravity center and the rear tire of the object, which is artificially set to 0.4-0.5 times the wheelbase. (7) is the embodiment of retaining the rigid structure of the object, and it also constitutes the major distinction between *CTRA Model* and *Bicycle Model*. The reason for introducing β is that the instantaneous center of the object in the *Bicycle Model* is not on the body of the object. In addition, incorporating l in (7) signifies a deeper utilization of object observation and state information, enhancing motion accuracy. However, a crucial observation that follows is that *Bicycle Model* is susceptible to erroneous predictions caused by incorrect object structure information, thereby rendering it unsuitable for object categories where detectors tend to produce inaccurate detections.

$\theta(\tau)$ represents the heading angle transition function of an object, which is expressed uniformly in all models as:

$$\theta(\tau) = \theta_{t-1} + \omega(\tau)\Delta t. \quad (8)$$

$\omega(\tau)$ in (8) describes the turn rate transition function, which is formulated as:

$$\omega(\tau) = \begin{cases} \omega_{t-1} & if \ T = T^{CTRA} \\ \frac{v(\tau)sin(\beta(\tau))}{l_r} & if \ T = T^{BIC} \end{cases}, \quad (9)$$

which is actually constant in all motion models. (2)-(9) are the complete expression of state transition function $f(\cdot)$.

C. Multi-Category Data Repetition Association Module

In the data association process, a crucial but frequently disregarded fact exists: *Different object categories are sensitive to various similarity metrics and association thresholds as a result of their unique geometric characteristics.* However, most existing 3D MOT methods [3], [9] leverage a single tracking standard for each category in multi-category scenarios, resulting in inferior tracking performance due to the lack

TABLE II

A COMPARISON OF EXISTING ALGORITHMS APPLIED TO THE NUSCENES TEST SET. THE BEST PERFORMANCE IS MARKED IN RED, THE SECOND IS MARKED IN BLUE. (BIC, MOTOR, PED, TRA, TRU) REFERS TO (BICYCLE, MOTORCYCLE, PEDESTRIAN, TRAILER, TRUCK).

Method	Detector	Input Data	AMOTA↑								IDS↓	FP↓	FN↓
			Overall	Bic	Bus	Car	Motor	Ped	Tra	Tru			
CAMO-MOT [5]	BEVFuison [16] & FocalsConv [17]	2D + 3D	75.3	59.2	77.7	85.8	78.2	85.8	72.3	67.7	324	17269	18192
CBMOT [18]	CenterPoint [8] & CenterTrack [20]	2D + 3D	68.1	46.2	66.8	83.3	70.7	82.3	69.6	57.5	709	21604	22828
EagerMOT [4]	CenterPoint [8] & Cascade R-CNN [21]	2D + 3D	67.7	58.3	74.0	81.0	62.5	74.4	63.6	59.7	1156	17705	24925
Minkowski Tracker [6]	Minkowski Tracker [6]	3D	69.8	44.3	72.3	83.9	72.6	76.8	75.3	63.4	325	19340	21220
SimpleTrack [9]	CenterPoint [8]	3D	66.8	40.7	71.5	82.3	67.4	79.6	67.3	58.7	575	17514	23451
OGR3MOT [22]	CenterPoint [8]	3D	65.6	38.0	71.1	81.6	64.0	78.7	67.1	59.0	288	17877	24013
CenterPoint [8]	CenterPoint [8]	3D	65.0	33.1	71.5	81.8	58.7	78.0	69.3	62.5	684	17355	24557
Ours	LargeKernel3D [19]	2D + 3D	75.4	58.2	78.6	86.5	81.0	82.0	75.1	66.2	292	19673	17956

of category-specific pertinence. To address these issues, we introduce Multi-Category Data Repetition Association Module that enables the tracker to choose the optimal similarity metric from a set of custom multiple metrics for each object category, thereby improving the accuracy and robustness of the MOT system. In addition, a two-stage association strategy based on different similarity metrics is applied to the module to reduce false negative matches.

