End-to-end 3D Tracking with Decoupled Queries

Yanwei Li[1*] Zhiding Yu[2] Jonah Philion[2,3] Anima Anandkumar[2,4]
Sanja Fidler[2,3,5] Jiaya Jia[1,6] Jose Alvarez[2]

CUHK[1] NVIDIA[2] University of Toronto[3]
Caltech[4] Vector Institute[5] SmartMore[6]

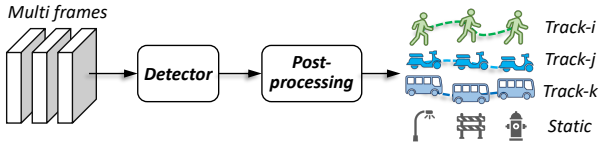
Abstract

In this work, we present an end-to-end framework for camera-based 3D multi-object tracking, called DQTrack. To avoid heuristic design in detection-based trackers, recent query-based approaches deal with identity-agnostic detection and identity-aware tracking in a single embedding. However, it brings inferior performance because of the inherent representation conflict. To address this issue, we decouple the single embedding into separated queries, i.e., object query and track query. Unlike previous detection-based and query-based methods, the decoupled-query paradigm utilizes task-specific queries and still maintains the compact pipeline without complex post-processing. Moreover, the learnable association and temporal update are designed to provide differentiable trajectory association and frame-by-frame query update, respectively. The proposed DQTrack is demonstrated to achieve consistent gains in various benchmarks, outperforming previous tracking-by-detection and learning-based methods on the nuScenes dataset.[1]

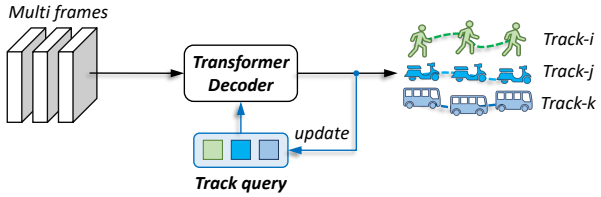
1. Introduction

Multi-object tracking (MOT) in 3D scenes is regarded as a crucial task to provide accurate object locations and identities for downstream applications, like autonomous driving and robotics. For its complex nature, previous work has typically employed pre-defined 3D detectors to localize objects and tailored post-processing to track them, *i.e.*, the tracking-by-detection pipeline. Such an approach focuses more on motion model that heavily relies on geometry cues, like Euclidean distance [38, 43] and 3D IoU [33, 26]. Although it performs well in LiDAR scenarios [38, 26], it impedes the usage of appearance features in camera-based settings for better identity distinguish.

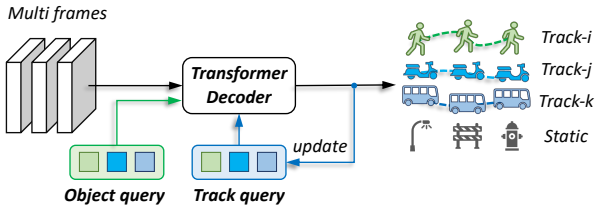
To this end, recent learning-based methods have emerged for end-to-end tracking. A typical stream is tracking-with-query [41], which employs queries in the transformer for the differentiable association. As presented in Figure 1b, current tracking-with-query approaches [24, 39, 41] usually adopt single query to solve the *identity-agnostic* detection and *identity-aware* tracking. However, it is hard to well balance the performance of detection and tracking simultaneously in 3D scenes, as experimentally validated in Table 3. Although some studies [30] attempt to use decoupled queries in 2D MOT, their methods are limited by the heavy decoder for each query and the non-differentiable association part. Therefore, a well-designed end-to-end manner is desired for accurate 3D tracking and detection.



(a) Tracking-by-detection workflow



(b) Tracking-with-query workflow



(c) Decoupled-query workflow

Figure 1: Compared with others, the proposed *decoupled-query* approach in 1c avoids the representation conflict in single query of the *tracking-with-query* method 1b and heuristic processing in *tracking-by-detection* manner 1a.

*Work done during an internship at NVIDIA.
[1]Code and demo: https://sites.google.com/view/dqtrack

In this work, we propose a simple yet effective framework for 3D object tracking, termed DQTrack. Specifically, it utilizes decoupled queries to address the task conflict representation in previous query-based approaches [24, 39, 41], as briefly illustrated in Figure 1c. With the designed task-specific queries, DQTrack enhances the query capability while maintaining a compact tracking pipeline. Moreover, unlike previous query-based work [41] in Figure 1b for movable objects only, both static and movable objects are well perceived in our proposed workflow.

Given the decoupled-query guideline, 3D object detection can be easily solved with object query and DETR-type decoder [45]. The key question is how to track objects in an end-to-end framework. We provide the solution by solving two inherent issues in tracking, that is *how to associate objects to trajectories* and *how to update track representation at each frame*. For the first problem, the learnable association is introduced to establish the differentiable relationship between objects and tracks. In particular, embedding interaction is utilized to provide sufficiently interacted object embedding for track query initialization. And query association is employed to better combine the appearance and geometry feature in track query, which enhances the robustness of the association. For the second one, we keep the representation of each track up-to-date for the next frame. Here, the appearance and geometry features of each track query are updated in a frame-by-frame manner. With the above designs, we can easily optimize the framework and perform end-to-end 3D object tracking and detection.

Generally, our decoupled-query method is distinguished from two aspects. On one hand, compared with traditional tracking-by-detection methods, it avoids the heuristic design with NMS and formulates an end-to-end framework with the learnable association. On the other hand, compared with recent tracking-with-query approaches, it overcomes the task conflict in single query embedding, thus further unleashing the potential of query-based trackers.

The overall framework, called DQTrack, can be easily instantiated with various encoders and decoders for 3D tracking and detection, as fully elaborated in Section 3. Extensive empirical studies are conducted in Section 4 to reveal the effectiveness of DQTrack. And we further compare with state-of-the-art approaches on the widely-adopted nuScenes [4] dataset. DQTrack is demonstrated to surpass all previous learning-based trackers and achieves leading performance with **52.3**% AMOTA on the nuScenes *test* set.

