2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

GRU-Based Kalman Filtering for 3D Multi-Object Tracking

Zikang Yuan[1,2†], Xiaoxiang Wang[3†], Jiaxin Liu[3], Miaojie Feng[3], Zhaoxing Zhang[3] and Xin Yang[3✉]

***Abstract*— 3D multi-object tracking is a crucial task for autonomous driving and robotics. Although tracking-by-detection-based approaches have demonstrated excellent performance in recent years, they ignore varying motion characteristics of different objects and utilize a single state space and estimator to model multiple categories. On the other hand, they model the noise of motion state as a ideal Gaussian distribution, which fails to capture the true complexity of the noise dynamics. These oversimplified modeling assumptions results in significant reductions of tracking precision. To address this limitation, we propose a learnable Gated Recurrent Unit (GRU)-based Kalman filtering, which is able to learn object motion characteristics through data-driven learning, thereby avoiding the necessity for manual motion and noise model design. To avoid abnormal supervision caused by the wrong association between annotations and trajectories, we adopt a hybrid-supervised learning strategy to accelerate the convergence speed and improve the robustness of the proposed method. Experimental results on two public datasets demonstrate that the proposed GRU-based Kalman filtering exhibits superior performance and significant potential compared to existing state-of-the-art approaches.**

I. INTRODUCTION

3D multi-object tracking (MOT) is a crucial task for many applications, e.g., autonomous driving [1]–[3] and robotics [4]–[7]. In rencent years, many tracking-by-detection (TBD)-based approaches [8]–[10] have demonstrated superior accuracy and robustness with the increasing performance of 3D multi-object detection (MOD) [11]–[14]. Unlike TBD-based methods that focus on the motion process of the target object, joint detection and tracking (JDT)-based approaches typically accomplish the object tracking task end-to-end through a learnable network. However, due to the scarcity of accurately annotated data, the accuracy and generalization of such methods are generally inferior to TBD-based approaches.

TBD-based approaches incrementally update the state of tracked objects by constructing motion models and employing recursive Bayesian filter estimators. However, due to the varying motion characteristics of different objects within the scene, a single state space and estimator cannot effectively match the distinct motion characteristics across various categories. This discrepancy reduces the consistency between the motion state updates and the actual conditions, leading to false matching and inaccurate state updates. Some approaches [10], [15], [16] have addressed this issue by designing motion parameters or association strategies tailored for each category, making them more relevant to the unique characteristics of different categories. However, these methods fail to fundamentally address the problem of multi-category differences. On one hand, with the continuous addition or refinement of categories, designing motion models for each category becomes not only tedious but also overly reliant on the designer's experience. On the other hand, most TBD-based approaches adopt a motion module based on Kalman filtering. The idealized linear assumptions employed by this framework do not adapt well to MOT whose model is usually nonlinear and complex. Furthermore, the noise modeling still utilizes the idealized assumption of Gaussian white noise, which does not accurately reflect the actual conditions. These combined factors lead to suboptimal precision in object tracking.

As neural network-based state estimation has demonstrated the capability to capture the motion characteristics of complex processes [17], [18], which is also applicable to state transitions in MOT. Based on this, we propose a partially learnable MOT method by introducing a Gated Recurrent Unit (GRU)-based Kalman filtering into the motion module of TBD-based approaches. This method can replace the traditional manual model design of Kalman filtering in data-driven manner, thus eliminating the necessity to design motion and noise model for each category. Specifically, we use multiple GRUs to simulate the loops in recursive Bayesian filtering. The model automatically learns the noise distribution, state covariance matrix and observation covariance matrix. It avoids the mismatch in noise modeling and the loss of precision associated with linearizing the state transition and observation functions. However, directly applying the proposed learnable Kalman filtering in MOT is not feasible. The partial annotation of the dataset results in a limited amount of trainable data, which increases the risk of overfitting. Furthermore, the annotations and trajectories are linked through a manually designed association strategy, meaning that any errors in association can introduce erroneous supervision into the framework. Therefore, we propose to parallel a traditional Kalman filter during the training process to generate pseudo-labels for the unlabeled data, facilitating hybrid-supervised training. Experimental results on two public datasets demonstrate that the proposed method outperforms existing state-of-the-arts according to tracking

This research was partially conducted by ACCESS – AI Chip Center for Emerging Smart Systems, supported by the InnoHK initiative of the Innovation and Technology Commission of the Hong Kong Special Administrative Region Government, and was partially conducted by National Natural Science Foundation of China (62122029, 62472184) and the Fundamental Research Funds for the Central Universities.

[1]ACCESS - AI Chip Center for Emerging Smart Systems, InnoHK Centers, Hong Kong Science Park, Hong Kong, China.
[2]HongKong University of Science and Technology, HongKong, China.
[3]Huazhong University of Science and Technology, Wuhan, China.
E-mail: xinyang2014@hust.edu.cn
† means that the two authors contributed equally to this work.
✉ means corresponding author.

accuracy and generalization.

To summarize, the main contributions of this work are two folds: 1) We propose a data-driven MOT method by using a GRU-based Kalman filtering to avoid the precision loss of traditional methods for noise mismatch modeling and motion process linearization; 2) We design a pseudo-label-based hybrid-supervised training strategy, which greatly expands the amount of available training data and in turn increase the converge speed of network. 3) We have release the code for the development of the community[1].

The rest of this paper is structured as follows. In Sec. II, we briefly discuss the relevant literature. Sec. III details our methodology. Sec. IV provides experimental evaluation. Finally, we conclude this paper in Sec. V.