First Association. After obtaining $T_{t,t-1}$ and D_t, affinity between $T_{t,t-1}$ and D_t need to be calculated at each frame t. We first design three robust similarity metrics for distinct object categories to construct the first motion cost matrix $C_t^1 \in R^{N_{cls} \times N_{det,t} \times N_{tra,t-1}}$ between D_t and $T_{t,t-1}$. $N_{tra,t-1}$ and $N_{det,t}$ represent the number of $T_{t,t-1}$ and D_t, respectively. N_{cls} is the number of categories in the dataset. We propose two similarity metrics (11), (12), (13) by the first time. In addition, we introduce a rotation angle penalty factor in a specific metric to avoid false-positive associations in the opposite direction. These three similarity metrics, including 3D Generalized Intersection over Union ($gIoU_{3d}$), BEV Generalized Intersection over Union ($gIoU_{bev}$), and Euclidean Distance (d_{eucl}), are described as follows:

$$gIoU_{3d}(B_i, B_j) = IoU_{3d}(B_i, B_j) + \frac{V(B_i \cup B_j)}{V_{3dhull}(B_i, B_j)} - 1, \tag{10}$$

$$gIoU_{bev}(B_i, B_j) = IoU_{bev}(B_i, B_j) + \frac{A(B_i \cup B_j)}{A_{bevhull}(B_i, B_j)} - 1, \tag{11}$$

$$d_{eucl}(B_i, B_j) = d(B_i, B_j) * (2 - cos|\Delta\theta|), \tag{12}$$

$$d(B_i, B_j) = \gamma_{geo}||B_i^{wlh} - B_j^{wlh}||_2 + \gamma_{dis}||B_i^{xyz} - B_j^{xyz}||_2, \tag{13}$$

where B is formulated as a high-dimensional vector representing the states of $T_{t,t-1}$ or D_t, which contain the 3D size and 3D center position. IoU_{3d} and IoU_{bev} are Intersection over Union in the 3D and bird's-eye view (BEV) representation space. $V(B_i \cup B_j)$ and $A(B_i \cup B_j)$ are the union volume and area of B_i and B_j. $V_{3dhull}(B_i, B_j)$ and $A_{bevhull}(B_i, B_j)$ are the convex hulls computed by B_i and B_j in the 3D and BEV representation space. B^{xyz} and B^{wlh}

are the vectors containing the 3D center position and 3D size of B. γ_{geo} and γ_{dis} are geometric and spatial distance ratios to the overall distance. $\Delta\theta \in [0, \pi]$ is the heading angle difference between B_i and B_j. $||\cdot||_2$ is the 2-norm function.

For each category, we obtain the cost matrix $C_{t,cls}^1 \in R^{N_{det,t} \times N_{tra,t-1}}$ by utilizing its optimal-performing similarity metric to compute the affinity of this category between D_t^{cls} and $T_{t,t-1}^{cls}$[2]. After aggregating $C_{t,cls}^1$, we end up with C_t^1. Hungarian algorithm [15] is employed to match D_t and $T_{t,t-1}$ based on C_t^1. To account for the geometric size differences between objects of different categories, we employ different association thresholds $\theta_{fm} = \left(\theta_{fm}^1, \cdots, \theta_{fm}^{N_{cls}}\right)$ to constrain the matching process. After matching, we obtain three classes of matching instances, including matched pairs $DT_t^{1m} = \left\{ \left(D_t^i, T_{t,t-1}^j\right), \cdots \right\}$, unmatched detections $D_t^{1u} \subseteq D_t$, and unmatched trajectories $T_{t-1}^{1u} \subseteq T_{t-1}$. D_t^{1u} and T_{t-1}^{1u} will be further associated in the second stage.

Second Association. To reduce false-negative associations, we use $gIoU_{bev}$ for objects of all categories[3] to construct the cost matrix $C_t^2 \in R^{N_{umdet,t} \times N_{umtra,t-1}}$ between D_t^{1u} and T_{t-1}^{1u} in the second stage[2]. $N_{umdet,t}$ and $N_{umtra,t-1}$ are the number of D_t^{1u} and T_{t-1}^{1u}, respectively. We use the Hungarian Algorithm with a strict threshold θ_{sm} based on the cost matrix C_t^2 to match D_t^{1u} and T_{t-1}^{1u}. After aggregating the matching results of the two-stage association, we obtain the final matched pairs DT_t^m, unmatched detections $D_t^u \subseteq D_t$, and unmatched trajectories $T_{t-1}^u \subseteq T_{t-1}$.

D. Trajectory Management Module

Following most 3D MOT methods [3], [4], the trajectory management module is also responsible for four key functions, which include trajectory updating, trajectory initialization, trajectory death, and output file organization.