2. Related Work

3D Tracking. Multi-object tracking (MOT) in 3D scenes mainly takes multi-view images from surrounding cameras or point clouds from LiDAR to track multiple objects across frames. Taking advances in 3D object detection [38, 31, 8, 19, 16], recent 3D trackers usually follow the tracking-by-detection paradigm and associate trajectories with detected boxes by post-processing. In particular, object location is often utilized to provide motion cues for box association using Euclidean distance [38, 1, 19] or 3D IoU [33, 26]. Despite its popularity, the tracking-by-detection pipeline heavily relies on complex human designs. Towards a compact tracking pipeline, learning-based methods are proposed to better combine the appearance and motion model using neural networks. Specifically, the affinity matrix is computed from the appearance feature with the learnable module [9], GNN [34, 35], LSTM [14, 23], or Transformer [40, 41]. In this study, we mainly focus on end-to-end tracking in a *tracking-with-query* paradigm, which is discussed in the following sections.

Tracking by Detection. Most 3D or 2D MOT methods follow the tracking-by-detection pipeline, where tracking is treated as a post-processing step after object detection. A typical workflow [33, 26, 3, 36, 42] is to firstly find the existing objects at each frame with accurate detector [38, 44, 28]. Then, the detected objects are associated with existing trajectories with the designed measurement, like feature-based [14, 36], distance-based [38, 43], or IoU-based [33, 26] metric. Meanwhile, motion models are usually adopted to update the status of each trajectory frame by frame, *e.g.,* Kalman filter [33, 36] and velocity update [38, 43]. With the calculated affinity, Hungarian [15] or greedy [38] matching algorithm is widely used to match objects with trajectories. New-born and dead trajectories are managed with specially designed life cycle strategies [36, 2]. Different from them, our method aims to achieve end-to-end 3D tracking with the assistance of transformer, with no need for heuristic post-processing.

Tracking with Query. In light of advances in vision transformer [10, 22, 5, 45], recent MOT methods [30, 24, 39] try to model the tracking process with transformer queries. In particular, TransTrack [30] utilizes separate decoders to predict object and trajectory boxes, which are associated with the IoU matching. Although transformer queries are used to produce boxes, the core association part is still non-differentiable. To this end, TrackFormer [24] learns the association within two adjacent frames and employs track NMS to filter duplicate predictions. MOTR [39] models the relation with a longer temporal clip and predicts object trajectories directly with query update. To support camera-based 3D tracking, MUTR3D [41] extends MOTR-type tracker to the 3D domain and achieves promising results. However, as analysed before, it predicts object location and identity using the same query, which greatly harms the detection results and hinders the tracker from being more accurate. Our proposed method try to solve it with decoupled queries for better tracking and detection results.

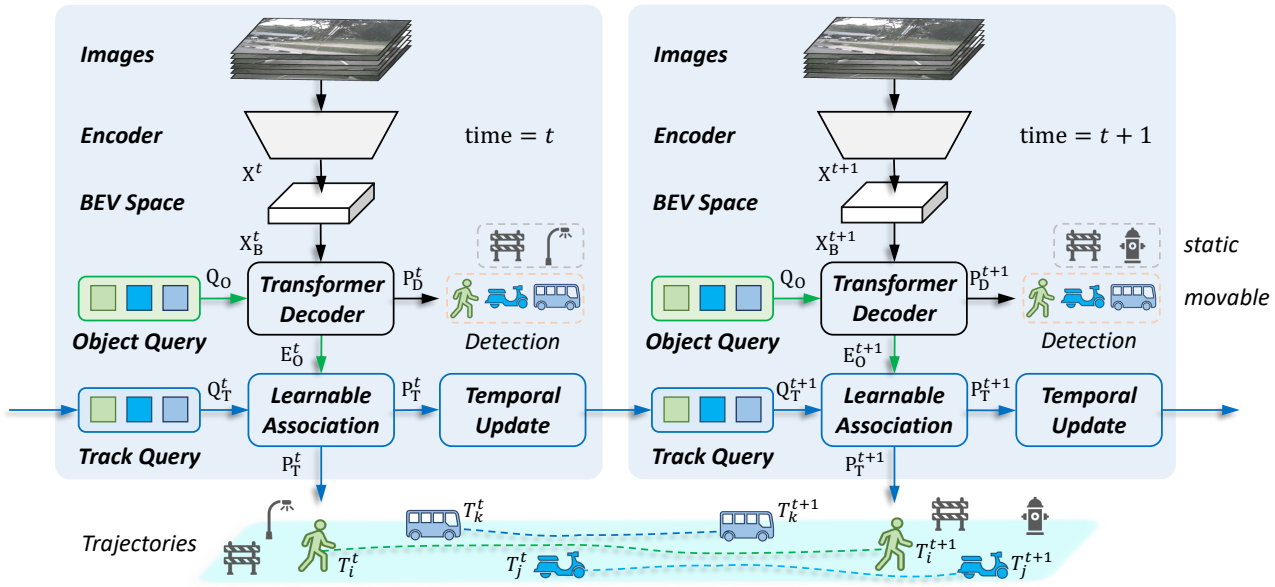


Figure 2: The framework of DQTrack for 3D object tracking. In particular, input images are first processed with the encoder and transformed to BEV space. Then, the transformer decoder takes the object query, as well as the value from BEV space, and outputs encoded object embedding. In *learnable association*, we facilitate identity awareness with embedding interaction and fuse appearance with motion embedding, which provides better affinity representation for trajectory association. And the *temporal update* is designed to refresh track query from appearance and geometry aspects for the next frame.

3. Tracking with Decoupled Queries

The framework of DQTrack is conceptually simple: encoder and transformer decoder are adopted to encode input images and provide object-aware embedding aided by object query; Learnable association utilizes tailored modules to enhance affinity representation and associate trajectories; Temporal update is designed to update the appearance and geometry embedding in each track query. In this section, we delve into the workflow and give detailed elaboration.

3.1. Encoder and Decoder

Given input captured from surrounding cameras, we first use an encoder to process the data and transform the encoded feature to bird's-eye view (BEV) space, as illustrated in Figure 2. In this procedure, several approaches can be applied to instantiate the encoder, as presented in Table 1. Motivated by [18], we utilize the stereo-based method for preliminary validation, which takes adjacent frames for depth prediction. But different from [18], here we only take images as input for simplicity, *without* using depth supervision from LiDAR. For the transformer decoder, we keep identical with previous work in 3D detection [31, 19] and adopt Deformable DETR [45] to provide interacted object embedding and positional cues. As presented in Figure 2, with the assistance from the object query $\mathbf{Q}_{\mathrm{O}}$ and BEV value $\mathbf{X}_{\mathrm{B}}^{t}$, we can easily generate the object embedding $\mathbf{E}_{\mathrm{O}}^{t} \in \mathbb{R}^{M \times C}$ from the decoder. Here, M, N, and C indicate the number of objects, trajectories, and embedding channels, respectively. And detection results P_{D}^{t} are directly produced from the object embedding $\mathbf{E}_{\mathrm{O}}^{t}$ with several feed-forward networks (FFN) *without* non-maximum suppression (NMS).