II. RELATED WORK

A. TBD-Based Approaches

AB3DMOT [8] designs a 3D MOT framework based on Kalman filtering and intersection over union (IoU), laying the foundation for the design of TBD-based approaches. It divides the tracking detection paradigm into four stages: (1) preprocessing the upstream 3D multi-object detection results; (2) motion prediction and state updating for existing target trajectories; (3) predicting associations between targets and detected objects; and (4) managing the lifecycle of target trajectories. SimpleTrack [9] provides a detailed analysis of the impact on these four stages, examining the advantages and disadvantages of the widely used Kalman filtering motion model compared to the constant velocity motion model, as well as evaluating various metrics for data association. Finally, it proposes corresponding improvements for each stage module and combines them into a simple framework, demonstrating superior tracking competitiveness. Poly-MOT [10] considers the variability of motion across different categories and establishes multiple motion models based on the distinct kinematic features, enabling the capture of motion pattern differences between categories. Additionally, to address the differing shapes of various categories, it designs three custom similarity metrics and a novel two-stage data association strategy to ensure that different objects can utilize the best similarity metric for their respective categories, thereby reducing erroneous matches. Poly-MOT further optimizes the tracker within TBD framework from the perspective of multi-category variability, achieving the outstanding tracking performance. In the association matching stage of TBD, several methods have also conducted in-depth optimization research. Florian et. al. [19] proposes a statistical data association method under a Bayesian estimation framework. NEBP [20] combines model-based and data-driven multi-object tracking, with a proposed neural-enhanced confidence propagation method that supplements the statistical model of confidence propagation by learning from the information derived from the original sensor data. This method infers learned information that can reduce model mismatches, thus improving the false positive and false negative rates of data association. In addition to relying on positional information, some methods have attempted to improve the independent tracking phase by integrating other feature information. EagerMOT [15] and ShaSTA-Fuse [21] integrate image detection results with LiDAR detection results to achieve comprehensive perception of dynamic scenes. Similarly, CAMO-MOT [16] and Bevfusion [22] utilize both camera and LiDAR data, significantly reducing tracking failures caused by occlusions and misdetections. It also introduces an occlusion head that effectively selects the best object appearance features multiple times, further mitigating the impact of occlusion.

B. JDT-Based Approaches

Motiontrack [23] constructs an end-to-end joint optimization algorithm based on a full transformer architecture. This model inherits the advantages of cross-modal feature fusion from the TransFusion [24] detection framework and innovatively introduces a temporal-aware data association module and a query decoder enhancement mechanism, achieving collaborative optimization for 3D object detection and inter-frame tracking. Its architectural design possesses multimodal compatibility, supporting both single-modal point cloud inputs and multimodal fusion inputs. OGR3MOT [25] employs an end-to-end approach utilizing graph neural networks to solve 3D MOT problem, achieving the outstanding identity switch (IDS) metrics. Some methods enhance tracking by leveraging feature representations from detection networks. For instance, ShaSTA [21] utilizes feature representations from LiDAR 3D detectors to learn the shape and spatiotemporal affinities between trajectories and detections across consecutive frames, providing a probabilistic matching mechanism that enables robust data association, track lifecycle management, false positive elimination, false negative propagation, and track confidence refinement. ShaSTA effectively addresses false positives and missed detections in cluttered scenes and occlusions, achieving advanced performance. PolarMOT [26] proposes a purely geometry-driven method for 3D MOT, modeling the spatiotemporal geometric relationships between targets using a local polar coordinate system. It constructs a graph neural network that transforms data association into an edge classification task, avoiding reliance on appearance features. Gwak et. al. [27] introduces a sparse spatiotemporal framework for end-to-end joint detection and tracking, generating bird's-eye view (BEV) features from 4D point cloud inputs. Then [27] aggregates trajectory region of interest features directly from the BEV to predict the matching probabilities for detections and trajectories for updating tracks. For these JDT-based approaches, utilizing large-scale feature representations during the tracking phase results in excessive computational demands, presenting significant bottlenecks for practical deployment. Additionally, JDT-based approaches reduce the interpretability of optimality in traditional TBD-based approaches.

[1]https://github.com/xiang-1208/GRUTrack

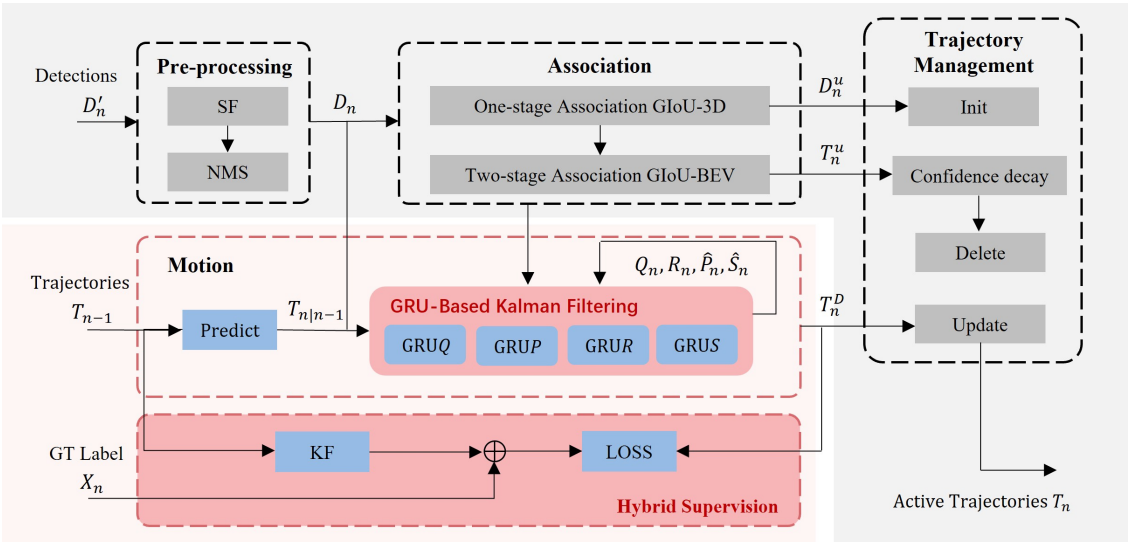


Fig. 1. Overview of our framework, which is based on the fundamental system paradigm of TBD and can be divided into four parts: pre-processing module, motion module, association module, and trajectory management module. Our contributions focus on the motion module including two parts, i.e., GRU-based Kalman filtering and hybrid-supervised training.