Trajectory Update. We utilize the detection in DT_t^m and the standard update process of EKF to update the state of the corresponding trajectory and covariance matrix. It is important to note that in the state-measurement transition

---

[2]Costs between different categories are filled with invalid values.
[3]If an object utilizes $gIoU_{bev}$ in the first association, then $gIoU_{3d}$ will be applied in the second stage, as the core of multi-stage association is to use different metrics to perform repeated associations.

TABLE III

A COMPARISON OF EXISTING METHODS APPLIED TO THE NUSCENES VAL SET. ALL MAIN METRICS REPORTED IN COMPETITOR PAPERS ARE LISTED.

Method	Detector	Input Data	AMOTA↑	AMOTP↓	IDS↓
CBMOT [18]	CenterPoint [8] & CenterTrack [20]	2D + 3D	72.0	**48.7**	479
EagerMOT [4]	CenterPoint [8] & Cascade R-CNN [21]	2D + 3D	71.2	56.9	899
SimpleTrack [9]	CenterPoint [8]	3D	69.6	54.7	405
CenterPoint [8]	CenterPoint [8]	3D	66.5	56.7	562
OGR3MOT [22]	CenterPoint [8]	3D	69.3	62.7	**262**
Ours	CenterPoint [8]	3D	**73.1**	**52.1**	281
Ours	LargeKernel3D-L [19]	3D	**75.2**	54.1	**252**

function $h(\cdot)$ of *Bicycle model*, the geometric center of objects should be calculated based on the gravitational center.

Trajectory Initialization. We employ the *count-based* approach to initialize D_t^u as new tentative trajectories $T_{ten,t}$. If the j-th $T_{ten,t}$ is continuously hit in the next hit_{min} frames, $T_{ten,t}^j$ will change to an activate trajectory and be merged into still active trajectories.

Trajectory Death. We adopt the *count-based* scheme to discard T_{t-1}^u. Part of the trajectory in T_{t-1}^u will be discarded if it has not been updated in the last *max-age* frames. Trajectories that are not deleted are still considered active, but we penalize the confidence scores of these trajectories using α_{pun} and the exponential function $exp(\cdot)$.

Result Output. After obtaining all active trajectories T_t at the current frame t, the updated trajectories (estimated motion state), newly initialized trajectories, and parts of the penalized trajectories are output to the result file. Note that, to reduce false-positive predictions, we only output N_{pun} frames of the penalized trajectories' predicted state to the result file, and also apply NMS with $\theta_{nms} = 0.08$ to all output trajectory states.

IV. EXPERIMENTS

A. Datasets

NuScenes. NuScenes [12] contains 850 training sequences and 150 test sequences, each comprising approximately 40 frames showcasing diverse scenarios such as rainy days and nights. The keyframes are sampled at a frequency of 2Hz, and annotation information is provided for each keyframe. However, this keyframe frequency poses a challenge for precise motion model prediction, leading to significant inter-frame displacement. The official evaluator utilizes AMOTA as the primary evaluation metric [3].

B. Implementation Details

NuScenes. Our tracking method is implemented in Python under the Intel® 9940X without any GPU. Hyperparameters are chosen based on the best AMOTA identified in the validation set. We utilize $\theta_{nms} = 0.08$ for all categories and 3D detectors. θ_{SF} is detector-specific. IoU_{bev} is used as the metric in NMS. During NMS process, objects of all categories are blended together. We employ *Bicycle model* with $\gamma = 0.8$ for *(bicycle, motorcycle)* and *CTRA model* for the remaining categories. The similarity metric for *bus* and *(bicycle, motorcycle, car, trailer, truck, pedestrian)* are $gIoU_{bev}$ and $gIoU_{3d}$, respectively. We apply $\theta_{fm} = (1.6, 1.4, 1.3, 1.3, 1.3, 1.2, 1.7)$ and *max-age* $=$ $(10, 20, 10, 15, 10, 20, 10)$ for *bicycle, motorcycle, bus, car, trailer, truck, pedestrian* and $\theta_{sm} = 1$ for all seven categories in the data association module. For trajectory management, we set $hit_{min} = 0$, $\alpha_{pun} = 0.05$, $N_{pun} = 1$.

C. Experimental Results

1) Run-time discussion: To solve the real-time challenge caused by extensive affinity calculations brought by a large number of objects, we first proposed the half-parallel[4] $gIoU$ operator under the Python implementation. On the NuScenes, Poly-MOT can run at 3 FPS (Frame Per Second) on Intel 9940X, which has surpassed most advanced 3D MOT methods (SimpleTrack 0.51 FPS, Minkowski Tracker 1.7 FPS).