3.2. Learnable Association

With the above-generated object embedding $\mathbf{E}_{\mathrm{O}}^{t}$ at time t, the learnable association is proposed to better represent the affinity between objects and trajectories, as shown in Figure 2. Meanwhile, we update the object embedding $\mathbf{E}_{\mathrm{O}}^{t}$ with previous output $\mathbf{E}_{\mathrm{U}}^{t-1}$, as elaborated in Section 3.3.

Embedding Interaction. With the updated embedding $\mathbf{E}_{\mathrm{U}}^{t}$, embedding interaction can be easily conducted. As presented in Figure 3, the embedding $\mathbf{E}_{\mathrm{U}}^{t} \in \mathbb{R}^{M \times C}$ is first processed with separate FFN. It is designed to generate distinct embedding $\mathbf{E}_{\mathrm{UO}}^{t}$ and $\mathbf{E}_{\mathrm{UT}}^{t}$. Here, $\mathbf{E}_{\mathrm{UO}}^{t}$ is utilized to generate object embedding $\mathbf{E}_{\mathrm{A}}^{t}$ for association, while $\mathbf{E}_{\mathrm{UT}}^{t}$ is used to provide candidate track embedding $\mathbf{E}_{\mathrm{T}}^{t}$ for query update in Section 3.3. Then, they are concatenated and interacted in the self-attention module. In this way, the feature in $\mathbf{E}_{\mathrm{UO}}^{t}$ not only interacts with other objects within $\mathbf{E}_{\mathrm{UO}}^{t}$, but also establishes relation with the candidate tracks in $\mathbf{E}_{\mathrm{UT}}^{t}$. It is demonstrated to improve the trajectory association in Table 4. Finally, the interacted feature is encoded in the FFN and prepared for the subsequent query association.

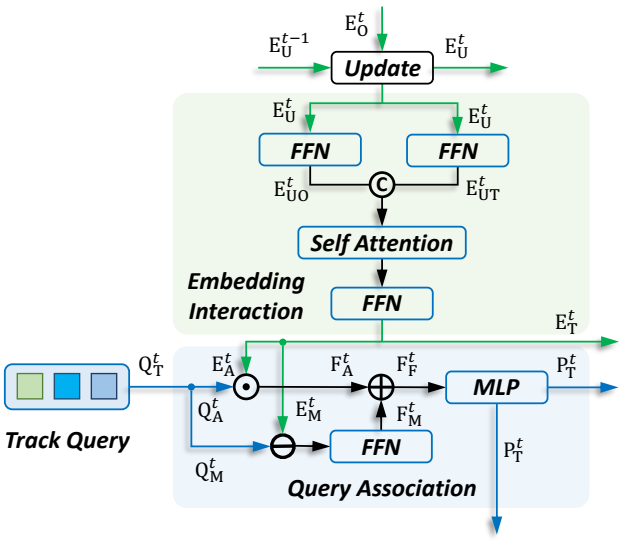


Figure 3: Details in the designed learnable association. ©, $\odot$, $\ominus$, $\oplus$ denote concatenation, Hadamard product, L2 distance, and summation, respectively.

Query Association. For the desired differentiable tracking, we further propose the query association to construct the affinity matrix. Taken the track query $\mathbf{Q}_{\mathrm{T}}^{t} \in \mathbb{R}^{N\times(C+2)}$ as input, we first split it into appearance query $\mathbf{Q}_{\mathrm{A}}^{t} \in \mathbb{R}^{N\times C}$ and motion query $\mathbf{Q}_{\mathrm{M}}^{t} \in \mathbb{R}^{N\times 2}$, as presented in Figure 3. Here, $\mathbf{Q}_{\mathrm{A}}^{t}$ represents the object appearance, while $\mathbf{Q}_{\mathrm{M}}^{t}$ indicates the trajectory motion at BEV position (x, y). We then compute the affinity feature between tracks and objects from the aspect of appearance and motion. For appearance affinity, we calculate the feature $\mathbf{F}_{\mathrm{A}}^{t} \in \mathbb{R}^{N\times M\times C}$ between $\mathbf{Q}_{\mathrm{A}}^{t}$ and $\mathbf{E}_{\mathrm{A}}^{t} \in \mathbb{R}^{M\times C}$ using the Hadamard product. As for motion affinity $\mathbf{F}_{\mathrm{M}}^{t} \in \mathbb{R}^{N\times M\times C}$, we use FFN to encode the L2 distance between each track query $\mathbf{Q}_{\mathrm{M}}^{t}$ and object embedding $\mathbf{E}_{\mathrm{M}}^{t} \in \mathbb{R}^{M\times 2}$. With these two affinity features, we directly obtain the fused feature $\mathbf{F}_{\mathrm{F}}^{t} \in \mathbb{R}^{N\times M\times C}$ by summation. Finally, we predict the affinity matrix $P_{\mathrm{T}}^{t} \in \mathbb{R}^{N\times M}$ using multi-layer perception (MLP). Overall, the entire association process can be formulated as

$$P_{\mathrm{T}}^{t} = \sigma(\mathrm{MLP}(\mathbf{Q}_{\mathrm{A}}^{t} \odot \mathbf{E}_{\mathrm{A}}^{t} + \mathrm{FFN}(\mathrm{L2}(\mathbf{Q}_{\mathrm{M}}^{t}, \mathbf{E}_{\mathrm{M}}^{t})))), \quad (1)$$

where σ indicates the softmax function to calculate affinity scores between tracks and objects.

3.3. Temporal Update

As illustrated in Figure 2, the temporal update is mainly designed to update the track query $\mathbf{Q}_{\mathrm{T}}^{t}$ for *next* timestamp $t+1$. Moreover, the object embedding $\mathbf{E}_{\mathrm{O}}^{t}$ is also updated here to better represent each object at *current* timestamp t, as shown in Figure 4.

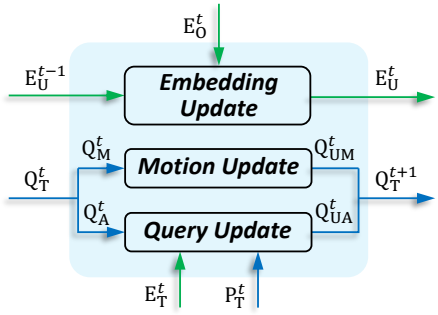


Figure 4: Details in the designed temporal update module.