III. METHODOLOGY

A. 3D MOT Pipeline

Our system can be divided into four parts: the pre-processing module, motion module, association module, and trajectory management module, as shown in Fig. 1.

1) Pre-processing Module: 3D MOD typically generates multiple bounding boxes for the same detection to minimize the risk of missed and false detections. To prevent these detections from causing redundant ID switches, we preprocess the original detections D'_n to reduce false matches. This preprocessing generally involves applying score filtering and non-maximum suppression (NMS) [9] to each sweep's detections, retaining only the bounding box with the highest confidence for each object. After preprocessing, our detection boxes $D_n = [x, y, z, w, l, h, v, \theta]$ include the center position of the bounding box, its dimensions, its velocity, and the heading angle.

2) Motion Module: The motion module is primarily responsible for predicting the state $\hat{X}_{n-1} = \left[\hat{x}_n^1, \ldots, \hat{x}_n^{num\_tra}\right]$ of the tracked trajectories and updating the state based on observations $Y_n = \left[y_n^1, \ldots, y_n^{num\_det}\right]$. The process of traditional Kalman filtering is illustrated in Fig. 2. The system continuously updates the state through the prior state $\hat{x}_{n-1}$, the observation y_n, the process noise covariance matrix Q_n, the measurement noise covariance matrix R_n, and the continuously maintained error state covariance P_n. Specifically, there are two steps:

a) Prediction Step: In this step, the system predicts the current prior state based on the posterior state from the previous time step:

$$\hat{x}_{n|n-1} = f(\hat{x}_{n-1}) \tag{1}$$

Simultaneously, the uncertainty of state vector (i.e., represented by covariance matrix) is propagated as:

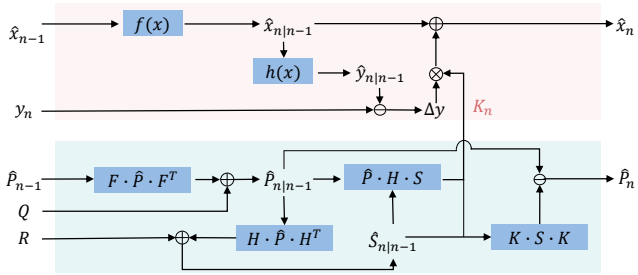


Fig. 2. Illustration of traditional Kalman filtering.

$$\hat{P}_{n|n-1} = F_n \cdot \hat{P}_{n-1} \cdot {F_n}^T + Q_n \tag{2}$$

b) Update Step: The update step involves calculating the Kalman gain K, which balances the weights between the prediction and the observation:

$$K = \hat{P}_{n|n-1} \cdot {H_n}^T \cdot (H_n \cdot \hat{P}_{n|n-1} \cdot {H_n}^T + R_n)^{-1} \tag{3}$$

$$\hat{S}_{n|n-1} = H_n \cdot \hat{P}_{n|n-1} \cdot {H_n}^T + R_n \tag{4}$$

where $\hat{S}_{n|n-1}$ is the observation covariance matrix. According to the current observation, the current state of posterior is updated as:

$$\hat{x}_n = \hat{x}_{n|n-1} + K \cdot (y_n - h(\hat{x}_{n|n-1})) \tag{5}$$

Meanwhile, the state covariance matrix is updated as:

$$\hat{P}_n = \hat{P}_{n|n-1} - K \cdot \hat{S}_{n|n-1} \cdot K^T \tag{6}$$

The traditional Kalman filtering utilizes the Jacobian matrix F_n and H_n to linearize the differentiable functions $f(x)$ and $h(x)$ in a time-dependent manner. This approach relies heavily on the accuracy of state space model setup, which often depends on the designer's experience and is difficult to transfer. On the other hand, for nonlinear motion, the

linearization of the state transition and observation equations can introduce additional errors. Additionally, the traditional Kalman filtering requires manual adjustment of the observation noise and process noise, assuming they follow a multi-dimensional Gaussian distribution. However, in MOT fields, observations are derived from upstream object detection results, which do not necessarily conform to a Gaussian distribution. Thus, explicitly modeled noise as Gaussian distribution may not be sufficient to fully adapt to MOT task.

It is reasonable to believe that the differences in motion for different categories are often also reflected in the state. If the module can adaptively obtain different Kalman gain results based on the observations and state, it can then be applied to all categories and adjust the noise accordingly.

We improve the learnable Kalman filtering [28] into our motion module, using a uniform state space model across all object categories. The specific neural network architecture is illustrated in Fig. 3. Following the design of KalmanNet, we represent each second-order statistical moment of the Kalman filter using separate GRU, with fully connected layers as input and output layers.