2) Comparative Evaluation: We compare Poly-MOT to published and peer-reviewed state-of-the-art methods on the test and validation sets of the NuScenes dataset.

NuScenes Test Set. Among all 3D MOT methods, Poly-MOT **ranks first** on the NuScenes tracking benchmark test set, i.e., 75.4% AMOTA, exceeding most 3D MOT methods. As shown in Table II, Poly-MOT achieves an impressively low IDS 292 while maintaining the highest AMOTA (75.4%) among all modal methods, which indicates that Poly-MOT is capable of achieving stable tracking without loss of recall. Without any image data as additional input, Poly-MOT still acquires state-of-the-art performance, surpassing the best-performing multi-modal tracker CAMO-MOT, which leverages a more superior integrated detector through [16], [17]. Additionally, Poly-MOT outperforms competing algorithms by a significant margin in the crucial category (*Car*). Compared to learning-based methods [5], [6], [8], Poly-MOT incurs minimal computational overhead and delivers a more impressive performance, highlighting the promising potential of integrating filter-based 3D MOT methods into practical robotic systems. Notably, the IDS of Poly-MOT is slightly inferior to that of OGR3MOT [22]. However, the FN/FP in Table II shows that Poly-MOT can offer the same robust continuous tracking capability without compromising recall.

NuScenes Val Set. As presented in Table III, Poly-MOT outperforms other trackers in terms of both higher AMOTA and lower IDS when adopting the same detector (CenterPoint [8]). Moreover, Poly-MOT yields an incredible tracking performance when assembled with a more strong LiDAR-only detector [19], i.e., 75.2% AMOTA, exceeding the best validation set accuracy reported by most methods.

---

[4]Since convex hull and rotation IoU calculations are still serial.

TABLE IV

THE RESULTS OF THE ABLATION STUDY OF EACH MODULE ON THE
NUSCENES VAL SET. **OS** MEANS ORIGINAL STATE. **PRE** MEANS
PRE-PROCESSING MODULE. **MO** MEANS TRAJECTORY MOTION
MODULE. **ASS** MEANS DATA ASSOCIATION MODULE.

Module	AMOTA↑	IDS↓	FN↓	FP↓
Os	67.4	467	21442	14009
Os + Pre	71.4	374	18099	13299
Os + Pre + Mo	71.9	443	18086	13340
Os + Pre + Ass	72.0	410	15979	15932
Os + Pre + Mo + Ass	73.1	281	17637	13437

TABLE V

THE ABLATION STUDY OF WHETHER OR NOT TO USE SCORE FILTER
AND NON-MAXIMUM SUPPRESSION. RUN-TIME REFERS TO THE
RUNNING TIME OF PRE-PROCESSING MODULE.

Variable	AMOTA↑	IDS↓	Run-Time (s) ↓
NMS + SF	73.1	281	0.055
NMS	71.8	320	0.093
SF	68.6	354	0.008

3) Ablation Studies: In this part, we conduct extensive ablation experiments to evaluate the individual performance of proposed modules in Poly-MOT. We select CenterPoint [8] as the 3D detector and employ *CA Model* with Linear Kalman Filter to predict the trajectory state from the Origin State (OS). We leverage $gIoU_{3d}$ and θ set to 0.14 as the similarity metric and association threshold, respectively. A series of experiments are then performed on the NuScenes validation set using various module combinations.

The effect of Pre-processing Module. The significant gap between "Os" and "Os+Pre" in Table IV showcases the impact of leveraging Pre-processing Module on the overall performance. We can observe that "Os+Pre" provides a +4% AMOTA boost and a 93 IDS drop, resulting in a significant performance boost. The reason is that SF can filter out low-score bounding boxes while NMS can remove duplicate bounding boxes with high confidence, which makes the remaining bounding boxes have superior quality. In addition, using SF before NMS brings inference 40% reduction in pre-processing inference time while boosting AMOTA by 1.3% compared with only using NMS, as demonstrated in Table V.