Track Query. We update the track query $\mathbf{Q}_{\mathrm{T}}^{t} \in \mathbb{R}^{N\times(C+2)}$ at each timestamp t by considering both motion and appearance information, resulting in the tailored representations $\mathbf{Q}_{\mathrm{M}}^{t} \in \mathbb{R}^{N\times 2}$ and $\mathbf{Q}_{\mathrm{A}}^{t} \in \mathbb{R}^{N\times C}$. To update the object position, we directly utilize the predicted velocity $P_{\mathrm{V}}^{t} \in \mathbb{R}^{N\times 2}$ in the detection output P_{D}^{t}, where $\mathbf{Q}_{\mathrm{UM}}^{t} = \mathbf{Q}_{\mathrm{M}}^{t} + \Delta t \times P_{\mathrm{V}}^{t}$. It is much simpler than previous work [3, 33] applying Kalman filter [32] as motion model that could bring potential improvement. For the appearance aspect, we adopt exponential moving average (EMA) to update the track feature $\mathbf{Q}_{\mathrm{A}}^{t}$ frame by frame. In particular, given the candidate track embedding $\mathbf{E}_{\mathrm{T}}^{t}$ in Figure 3, we first select matched pairs according to affinity matrix P_{T}^{t} for query update. Assuming the condition $M < N$, we can easily obtain the associated track-object pair H_{T}^{t} from the affinity matrix P_{T}^{t} using Hungarian matching. And the detailed matching process is elaborated in Section 3.4. Therefore, the query update procedure can be formulated as

$$\mathbf{Q}_{\mathrm{UA}}^{t} = \alpha \times \mathbf{Q}_{\mathrm{A}}^{t} + (1-\alpha) \times \mathbf{E}_{\mathrm{T}}^{t}[H_{\mathrm{T}}^{t}]. \quad (2)$$

Here, α denotes the EMA decay rate that is investigated in Table 7. And $[\cdot]$ represents the indexing operation.

Object Embedding. For object embedding update, we take the previously updated object embedding $\mathbf{E}_{\mathrm{U}}^{t-1}$ and the generated embedding $\mathbf{E}_{\mathrm{O}}^{t}$ from transformer decoder as input, as depicted in Figure 4. Similar to that in Equation (2), the EMA strategy is also utilized to update the object embedding. Thus, the embedding update process is termed as

$$\mathbf{E}_{\mathrm{U}}^{t} = \beta \times \mathbf{E}_{\mathrm{U}}^{t-1} + (1-\beta) \times \mathbf{E}_{\mathrm{O}}^{t}, \quad (3)$$

where β denotes the EMA decay rate that is investigated in Table 9. In this way, the whole update process is easily conducted without any heuristic design.

3.4. Optimization and Inference

The proposed DQTrack maintains the end-to-end tracking pipeline with the help of above designed learnable association and temporal update. In this section, we provide more details on model optimization and inference.

Optimization Objectives. In the whole framework, there are mainly two predictions that require supervision at each frame, *i.e.,* P_{D}^t for 3D detection and P_{T}^t for tracking association. To achieve detection prediction P_{D}^t, we follow a common pipeline in DETR-based 3D detectors [31, 19] and use Hungarian matching [15] for one-to-one target assignment in the training phase. Therefore, a set-to-set approach [5] is utilized to optimize the loss $\mathcal{L}_{\mathrm{Det}}^t$, including box regression and classification. For the tracking association P_{T}^t in Equation (1), we construct the optimization target Y_{T}^t by finding the identity correspondence across frames and calculate the tracking loss, denoted as $\mathcal{L}_{\mathrm{Track}}^t = \mathrm{CE}(P_{\mathrm{T}}^t, Y_{\mathrm{T}}^t)$. Here, CE indicates the cross entropy loss function.

To enhance the robustness in occlusion scenarios, we augment track queries $\mathbf{Q}_{\mathrm{T}}^t$ during training by randomly sampling several false positives from the candidate track embedding $\mathbf{E}_{\mathrm{T}}^t$. Moreover, to balance the probability of unmatched association pairs, we draw inspiration from [12] and utilize maximum entropy regularization $\mathcal{L}_{\mathrm{Reg}}^t$ to encourage similar scores for unmatched pairs. Experiments in Table 8 validate the effectiveness of these strategies. We optimize the target with a consecutive sequence clip that includes D frames at each step, as evaluated in Table 10. Therefore, the overall optimization target is formulated as

$$\mathcal{L} = \sum_{t}^{D} (\lambda_{\mathrm{Det}} \mathcal{L}_{\mathrm{Det}}^t + \lambda_{\mathrm{Track}} \mathcal{L}_{\mathrm{Track}}^t + \lambda_{\mathrm{Reg}} \mathcal{L}_{\mathrm{Reg}}^t), \quad (4)$$

where λ_{Det}, λ_{Track}, and λ_{Reg} indicate the loss balancing weights that are all set to 1.0 by default.

Inference Scheme. In the inference stage, we follow the general end-to-end tracking pipeline with the designed components, as summarized in Algorithm 1. Unlike the training stage, object tracking is performed frame-by-frame during inference, with no need for a video clip at each step. To be specific, we first adopt encoder and decoder in Section 3.1 to generate the object embedding $\mathbf{E}_{\mathrm{O}}^t$ and predict the detection result P_{D}^t. The object embedding $\mathbf{E}_{\mathrm{O}}^t$ is then updated and selected with corresponding scores in P_{D}^t over threshold τ. With the active embedding $\mathbf{E}_{\mathrm{U}}^t$, the designed learnable association in Section 3.2 is utilized to facilitate interaction and embedding fusion for the robust tracking affinity matrix P_{T}^t. Then, pairs in P_{T}^t with scores lower than the given threshold μ are filtered out. Here, Hungarian matching [15] is adopted for the trajectory-object association. Given the associated pair H_{T}^t, we update the relevant motion $\mathbf{Q}_{\mathrm{UM}}^t$ and appearance embedding $\mathbf{Q}_{\mathrm{UA}}^t$ using the temporal update in Section 3.3. Thus, we can easily find the matched $\mathcal{T}_{match}$, new-born $\mathcal{T}_{new}$, and missed trajectories $\mathcal{T}_{miss}$ according to H_{T}^t and missing threshold δ. The tracking result $\mathcal{T}$ and the track query $\mathbf{Q}_{\mathrm{T}}$ are correspondingly updated at the end of each forward.

Algorithm 1: Pseudo code of DQTrack.