GRUP takes $\hat{P}_{n|n-1}$ as a variable related to P_{n-1} and Q_n. Meanwhile, the posterior state error $\Delta\tilde{x}_n$ is utilized as a dynamic adjustment signal to correct the forward propagated covariance:

$$\hat{P}_{n|n-1} = GRU\left(\hat{P}_{n-1}, \left[\hat{Q}_n, \Delta\tilde{x}_n\right]\right) \tag{7}$$

When the object's motion is smooth, $\Delta\tilde{x}_n$ is minimal, allowing the network to maintain stable covariance matrix. Conversely, when a sudden change in object's motion is detected (e.g., abrupt acceleration, deceleration, or turning), $\Delta\tilde{x}_n$ experiences a significant change. In this case, the update gate of GRU dynamically enhances the trace of $\hat{P}_{n|n-1}$ to increase the uncertainty of prediction, thereby reducing the weight of the predicted state in subsequent updates.

GRUQ takes $\hat{Q}_n$ as a variable related to $\hat{Q}_{n-1}$, while also incorporating the posterior state difference Δx_n:

$$\hat{Q}_n = GRU\left(\hat{Q}_{n-1}, \Delta x_n\right) \tag{8}$$

The small value of $\hat{Q}_{n-1}$ means the prior and posterior states should closely align, resulting in a small state update difference. In this correct scenario, the network should appropriately reduce the process noise covariance. Conversely, when there is a significant deviation between the state transition process and the actual state, the state update difference will be larger, prompting the system to increase the process noise covariance accordingly.

GRUS takes $\hat{S}_{n|n-1}$ as a variable related to $P_{n|n-1}$ and R_n, while also incorporating the observation posterior error $\Delta\tilde{y}_n$:

$$\hat{S}_{n|n-1} = GRU\left(\hat{P}_{n|n-1}\left[\hat{R}_n, \Delta\tilde{y}_n\right]\right) \tag{9}$$

The small observation posterior error $\Delta\tilde{y}_n$ means relatively stable state during this time period. In this case, the network appropriately reduces the correction of the prior observation covariance, making it more aligned with the changes in

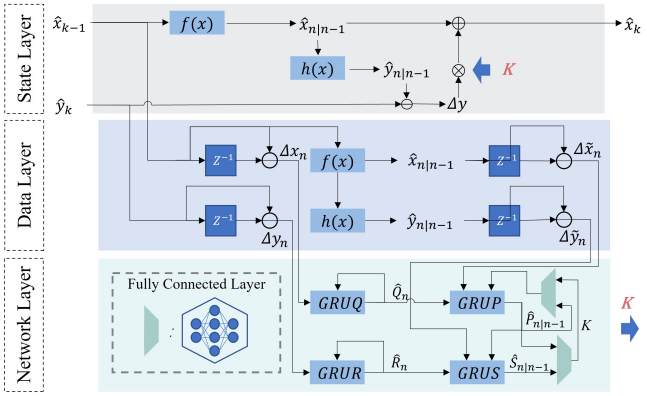


Fig. 3. Illustration of GRU-based Kalman filtering, where GRUs are utilized to simulate the loop iteration of process noise Q_n, measurement noise covariance matrix R_n, state covariance matrix $\hat{P}_{n|n-1}$ and observation covariance matrix $\hat{S}_{n|n-1}$ by inputting the state difference and observation difference, so as to reason about the Kalman gain K.

the prior observation covariance. Conversely, when there is a significant deviation between the current and previous observations, the observations become unstable, indicating that the vehicle may be experiencing sudden acceleration or deceleration. In such cases, the network adjusts the observation covariance appropriately to mitigate the weight of the prior observation in subsequent updates.

GRUR takes $\hat{R}_n$ as a variable related to $\hat{R}_{n-1}$, while also incorporating the observation update difference Δy_n:

$$\hat{R}_n = GRU\left(\hat{R}_{n-1}, \Delta y_n\right) \tag{10}$$

The observation update difference describes the discrepancy between the prior observation value inferred from the observation equation and the final updated posterior observation value. When the system's observation noise is small, the prior and posterior observation values should not differ significantly. In this case, the network appropriately reduces the observation noise covariance. Similarly, when the network detects low accuracy in results during this time period, the observation update difference will be larger, prompting the network to increase the observation noise covariance to enhance the uncertainty of the observations.

Finally, simulating the traditional Kalman filtering process, we take the state covariance matrix and observation covariance matrix as inputs to the full connected (FC) layer for ultimately outputing the Kalman gain K and updated state covariance matrix $\hat{P}_n$:

$$K = FC\left(\hat{P}_{n|n-1}, S^{-1}_{n|n-1}\right) \tag{11}$$

$$\hat{P}_n = FC\left(\hat{P}_{n|n-1}, K\right) \tag{12}$$

Compared with end-to-end deep neural network, this network structure emulates part of traditional Kalman filtering process, making it more parameter-efficient and easier to train.

3) Association Module: Following the baseline PolyMOT, we use a two-stage association strategy to reduce false negative associations. In the one-stage association, we use the

3D generalized intersection over union ($GIoU_{3D}$) [29] to as the match metric between the tracked trajectories and the detections. After one-stage association, we perform a wider threshold two-stage association between mismatched trajectories and mismatched detections. This association will be performed in BEV. By combining the two-stage association, we effectively obtain three results: updated trajectories T_n^D, mismatched trajectoriesT_n^u, and mismatched detections D_n^u.

4) Trajectory Management Module: Similar to most 3D MOT approaches, our system employs a confidence-based trajectory lifecycle management approach [30]. Specifically, when the tracker receives a detection D_n^u that is not associated with an existing trajectory, it is initialized as a new trajectory. During subsequent tracking, if a observation is associated with this trajectory, corresponding to T_n^D, the state of the trajectory is updated based on the new observation. If the trajectory is not observed in consecutive frames T_{n-1}^u, its confidence decays following an exponential function until it is ultimately deleted.