The effect of Multi-Category Trajectory Motion Module. In Table IV, we demonstrate the impact of the Multi-Category Trajectory Motion Module. "Os+Pre+Mo+Ass" achieves an AMOTA improvement of +1.1% and an IDS decrease of 129 compared to "Os+Pre+Ass". Benefiting from improved trajectory estimation, we can apply stricter thresholds to filter FP (-2495) in complex scenes (objects are dense and numerous, detectors exhibit poor performance, etc.) to achieve more stable tracking (-129 IDS) without incurring a significant loss in recall (+1658 FN). In addition, an intriguing observation is that while "Os+Pre+Mo" yields a +0.5% AMOTA boost over "Os+Pre" alone, it also causes more ID switches (+69). The key insight is that the more accurate motion models change the bias distribution between predictions and ground truths for individual object categories, which makes a single metric and threshold unable to accurately capture inter-object affinities, thereby obtaining

TABLE VI

ABLATION STUDIES USING DIFFERENT MOTION MODELS IN
MULTI-CATEGORY (FOR *Bic* AND *Moto*). AMOTA AND IDS ARE
REPORTED BEST FOR DIFFERENT MOTION MODULES.

Category	Motion Model	AMOTA↑	IDS↓	FP↓	FN↓
Bic	*Bicycle*	57.1	0	227	765
	CTRA	55.4	0	256	747
	CA	56.1	1	234	765
Moto	*Bicycle*	77.0	1	121	464
	CTRA	73.6	4	154	537
	CA	75.1	6	94	547

false matches and leading to IDS. Moreover, Table VI reveals that using an inappropriate motion model for objects would decrease tracking performance, underscoring the importance of carefully deciding the motion model for each category.

The effect of Multi-Category Data Repetition Association Module. As shown in Table IV, "Os+Pre+Mo+Ass" achieves a +1.2% AMOTA improvement and a -162 IDS reduction compared to "Os+Pre+Mo". This shows our proposed two-stage categorical association strategy can better capture the affinity between tracklet and detection of each category, enabling a more accurate matching relationship, improved tracking results and reduced FN matches.

D. Visualization

We qualitatively compare our Poly-MOT (LiDAR-only version) and advanced multi-modal 3D MOT method CB-MOT on the NuScenes val set. As shown in Fig. 4 (a), when the object moves intensely and quickly, CBMOT has ID switches (ID changes from *20* to *247*), while the Poly-MOT can still achieve stable tracking. As shown in Fig. 4 (b), when objects are dense and have irregular movement, CBMOT not only has ID switches (ID changes from *37* to *25*) but also fails to effectively suppress false-positive detection (ID: *231* at Frame 12), while Poly-MOT still maintains stable tracking. The above comparison results show that Poly-MOT can alleviate the problem that LiDAR-only trackers cannot accurately track objects with large inter-frame displacements. In addition, Poly-MOT can also achieve stable tracking when the object suffers from occlusion.

V. CONCLUSIONS

In this work, we introduce Poly-MOT, a polyhedral framework for 3D MOT under multi object category scenarios following the TBD framework. Poly-MOT achieves accurate matches between tracklets and detections in multi-category scenarios by ensuring prediction reliability and metric rationality, including: (1) Two distinct and nonlinear motion models (*CTRA* and *Bicycle* Model) are established to represent the motion patterns of different object categories; (2) Three similarity metrics ($gIoU_{3d}$, $gIoU_{bev}$, d_{eucl}) are designed to calculate the affinity of different object categories. Besides, a two-stage association strategy and confidence-based pre-processing module are applied to the tracker to reduce FN matches and eliminate the gap between detection and tracking. Without requiring additional training and GPU, Poly-MOT achieves state-of-the-art tracking performance with 75.4% AMOTA on the NuScenes dataset while achieving

(a) Scene1:Nuscene scene-0524   (b) Scene2:Nuscene scene-0919

Fig. 4. **Visualization of comparison results between Poly-MOT and CBMOT [18]**. All methods use CenterPoint as a 3D detector. CBMOT simultaneously uses CenterTrack [20] as a 2D detector for multi-modal fusion.

an impressive inference speed. Our method can be easily combined with multiple detectors, and we envision it serving as a general baseline for future 3D MOT methods.

REFERENCES

[1] Y. Zhang, P. Sun, Y. Jiang, D. Yu, F. Weng, Z. Yuan, P. Luo, W. Liu, and X. Wang, "Bytetrack: Multi-object tracking by associating every detection box," in *European Conference on Computer Vision*. Springer, 2022, pp. 1–21.