Input: Video sequence V; object score threshold τ; association score threshold μ; trajectory missing threshold δ
Output: Trajectory $\mathcal{T}$ and detection $\mathcal{D}$ of the video

Initialization: $\mathcal{T} \leftarrow \emptyset, \mathcal{D} \leftarrow \emptyset, \mathbf{Q}_{\mathrm{T}} \leftarrow \emptyset$
for V^t *in* V **do**
  /* **encoder & decoder** */
  $\mathbf{X}^t \leftarrow \mathtt{Encoder}(V^t)$
  $\mathbf{X}_{\mathrm{B}}^t \leftarrow \mathtt{BEVTrans}(\mathbf{X}^t)$
  $\mathbf{E}_{\mathrm{O}}^t \leftarrow \mathtt{Decoder}(\mathbf{Q}_{\mathrm{O}}, \mathbf{X}_{\mathrm{B}}^t)$
  $P_{\mathrm{D}}^t \leftarrow \mathtt{FFN}(\mathbf{E}_{\mathrm{O}}^t)$
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{P_{\mathrm{D}}^t\}$
  /* **embedding update** */
  $\mathbf{E}_{\mathrm{U}}^t \leftarrow \mathtt{EmbedUpdate}(\mathbf{E}_{\mathrm{U}}^{t-1}, \mathbf{E}_{\mathrm{O}}^t)$
  $\mathbf{E}_{\mathrm{U}}^t \leftarrow \mathbf{E}_{\mathrm{U}}^t[P_{\mathrm{D}}^t.score > \tau]$
  /* **learnable association** */
  $\mathbf{E}_{\mathrm{A}}^t, \mathbf{E}_{\mathrm{M}}^t, \mathbf{E}_{\mathrm{T}}^t \leftarrow \mathtt{EmbedInter}(\mathbf{E}_{\mathrm{U}}^t)$
  $\mathbf{Q}_{\mathrm{A}}^t, \mathbf{Q}_{\mathrm{M}}^t \leftarrow \mathtt{Split}(\mathbf{Q}_{\mathrm{T}})$
  $\mathbf{F}_{\mathrm{A}}^t \leftarrow \mathbf{Q}_{\mathrm{A}}^t \odot \mathbf{E}_{\mathrm{A}}^t$
  $\mathbf{F}_{\mathrm{M}}^t \leftarrow \mathtt{FFN}(\mathtt{L2}(\mathbf{Q}_{\mathrm{M}}^t, \mathbf{E}_{\mathrm{M}}^t))$
  $P_{\mathrm{T}}^t \leftarrow \mathtt{Softmax}(\mathtt{MLP}(\mathbf{F}_{\mathrm{A}}^t + \mathbf{F}_{\mathrm{M}}^t))$
  /* **temporal update** */
  $P_{\mathrm{T}}^t \leftarrow P_{\mathrm{T}}^t[P_{\mathrm{T}}^t.score > \mu]$
  $H_{\mathrm{T}}^t \leftarrow \mathtt{Hungarian}(P_{\mathrm{T}}^t)$
  $\mathbf{Q}_{\mathrm{UM}}^t \leftarrow \mathtt{MotionUpdate}(\mathbf{Q}_{\mathrm{M}}^t)$
  $\mathbf{Q}_{\mathrm{UA}}^t \leftarrow \mathtt{QueryUpdate}(\mathbf{Q}_{\mathrm{A}}^t, \mathbf{E}_{\mathrm{T}}^t, H_{\mathrm{T}}^t)$
  $\mathbf{Q}_{\mathrm{T}}^{t+1} \leftarrow \mathtt{Cat}(\mathbf{Q}_{\mathrm{UA}}^t, \mathbf{Q}_{\mathrm{UM}}^t)$
  /* **trajectory update** */
  $\mathcal{T}_{match} \leftarrow \mathtt{FindMatch}(\mathcal{T}, H_{\mathrm{T}}^t)$
  $\mathcal{T}_{new} \leftarrow \mathtt{FindNew}(\mathcal{T}, H_{\mathrm{T}}^t, P_{\mathrm{D}}^t)$
  $\mathcal{T}_{miss} \leftarrow \mathtt{FindMiss}(\mathcal{T}, H_{\mathrm{T}}^t, \delta)$
  $\mathcal{T} \leftarrow \mathcal{T} \cup \mathcal{T}_{new} \setminus \mathcal{T}_{miss}$
  $\mathbf{Q}_{\mathrm{T}} \leftarrow \mathtt{Init}(\mathbf{E}_{\mathrm{T}}^t, \mathcal{T}_{new}) \cup \mathtt{Pop}(\mathbf{Q}_{\mathrm{T}}^{t+1}, \mathcal{T}_{miss})$
end
Return: $\mathcal{T}, \mathcal{D}$

4. Experiments

In this section, the detailed experimental setup is first introduced. Then, we demonstrate the effectiveness of DQTrack and compare it with leading approaches. The analyses of each component are presented in the end.

4.1. Experimental Setup

Dataset. The nuScenes [4] dataset is a popular benchmark for large-scale autonomous driving. It is widely adopted for 3D object detection and multi-object tracking. It comprises 700, 150, 150 scenes for *training*, *validation*, and *testing*, respectively. The dataset provides synchronized data from 6 surrounding cameras capturing 10 object categories for detection and 7 movable classes for tracking in a 360-degree field of view at a frequency of 12Hz.

Implementation Details. In this study, we assess the generalization of DQTrack through experiments using different

Table 1: Comparisons with previous methods on the nuScenes *val* set. We report results with traditional work [31, 19, 21] and the simple stereo-based decoder. TBD and Geo represent the tracking-by-detection manner and the geometry-based matching in [38]. FPS is evaluated in a single NVIDIA A100 GPU from input images to tracking results. * denotes results from [41].