B. Training methods

The proposed GRU-based filtering is supervised trained using the true trajectory annotations from the dataset. The entire state estimation model is trained end-to-end by constructing a mean squared loss between the true state X_n and the posterior state $\hat{X}_n$ output by the motion model:

$$\mathcal{L} = \sum_{n=0}^{seq} \left\| X_n - \hat{X}_n \right\|^2 \tag{13}$$

Despite the motion model using a deep learning network to learn the Kalman gain rather than directly inferring the posterior state, it benefits from the fully differentiable nature of the Kalman filtering, which allows for calculability:

$$\begin{aligned} \frac{\partial \mathcal{L}_n}{\partial K_n} &= \frac{\partial \left\| K_n \Delta \mathbf{y}_n - \Delta X_n \right\|^2}{\partial K_n} \\ &= 2 \cdot (K_n \cdot \Delta y_n - \Delta X_n) \cdot \Delta y_n^\top \end{aligned} \tag{14}$$

where $\Delta X_n = X_n - \hat{X}_{n|n-1}$. This indicates that the gradient signal can be backpropagated to the network parameters through the state update equations. By using the mean squared loss of the state to train the state estimation model end-to-end, the network layers can learn the computation of the Kalman gain without the necessity for externally provided ground truth for the Kalman gain during training.

However, due to the lack of guaranteed one-to-one correspondence between output trajectories and the annotated boxes in the dataset, leading to a mismatch with the true annotations based on distance matching. This flaw results in a significant number of trajectories being unsupervised during the actual training process. On the other hand, since the network parameters have not yet converged in the early stages of training, the network has not learned the correct representation of the Kalman gain, causing the positions and shapes of the trajectories to deviate significantly from the actual situation. This discrepancy leads to many trajectories not being associated with their corresponding true annotations, resulting in the slow convergence of network and limited accuracy.

To address this issue, we propose a hybrid-supervised training strategy that combines real annotations with pseudo-label generation techniques. In addition to the dataset annotations, we also use the traditional Kalman filtering to generate pseudo-labels as auxiliary supervisory signals for trajectory samples that are unassociated after the Hungarian matching. Similarly, to maintain the consistency of supervisory information, the output of motion model is used as pseudo-label supervisory information. Consequently, our final loss function is as:

$$\mathcal{L} = \sum \left\| X_n - \hat{X}_n \right\|^2 + \sum \left\| \tilde{X}_n - \hat{X}_n \right\|^2 \tag{15}$$

where X_n is the true annotation of the dataset, $\tilde{X}_n$is the update output of traditional Kalman filtering, and $\hat{X}_n$ is the update output of GRU-based Kalman filtering.

TABLE I

QUANTITATIVE COMPARISON WITH SOTAS ON *Nuscenes* TEST SET

Method	Detector	AMOTA (%)	AMOTP (m)	IDS
CBMOT [30]	CenterPoint	68.1	0.528	761
SimpleTrack [9]	CenterPoint	66.8	0.550	575
CenterPoint [11]	CenterPoint	65.0	0.535	684
Poly-MOT [10]	CenterPoint	**70.0**	0.509	331
OGR3MOT [25]	CenterPoint	65.6	0.620	288
PolarMOT [26]	CenterPoint	66.4	0.566	242
NEBP [20]	CenterPoint	68.3	0.624	**227**
ShaSTA [21]	CenterPoint	69.6	0.540	473
Gwak et. al. [27]	CenterPoint	69.8	0.540	325
Ours	CenterPoint	**70.0**	**0.501**	265

Denotations: The first four methods are based on TBD, while the last five methods are based on JDT.

TABLE II

QUANTITATIVE COMPARISON WITH SOTAS ON *Nuscenes* VAL. SET

Method	Detector	AMOTA(%)	AMOTP(m)	IDS
CBMOT [30]	CenterPoint	67.7	0.551	616
SimpleTrack [9]	CenterPoint	69.6	0.547	405
CenterPoint [11]	CenterPoint	66.5	0.567	562
Poly-MOT [10]	CenterPoint	**73.1**	0.517	232
OGR3MOT [25]	CenterPoint	69.3	0.627	262
PolarMOT [26]	CenterPoint	67.3	0.595	439
NEBP [20]	CenterPoint	70.8	-	**172**
ShaSTA [21]	CenterPoint	72.8	-	-
Gwak et. al. [27]	CenterPoint	70.3	-	-
Ours	CenterPoint	**73.1**	**0.508**	322

Denotations: The first four methods are based on TBD, while the last five methods are based on JDT. "-" indicates that the corresponding metric cannot be obtained as the results are not published in the respective paper and the code is not open-sourced.

IV. EXPERIMENTS

A. Datasets

1) NuScenes: The *NuScenes* dataset [33] consists of 850 training sequences and 150 validation sequences, with each sequence comprising approximately 40 frames sampled at a

TABLE III

HOTA COMPARISON ON *Argoverse2* VAL. SET

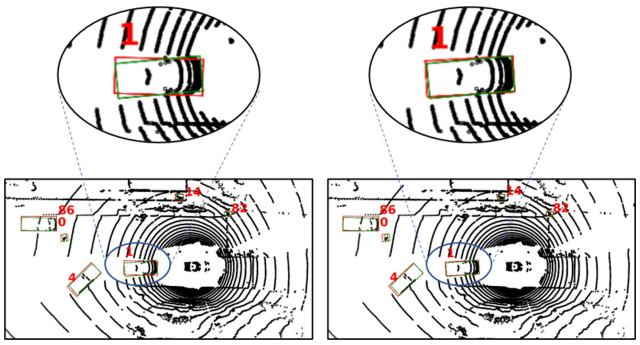
	Detector	Regular_Vehicle	Pedestrian	Bicycle	Large_Vehicle	Bus	Box_Truck	Truck	Average
Greedy [31]	LT3D	58.9	59.1	51.6	28.5	48.2	44.0	31.3	46.0
AB3DMOT [32]	LT3D	59.2	54.6	48.7	26.7	47.0	43.3	34.2	42.7
Ours	LT3D	**73.3**	**71.9**	**53.8**	**29.3**	**59.2**	**55.3**	**36.0**	**47.3**