[2] Y. Zhang, C. Wang, X. Wang, W. Zeng, and W. Liu, "Fairmot: On the fairness of detection and re-identification in multiple object tracking," *International Journal of Computer Vision*, vol. 129, no. 11, pp. 3069–3087, 2021.

[3] X. Weng, J. Wang, D. Held, and K. Kitani, "3d multi-object tracking: A baseline and new evaluation metrics," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 10 359–10 366.

[4] A. Kim, A. Ošep, and L. Leal-Taixé, "Eagermot: 3d multi-object tracking via sensor fusion," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 11 315–11 321.

[5] L. Wang, X. Zhang, W. Qin, X. Li, L. Yang, Z. Li, L. Zhu, H. Wang, J. Li, and H. Liu, "Camo-mot: Combined appearance-motion optimization for 3d multi-object tracking with camera-lidar fusion," 2022. [Online]. Available: https://arxiv.org/abs/2209.02540

[6] J. Y. Gwak, S. Savarese, and J. Bohg, "Minkowski tracker: A sparse spatio-temporal r-cnn for joint object detection and tracking," *arXiv e-prints*, 2022.

[7] K. Huang and Q. Hao, "Joint multi-object detection and tracking with camera-lidar fusion for autonomous driving," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 6983–6989.

[8] T. Yin, X. Zhou, and P. Krähenbühl, "Center-based 3d object detection and tracking," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 11 779–11 788.

[9] Z. Pang, Z. Li, and N. Wang, "Simpletrack: Understanding and rethinking 3d multi-object tracking," 2021.

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

[11] L. Leal-Taixé, A. Milan, I. Reid, S. Roth, and K. Schindler, "Motchallenge 2015: Towards a benchmark for multi-target tracking," *arXiv preprint arXiv:1504.01942*, 2015.

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

[13] Tao Xie, Kun Dai, Ke Wang, Ruifeng Li, and Lijun Zhao. Deep-matcher: A deep transformer-based network for robust and accurate local feature matching. *arXiv preprint arXiv:2301.02993*, 2023.

[14] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine *et al.*, "Scalability in perception for autonomous driving: Waymo open dataset," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 2446–2454.

[15] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[16] Z. Liu, H. Tang, A. Amini, X. Yang, H. Mao, D. Rus, and S. Han, "Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation." arXiv, 2022. [Online]. Available: https://arxiv.org/abs/2205.13542

[17] Y. Chen, Y. Li, X. Zhang, J. Sun, and J. Jia, "Focal sparse convolutional networks for 3d object detection," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 5418–5427.

[18] N. Benbarka, J. Schröder, and A. Zell, "Score refinement for confidence-based 3d multi-object tracking," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 8083–8090.

[19] Chen, Y., Liu, J., Qi, X., Zhang, X., Sun, J., and Jia, J. (2022). Scaling up kernels in 3d cnns. arXiv preprint arXiv:2206.10555.

[20] X. Zhou, V. Koltun, and P. Krähenbühl, "Tracking objects as points," in *European Conference on Computer Vision*. Springer, 2020, pp. 474–490.

[21] Z. Cai and N. Vasconcelos, "Cascade r-cnn: Delving into high quality object detection," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6154–6162.

[22] J.-N. Zaech, A. Liniger, D. Dai, M. Danelljan, and L. Van Gool, "Learnable online graph representations for 3d multi-object tracking," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 5103–5110, 2022.

[23] X. Bai, Z. Hu, X. Zhu, Q. Huang, Y. Chen, H. Fu, and C.-L. Tai, "Transfusion: Robust lidar-camera fusion for 3d object detection with transformers," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 1080–1089.

[24] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," in *2016 IEEE international conference on image processing (ICIP)*. IEEE, 2016, pp. 3464–3468.

[25] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in *2017 IEEE international conference on image processing (ICIP)*. IEEE, 2017, pp. 3645–3649.

[26] S. Sun, N. Akhtar, H. Song, A. Mian, and M. Shah, "Deep affinity network for multiple object tracking," *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 1, pp. 104–119, 2019.

[27] Tao Xie, Shiguang Wang, Ke Wang, Linqi Yang, Zhiqiang Jiang, Xingcheng Zhang, Kun Dai, Ruifeng Li, and Jian Cheng. Poly-pc: A polyhedral network for multiple point cloud tasks at once. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1233–1243, 2023.