Tracker	Encoder	Decoder	Resolution	Tracking				Detection	
				AMOTA	**AMOTP↓**	**IDS↓**	**FPS**	**NDS**	**mAP**
TBD-Kalman*	R101	DETR3D	900 × 1600	26.3%	1.569	4698	–	43.4%	34.9%
SimpleTrack* [26]	R101	DETR3D	900 × 1600	29.3%	**1.307**	1695	–	43.4%	34.9%
MUTR3D [41]	R101	DETR3D	900 × 1600	29.4%	1.498	3822	6.0	43.4%	34.9%
TBD-Geo	R101	DETR3D	900 × 1600	32.6%	1.424	2564	5.1	43.4%	34.9%
DQTrack	R101	DETR3D	900 × 1600	**36.7%**	1.351	**1120**	4.8	43.4%	34.9%
TBD-Geo	R101	UVTR-C	900 × 1600	36.6%	1.386	2708	4.5	44.1%	34.2%
DQTrack	R101	UVTR-C	900 × 1600	**39.6%**	**1.310**	**1267**	3.9	44.1%	34.2%
TBD-Geo	R101	Stereo	512 × 1408	36.2%	1.368	2169	3.1	47.1%	36.1%
DQTrack	R101	Stereo	512 × 1408	**40.7%**	**1.318**	**1003**	2.8	47.1%	36.1%
TBD-Geo	V2-99	PETRV2	320 × 800	40.8%	1.309	2487	7.7	50.3%	41.0%
DQTrack	V2-99	PETRV2	320 × 800	**44.6%**	**1.251**	**1193**	6.0	50.3%	41.0%

encoders and decoders. In all the ablation studies, we adopt the stereo-based approach, where the number M of object query $\mathbf{Q}_{\mathrm{O}}$ is set to 300. Meanwhile, we employ the AdamW optimizer to train the framework in an end-to-end manner for 24 epochs, with an initial learning rate $2e^{-4}$. For other decoders [31, 19, 21], we fix pre-trained models following original settings and optimize the proposed module for 5 epochs with an initial learning rate $1e^{-5}$. By default, the object threshold τ, association threshold μ, and trajectory missing threshold δ in Algorithm 1 are set to 0.2, 0.1, and 7, respectively. And τ is set to 0.3 for PETRV2 [21].

4.2. Main Results

Results with Different Trackers. To demonstrate the effectiveness of the decoupled-query approach, we carry out experiments with different trackers in Table 1. For fair comparisons, we adopt the same encoder and input resolution for all methods. We first set up a strong benchmark using the tracking-by-detection pipeline and the tailored geometry-based matching in [38], marked as TBD-Geo. As shown in Table 1, the strong benchmark outperforms previous work that uses the same decoder DETR3D [31]. Compared with the strong benchmark, the proposed DQTrack achieves a significant improvement of **4.1**% AMOTA. And the identity switch (IDS) is reduced to **43.7**% from 2564 to 1120. When compared with the previous query-based tracker MUTR3D [41], the performance gap is further enlarged to **7.3**% AMOTA, with IDS reduced to **29.3**%.

Results with Different Decoders. In Table 1, we validate the generality of DQTrack by applying different decoders. For simplicity, we optimize the proposed modules on the pre-trained models [31, 19, 21]. Compared with the strong benchmark TBD-Geo, DQTrack achieves consistent gains with more than **3.0**% AMOTA using different encoders, decoders, and input resolutions. In particular, DQTrack attains greater improvement with a stronger encoder ResNet-101 than ResNet-50, using the same stereo-based decoder. This proves that the capability of the proposed approach can be scaled up with stronger backbones. Equipped with PETRV2 and V2-99 from DD3D [27], the proposed framework achieves 44.6% AMOTA on the nuScenes *val* set.

Comparisons with State-of-the-arts. We present comparisons with state-of-the-art approaches on the nuScenes *test* set in Table 2. Following previous work [19, 20, 25], we adopt the V2-99 as encoder and build the DQTrack upon the PETRV2 [21] decoder. Unlike classic tracking-by-detection methods, DQTrack maintains an end-to-end tracking pipeline without heuristic post-processing and achieves leading performance with **52.3**% AMOTA. We report results without bells-and-whistles like NMS or test-time augmentation. When compared with similar learning-based approaches, DQTrack surpasses previous and even concurrent study PF-Track [25] by a large margin of **8.9**% AMOTA.

4.3. Component-wise Analysis

In this subsection, we jointly optimize tracking and detection with the input image resized to 256×704, where the stereo-based framework with ResNet-50 [13] is adopted.

Decoupled Queries. The decoupled-query paradigm for end-to-end 3D MOT is the core design. In Table 3, we validate it by comparing against the tracking-by-detection [38] and the single-query manner [41], where the same encoder and thresholds are adopted. As analysed before, the single-query approach suffers from task conflict and thus performs

Table 2: Comparisons with leading camera-based methods on the nuScenes *test* set. We evaluate our model in an end-to-end manner *without* bells-and-whistles like NMS or test-time augmentation, which could bring potential improvement.

Method	Encoder	NMS	**AMOTA**	**AMOTP**↓	MOTA	MOTP↓	RECALL	IDS↓
Tracking-by-detection								
PolarDETR [7]	V2-99	✓	27.3%	1.185	23.8%	0.719	40.4%	2170
QTrack [37]	V2-99	✓	48.0%	1.100	43.1%	0.597	58.3%	1484
Sparse4D [20]	V2-99	✗	51.9%	1.078	45.9%	0.622	63.3%	1090
UVTR-Greedy [19]	V2-99	✓	51.9%	1.125	44.7%	0.650	59.9%	2204
Learning-based tracking								
CenterTrack [43]	DLA-34	✗	4.6%	1.543	4.3%	0.753	23.3%	3807
DEFT [6]	DLA-34	✗	17.7%	1.564	15.6%	0.770	33.8%	6901
Time3D [17]	DLA-34	✗	21.4%	1.360	17.3%	0.750	–	–
QD3DT [14]	R101	✓	21.7%	1.550	19.8%	0.773	37.5%	6856
MUTR3D [41]	R101	✗	27.0%	1.494	24.5%	0.709	41.1%	6018
SRCN3D [29]	V2-99	✗	39.8%	1.317	35.9%	0.709	53.8%	4090
CC-3DT [11]	R101	✓	41.0%	1.274	35.7%	0.676	53.8%	3334
PF-Track [25]	V2-99	✗	43.4%	1.252	37.8%	0.674	53.8%	**249**
DQTrack	V2-99	✗	**52.3%**	**1.096**	**44.4%**	**0.649**	**62.2%**	1204

Table 3: Results with different trackers on the nuScenes *val* set. TBD denotes the tracking-by-detection method in [38]. SinQuery indicates the single-query approach in [41].

tracker	*e2e*	Tracking		Detection (7 cls)	
		AMOTA	**AMOTP**↓	**NDS**	**mAP**
TBD	✗	25.7%	1.563	40.8%	27.5%
SinQuery	✓	24.5%	1.603	40.7%	20.1%
DeQuery	✓	**28.5%**	**1.465**	**40.8%**	**27.5%**

Table 4: Effect of embedding interaction on the nuScenes *val* set. *ei* denotes the embedding interaction in Section 3.2.