TABLE IV

COMPARISON OF AMOTP USING DIFFERENT MOTION MODELS ON *NuScenes* VAL. SET

	Poly-MOT (Bicycle)	Ours (Bicycle)	Poly-MOT (CTRA)	Ours (CTRA)
Bicycle	0.507	**0.459**	0.461	**0.459**
Bus	**0.482**	**0.482**	**0.480**	0.482
Car	0.342	**0.337**	0.340	**0.337**
Motorbike	0.459	**0.449**	0.459	**0.448**
Pedestrian	0.362	**0.355**	0.359	**0.355**
Trailer	0.930	**0.925**	0.926	**0.925**
Truck	0.536	**0.530**	0.535	**0.529**
Average	0.517	**0.505**	0.508	**0.505**

Denotations: CTRA denotes Constant Turn Rate and Acceleration.



(a) (b)

Fig. 4. Visualization comparison of (a) Poly-MOT and (b) our method. Our object-state tracking aligns well with the ground truth, while Poly-MOT shows estimation errors for some individual objects.

rate of 2 Hz to obtain keyframes. Keyframes are annotated with detailed geometric information of object bounding boxes and their unique scene-specific identifiers. The official evaluation primarily employs the accuracy metric Average Multi-Object Tracking Accuracy (AMOTA), along with secondary metrics including Average Multi-Object Tracking Precision (AMOTP) and ID Switches (IDS).

2) Argoverse2: The *Argoverse2* dataset [34] expands upon *Argoverse1* [35] by collecting 1000 scene clips across six U.S. cities. Each sequence lasts 15 seconds and is sampled and annotated at a frequency of 10 Hz, with an average of 75 annotated objects per frame. The dataset features over 30 object classes and encompasses multiple complex urban environments. The official evaluation utilizes Higher Order Tracking Accuracy (HOTA) [36] as the key metric.

B. Implementation Details

The system input comprises the detection results from the 3D MOD. For dataset splitting, we adhere to the official division of training, validation, and test sets. In terms of training parameters, we utilize the AdamW optimizer [37] with a maximum learning rate of 10^{-5} and a weight decay of 10^{-5}. The cosine annealing learning rate adjustment method and smooth-loss supervision are employed to assist the model in avoiding local minima. Gradient accumulation is performed across each object pair per frame in the sequence, followed by backpropagation at the end of the sequence.

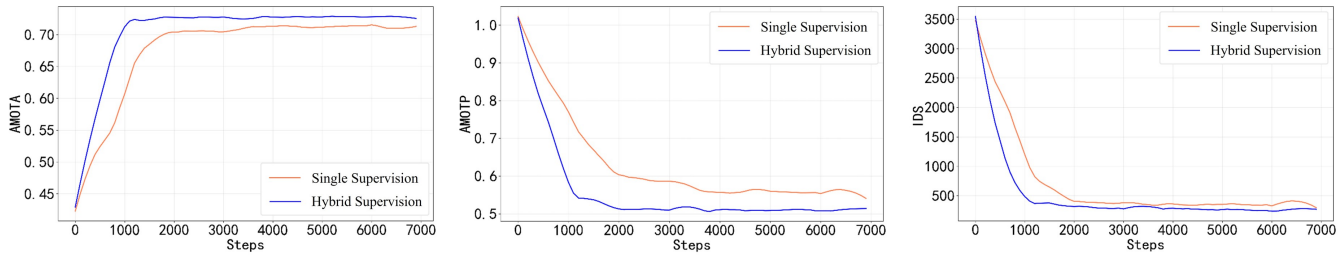
C. Experimental Results

1) Comparative Evaluation: We compare our approach with several state-of-the-art approaches on the test and validation sets of the *NuScenes* dataset. For the *Argoverse2* dataset, due to the diverse object categories it contains, and given that previous state-of-the-art approaches are defined by specific categories, it is challenging to directly apply them without modifications. Therefore, we selected LT3D [38] as the object detector (the 3D MOD officially supported by *Argoverse2*) and paired it with two classical object tracking algorithms, i.e., Greedy [31] and AB3DMOT [32], for comparison with our method.

NuScenes Test and Val. Sets: We compared our system with nine state-of-the-art 3D MOT approaches, including both TBD-based approaches (i.e., CBMOT [30], SimpleTrack [9], CenterPoint [11] and Poly-MOT [10]) and JDT-based approaches (i.e., OGR3MOT [25], PolarMOT [26], ShaSTA [21] and Gwak et. al. [27]).

Results in Table I and II demonstrate that the proposed method outperforms most state-of-the-art methods when using the same detector (i.e., CenterPoint [11]) according to the evaluation metrics of AMOTA and AMOTP. Even when compared to OGR3MOT [25] that incorporates graph neural network, our method demonstrates superior performance. The best AMOTP results indicate that our system provides more accurate final trajectory box information than traditional Kalman and constant velocity models. Furthermore, in comparison with most TBD-based and JDT-based approaches, our method exhibits excellent performance without the necessity of manually designing motion models, highlighting its significant potential for future development.