ei	Tracking		Detection	
	AMOTA	**AMOTP**↓	**NDS**	**mAP**
✗	27.8%	1.467	41.2%	28.8%
✓	**28.5%**	**1.465**	**41.3%**	**28.9%**

Table 5: Results with different embedding for query association on the nuScenes *val* set. *appear* and *motion* indicate appearance $\mathbf{F}_{\mathrm{A}}$ and motion feature $\mathbf{F}_{\mathrm{M}}$ in Section 3.2.

appear	*motion*	Tracking		Detection	
		AMOTA	**AMOTP**↓	**NDS**	**mAP**
✓	✗	25.3%	1.509	40.8%	28.5%
✗	✓	26.9%	1.480	40.5%	27.8%
✓	✓	**28.5%**	**1.465**	**41.3%**	**28.9%**

Table 6: Results with different features for embedding $\mathbf{E}_{\mathrm{O}}$ on the nuScenes *val* set. *de* denotes the transformer decoder.

de	Tracking		Detection	
	AMOTA	**AMOTP**↓	**NDS**	**mAP**
✗	26.4%	1.543	39.9%	28.1%
✓	**28.5%**	**1.465**	**41.3%**	**28.9%**

worse than the tracking-by-detection method. DQTrack is proven to achieve significant gain, which surpasses the single-query method with **4.0**% AMOTA and **7.4**% mAP.

Embedding Interaction. In Table 4, we investigate the effect of embedding interaction proposed in Section 3.2. It is clear that the designed embedding interaction facilitates the object-level information exchange for better trajectory association and leads to a 0.7% AMOTA gain in tracking.

Embedding for Association. In Section 3.2, the affinity score P_{T}^{t} is calculated from object appearance and motion with $\mathbf{F}_{\mathrm{A}}$ and $\mathbf{F}_{\mathrm{M}}$, as depicted in Figure 3 and Equation (1). Here, we evaluate the effect of embedding type in Table 5. Compared with appearance $\mathbf{F}_{\mathrm{A}}$ or motion feature $\mathbf{F}_{\mathrm{M}}$ only, the fused embedding $\mathbf{F}_{\mathrm{F}}$ performs much better, which attains 28.5% AMOTA and 41.3% NDS. Although motion features are essential, the appearance feature plays an important role and contributes 1.6% AMOTA and 1.1% mAP.

Feature for Object Embedding. In Section 3.1, we take the object embedding $\mathbf{E}_{\mathrm{O}}$ from transformer decoder. Here, we delve into the feature type for $\mathbf{E}_{\mathrm{O}}$ generation in Table 6.

Table 7: Results with different rates for query update on the nuScenes *val* set. α denotes the rate in Equation (2).

α	Tracking		Detection	
	AMOTA	**AMOTP**↓	**NDS**	**mAP**
–	27.4%	1.473	40.9%	28.6%
0.4	28.5%	1.466	41.2%	28.9%
0.5	**28.5%**	**1.465**	**41.3%**	**28.9%**
0.6	27.5%	1.477	40.7%	28.1%

Table 8: Results with different optimization strategies. *aug* and *reg* indicate query augmentation and entropy regulation.

aug	*reg*	Tracking		Detection	
		AMOTA	**AMOTP**↓	**NDS**	**mAP**
✓	✗	27.9%	1.481	40.5%	28.4%
✗	✓	28.1%	1.478	40.8%	28.5%
✓	✓	**28.5%**	**1.465**	**41.3%**	**28.9%**

Table 9: Results with different rates for embedding update on the nuScenes *val* set. β denotes the rate in Equation (3).

β	Tracking		Detection	
	AMOTA	**AMOTP**↓	**NDS**	**mAP**
–	26.9%	1.491	40.4%	27.7%
0.2	27.8%	**1.463**	40.4%	28.4%
0.3	**28.5%**	1.465	**41.3%**	**28.9%**
0.4	27.6%	1.467	41.3%	28.7%

Table 10: Results with different frames for training on the nuScenes *val* set. D denotes frame number in Equation (4).

D	Tracking		Detection	
	AMOTA	**AMOTP**↓	**NDS**	**mAP**
2	26.8%	1.465	40.5%	28.7%
3	**28.5%**	**1.465**	**41.3%**	**28.9%**
4	27.0%	1.474	40.6%	28.3%

Compared with the raw feature directly from BEV space $\mathbf{X}_{\mathrm{B}}$, the interacted embedding from transformer decoder for $\mathbf{E}_{\mathrm{O}}$ contributes 2.1% AMOTA and 1.4% NDS gain.

Query Update Rate. In Section 3.3, we present the EMA strategy for updating the track query $\mathbf{Q}_{\mathrm{T}}$ in a frame-by-frame manner, as shown in Equation (2). The EMA decay rate α is used to balance the representation of preserved and newly associated objects. We compare the performance of different decay rates in Table 7. The model achieves the best result when α is set to 0.5. Compared to the result that does not use temporal update in the first row, the proposed module achieves the desired target with 1.1% AMOTA gain.

Optimization Strategy. For robustness in complex scenarios, we simulate occluded tracks during training by adding false positives from randomly sampled queries with a probability of 0.2. We mark it as track query augmentation in Table 8. It is clear that the designed augmentation brings 0.4% AMOTA and 0.5% NDS gain. Additionally, we adopt an entropy regulation loss $\mathcal{L}_{\mathrm{Reg}}$ in Equation (4) to encourage unpaired samples with a similar score. It restrains the association with low confidence and highlights high-quality pairs, which contributes 0.6% AMOTA and 0.8% NDS.

Embedding Update Rate. To better represent objects for association, we update the object embedding $\mathbf{E}_{\mathrm{O}}$ at each timestamp, as depicted in Figures 3 and 4. In Table 9, we present the results with different EMA decay rates β. DQTrack performs best with a decay rate of 0.3, indicating that it relies more on the newly generated embedding. Similar to query update, the EMA strategy achieves significant gain of 1.6% AMOTA and 1.2% mAP over the baseline in the first row. Notably, the embedding update module is only used in the stereo-based method, but it could be further applied to improve the pre-trained models [31, 19, 21] in Table 1.

Frame Number for Training. Recall from Equations (1) and (4), multiple frames are necessary to establish identity correspondence and optimize the affinity score P_{T}^{t} during training. In Table 10, we compare the performance trained with different numbers of frames D. Experiments show that the model achieves the best performance when trained with 3 frames, while others significantly reduce the result.