Argoverse2 Val. Set: We used the model trained on the *NuScenes* training set, and evaluate the generalization of our method on *Argoverse2* validation set. As previously mentioned, the *Argoverse2* dataset contains ground-truth labels for 30 object categories. Since most previously established state-of-the-art approaches are defined by specific categories,



(a) (b) (c)

Fig. 5. Experiments comparing the training convergence speed and evaluation metrics of single-supervised and hybrid-supervised training.

it becomes challenging to apply them directly without modifications. Therefore, we chose LT3D [38] as the object detector (a 3D MOD officially supported by *Argoverse2*) and combined it with two classic object tracking algorithms, i.e., Greedy [31] and AB3DMOT [32], for comparison with our method. It is worth mentioning that our method does not differentiate between categories, which allows for direct application on *Argoverse2* dataset. This further demonstrates the superiority and unique potential of our approach.

Results in Table III demonstrate that our method outperforms Greedy and AB3DMOT for 7 most prevalent dynamic object categories, demonstrating a strong level-generalization of our method.

2) Ablation Study: In this section, we conduct comprehensive ablation experiments to evaluate the performance of the proposed method. We utilized consistent preprocessing and association parameters, subsequently replacing the motion module and employing various training strategies for a series of experiments.

GRU-based Learnable Motion Module: The GRU-based Kalman filtering method adjusts covariance based on the object's own motion dynamics rather than relying on a strict state matrix, significantly mitigating the accuracy impact caused by mismatched state spaces. Even when different motion models are employed for a specific category, excellent tracking accuracy can still be achieved. This section verifies the influence of inaccurate motion models on traditional Kalman filtering approach represented by Poly-MOT [10] and the proposed GRU-based Kalman filtering method by modifying the motion models corresponding to each tracking category.

Experiments were conducted to evaluate the tracking accuracy of our method compared to Poly-MOT under different motion model assignments in multi-object tracking tasks. Results in Table IV demonstrate that our method can still achieve superior tracking accuracy for each category, even when employing different motion models. In contrast, when the traditional Kalman architecture alters the motion model, it results in a certain degree of variation in tracking accuracy across categories. Moreover, due to the inherent static noise definitions and linearization errors, the traditional Kalman architecture even with relatively accurate motion models, cannot achieve the same level of accuracy as the proposed GRU-based Kalman filtering method. Fig. 4 provides the visualization comparison of Poly-MOT and out method, which further demonstrates the superiority of our approach.

The proposed GRU-based Kalman filtering method successfully innovates a data-driven tracking paradigm by combining the interpretability of traditional filtering theory with the representational capabilities of deep learning. This characteristic enables the system to maintain stable tracking performance in the face of emerging target categories or complex motion patterns without the necessity for manual modeling, providing key technological support for cross-scenario generalization in autonomous driving systems.

Hybrid Supervision: This section validates the effectiveness of the proposed hybrid supervision method by comparing the training convergence process of the hybrid supervision approach with that of training using only the true labels (i.e., single supervision) from the dataset.

Results in Fig. 5 demonstrate that the proposed pseudo-label-based hybrid supervision significantly outperforms single supervision in both training efficiency and model performance. According to the convergence process, the hybrid supervision requires only 1400 iterations (approximately 2 training epochs) to reach a performance plateau, compared to 2100 iterations (3 epochs) needed by the labeled supervision method using only the true experimental results from the dataset, thus reducing training time by 33%. According to the evaluation metrics, the hybrid supervision enables a multidimensional enhancement of the model: tracking accuracy (i.e., AMOTA) increases by 2 percentage points to 73%, the estimation accuracy of motion state (i.e., AMOTP) decreases by 0.05 m to 0.505 m, and the number of ID switches (i.e., IDS) is reduced by 15%.

The pseudo-label injection increases the amount of data available for supervision by creating pseudo-labels for trajectories, thereby effectively improving the model's convergence speed. Meanwhile, it prevent mismatches between true labels and trajectories in the early stages of training, ensuring normal network training. The aforementioned factors collectively contribute to successfully achieving improvements in model convergence speed and final performance.

V. CONCLUSIONS

In this work, we propose a partially learnable MOT method by introducing a GRU-based Kalman filtering into the motion module of TBD framework. This method elimi-

nates the adverse effects of inaccurate noise parameterization and reduces the error of linearization. The proposed framework demonstrates that the GRU-based Kalman filtering is well-suited for 3D MOT task. Additionally, we introduce a hybrid-supervised training strategy that leverages pseudo-labels, which accelerates training by increasing data volume and minimizing association errors. We believe that integrating deep learning methods with interpretable mathematical models can enhance the performance of 3D MOT.

REFERENCES

[1] Z. Yuan, F. Lang, T. Xu, and X. Yang, "Sr-lio: Lidar-inertial odometry with sweep reconstruction," pp. 7862–7869, 2024.

[2] Z. Yuan, J. Deng, R. Ming, F. Lang, and X. Yang, "Sr-livo: Lidar-inertial-visual odometry and mapping with sweep reconstruction," *IEEE Robotics and Automation Letters*, vol. 9, no. 6, pp. 5110–5117, 2024.

[3] Z. Yuan, F. Lang, J. Deng, H. Luo, and X. Yang, "Voxel-svio: Stereo visual-inertial odometry based on voxel map," *IEEE Robotics and Automation Letters*, vol. 10, no. 6, pp. 6352–6359, 2025.

[4] S. Xu, Y. Wang, W. Zhang, C. P. Ho, and L. Zhu, "Observer-based distributed mpc for collaborative quadrotor-quadruped manipulation of a cable-towed load," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 4591–4597.

[5] S. Xu, L. Zhu, H.-T. Zhang, and C. P. Ho, "Robust convex model predictive control for quadruped locomotion under uncertainties," *IEEE Transactions on Robotics*, vol. 39, no. 6, pp. 4837–4854, 2023.

[6] S. Xu, L. Zhu, and C. P. Ho, "Learning efficient and robust multi-modal quadruped locomotion: A hierarchical approach," in *2022 international conference on robotics and automation (ICRA)*. IEEE, 2022, pp. 4649–4655.

[7] Z. Yuan, Q. Wang, K. Cheng, T. Hao, and X. Yang, "Sdv-loam: Semi-direct visual–lidar odometry and mapping," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 9, pp. 11 203–11 220, 2023.

[8] 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)*. IEEE, 2020, pp. 10 359–10 366.

[9] Z. Pang, Z. Li, and N. Wang, "Simpletrack: Understanding and rethinking 3d multi-object tracking," in *European Conference on Computer Vision*. Springer, 2022, pp. 680–696.

[10] X. Li, T. Xie, D. Liu, J. Gao, K. Dai, Z. Jiang, L. Zhao, and K. Wang, "Poly-mot: A polyhedral framework for 3d multi-object tracking," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 9391–9398.

[11] T. Yin, X. Zhou, and P. Krahenbuhl, "Center-based 3d object detection and tracking," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 11 784–11 793.

[12] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[13] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European conference on computer vision*. Springer, 2020, pp. 213–229.

[14] Y. Chen, J. Liu, X. Zhang, X. Qi, and J. Jia, "Voxelnext: Fully sparse voxelnet for 3d object detection and tracking," pp. 21 674–21 683, 2023.

[15] 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)*. IEEE, 2021, pp. 11 315–11 321.

[16] L. Wang, X. Zhang, W. Qin, X. Li, J. Gao, L. Yang, Z. Li, J. Li, L. Zhu, H. Wang, *et al.*, "Camo-mot: Combined appearance-motion optimization for 3d multi-object tracking with camera-lidar fusion," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 11, pp. 11 981–11 996, 2023.

[17] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[18] A. Graves and A. Graves, "Long short-term memory," *Supervised sequence labelling with recurrent neural networks*, pp. 37–45, 2012.

[19] F. Meyer, T. Kropfreiter, J. L. Williams, R. Lau, F. Hlawatsch, P. Braca, and M. Z. Win, "Message passing algorithms for scalable multitarget tracking," *Proceedings of the IEEE*, vol. 106, no. 2, pp. 221–259, 2018.

[20] M. Liang and F. Meyer, "Neural enhanced belief propagation for multiobject tracking," *IEEE Transactions on Signal Processing*, vol. 72, pp. 15–30, 2023.

[21] T. Sadjadpour, J. Li, R. Ambrus, and J. Bohg, "Shasta: Modeling shape and spatio-temporal affinities for 3d multi-object tracking," *IEEE Robotics and Automation Letters*, vol. 9, no. 5, pp. 4273–4280, 2023.

[22] T. Liang, H. Xie, K. Yu, Z. Xia, Z. Lin, Y. Wang, T. Tang, B. Wang, and Z. Tang, "Bevfusion: A simple and robust lidar-camera fusion framework," *Advances in neural information processing systems*, vol. 35, pp. 10 421–10 434, 2022.

[23] C. Zhang, C. Zhang, Y. Guo, L. Chen, and M. Happold, "Motiontrack: end-to-end transformer-based multi-object tracking with lidar-camera fusion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 151–160.

[24] 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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 1090–1099.

[25] 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.

[26] A. Kim, G. Brasó, A. Ošep, and L. Leal-Taixé, "Polarmot: How far can geometric relations take us in 3d multi-object tracking?" in *European Conference on Computer Vision*. Springer, 2022, pp. 41–58.

[27] J. Gwak, S. Savarese, and J. Bohg, "Minkowski tracker: A sparse spatio-temporal r-cnn for joint object detection and tracking," *arXiv preprint arXiv:2208.10056*, 2022.

[28] G. Revach, N. Shlezinger, X. Ni, A. L. Escoriza, R. J. Van Sloun, and Y. C. Eldar, "Kalmannet: Neural network aided kalman filtering for partially known dynamics," *IEEE Transactions on Signal Processing*, vol. 70, pp. 1532–1547, 2022.

[29] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized intersection over union: A metric and a loss for bounding box regression," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 658–666.

[30] 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)*. IEEE, 2021, pp. 8083–8090.

[31] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *IEEE transactions on pattern analysis and machine intelligence*, vol. 34, no. 7, pp. 1409–1422, 2011.

[32] X. Weng, J. Wang, D. Held, and K. Kitani, "Ab3dmot: A baseline for 3d multi-object tracking and new evaluation metrics," *CoRR*, 2020.

[33] 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.

[34] B. Wilson, W. Qi, T. Agarwal, J. Lambert, J. Singh, S. Khandelwal, B. Pan, R. Kumar, A. Hartnett, J. K. Pontes, *et al.*, "Argoverse 2: Next generation datasets for self-driving perception and forecasting," *arXiv preprint arXiv:2301.00493*, 2023.

[35] M.-F. Chang, J. W. Lambert, P. Sangkloy, J. Singh, S. Bak, A. Hartnett, D. Wang, P. Carr, S. Lucey, D. Ramanan, and J. Hays, "Argoverse: 3d tracking and forecasting with rich maps," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[36] J. Luiten, A. Osep, P. Dendorfer, P. Torr, A. Geiger, L. Leal-Taixé, and B. Leibe, "Hota: A higher order metric for evaluating multi-object tracking," *International journal of computer vision*, vol. 129, no. 2, pp. 548–578, 2021.

[37] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations*.

[38] N. Peri, A. Dave, D. Ramanan, and S. Kong, "Towards long-tailed 3d detection," in *Conference on Robot Learning*. PMLR, 2023, pp. 1904–1915.