5. Conclusion

We introduced DQTrack, a simple yet effective framework for camera-based 3D MOT. The core idea behinds DQTrack is to separate object and trajectory representation using decoupled queries, allowing for more accurate end-to-end 3D tracking. In particular, we propose the learnable association to establish the relationship between objects and tracks. Meanwhile, temporal update is designed to provide updated track query in each frame from appearance and geometry aspects. Experiments on the nuScenes dataset prove the superiority of DQTrack with consistent gains and outperform all previous trackers with leading performance.

6. Acknowledgement

Jiaya Jia was supported in part by the Research Grants Council under the Areas of Excellence scheme grant AoE/E-601/22-R.

References

[1] Xuyang Bai, Zeyu Hu, Xinge Zhu, Qingqiu Huang, Yilun Chen, Hongbo Fu, and Chiew-Lan Tai. Transfusion: Robust lidar-camera fusion for 3d object detection with transformers. In *CVPR*, 2022.

[2] Nuri Benbarka, Jona Schröder, and Andreas Zell. Score refinement for confidence-based 3d multi-object tracking. In *IROS*, 2021.

[3] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *ICIP*, 2016.

[4] 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 *CVPR*, 2020.

[5] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020.

[6] Mohamed Chaabane, Peter Zhang, J Ross Beveridge, and Stephen O'Hara. Deft: Detection embeddings for tracking. In *CVPRW*, 2021.

[7] Shaoyu Chen, Xinggang Wang, Tianheng Cheng, Qian Zhang, Chang Huang, and Wenyu Liu. Polar parametrization for vision-based surround-view 3d detection. *arXiv:2206.10965*, 2022.

[8] Yukang Chen, Yanwei Li, Xiangyu Zhang, Jian Sun, and Jiaya Jia. Focal sparse convolutional networks for 3d object detection. In *CVPR*, 2022.

[9] Hsu-kuang Chiu, Jie Li, Rareş Ambruş, and Jeannette Bohg. Probabilistic 3d multi-modal, multi-object tracking for autonomous driving. In *ICRA*, 2021.

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

[11] Tobias Fischer, Yung-Hsu Yang, Suryansh Kumar, Min Sun, and Fisher Yu. Cc-3dt: Panoramic 3d object tracking via cross-camera fusion. *arXiv:2212.01247*, 2022.

[12] Yang Fu, Sifei Liu, Umar Iqbal, Shalini De Mello, Humphrey Shi, and Jan Kautz. Learning to track instances without video annotations. In *CVPR*, 2021.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[14] Hou-Ning Hu, Yung-Hsu Yang, Tobias Fischer, Trevor Darrell, Fisher Yu, and Min Sun. Monocular quasi-dense 3d object tracking. *TPAMI*, 2022.

[15] Harold W Kuhn. The hungarian method for the assignment problem. *NRL*, 2005.

[16] Xin Lai, Yukang Chen, Fanbin Lu, Jianhui Liu, and Jiaya Jia. Spherical transformer for lidar-based 3d recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17545–17555, 2023.

[17] Peixuan Li and Jieyu Jin. Time3d: End-to-end joint monocular 3d object detection and tracking for autonomous driving. In *CVPR*, 2022.

[18] Yinhao Li, Han Bao, Zheng Ge, Jinrong Yang, Jianjian Sun, and Zeming Li. Bevstereo: Enhancing depth estimation in multi-view 3d object detection with dynamic temporal stereo. *arXiv:2209.10248*, 2022.

[19] Yanwei Li, Yilun Chen, Xiaojuan Qi, Zeming Li, Jian Sun, and Jiaya Jia. Unifying voxel-based representation with transformer for 3d object detection. In *NeurIPS*, 2022.

[20] Xuewu Lin, Tianwei Lin, Zixiang Pei, Lichao Huang, and Zhizhong Su. Sparse4d: Multi-view 3d object detection with sparse spatial-temporal fusion. *arXiv:2211.10581*, 2022.

[21] Yingfei Liu, Junjie Yan, Fan Jia, Shuailin Li, Qi Gao, Tiancai Wang, Xiangyu Zhang, and Jian Sun. Petrv2: A unified framework for 3d perception from multi-camera images. *arXiv:2206.01256*, 2022.

[22] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021.

[23] Nicola Marinello, Marc Proesmans, and Luc Van Gool. Triplettrack: 3d object tracking using triplet embeddings and lstm. In *CVPR*, 2022.

[24] Tim Meinhardt, Alexander Kirillov, Laura Leal-Taixe, and Christoph Feichtenhofer. Trackformer: Multi-object tracking with transformers. In *CVPR*, 2022.

[25] Ziqi Pang, Jie Li, Pavel Tokmakov, Dian Chen, Sergey Zagoruyko, and Yu-Xiong Wang. Standing between past and future: Spatio-temporal modeling for multi-camera 3d multi-object tracking. *arXiv:2302.03802*, 2023.

[26] Ziqi Pang, Zhichao Li, and Naiyan Wang. Simpletrack: Understanding and rethinking 3d multi-object tracking. *arXiv:2111.09621*, 2021.

[27] Dennis Park, Rares Ambrus, Vitor Guizilini, Jie Li, and Adrien Gaidon. Is pseudo-lidar needed for monocular 3d object detection? In *ICCV*, 2021.

[28] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015.

[29] Yining Shi, Jingyan Shen, Yifan Sun, Yunlong Wang, Jiaxin Li, Shiqi Sun, Kun Jiang, and Diange Yang. Srcn3d: Sparse r-cnn 3d surround-view camera object detection and tracking for autonomous driving. *arXiv:2206.14451*, 2022.

[30] Peize Sun, Jinkun Cao, Yi Jiang, Rufeng Zhang, Enze Xie, Zehuan Yuan, Changhu Wang, and Ping Luo. Transtrack: Multiple object tracking with transformer. *arXiv:2012.15460*, 2020.

[31] Yue Wang, Vitor Campagnolo Guizilini, Tianyuan Zhang, Yilun Wang, Hang Zhao, and Justin Solomon. Detr3d: 3d object detection from multi-view images via 3d-to-2d queries. In *CoRL*, 2022.

[32] Greg Welch, Gary Bishop, et al. An introduction to the kalman filter. 1995.

[33] Xinshuo Weng, Jianren Wang, David Held, and Kris Kitani. 3d multi-object tracking: A baseline and new evaluation metrics. In *IROS*, 2020.

[34] Xinshuo Weng, Yongxin Wang, Yunze Man, and Kris M Kitani. Gnn3dmot: Graph neural network for 3d multi-object tracking with 2d-3d multi-feature learning. In *CVPR*, 2020.

[35] Xinshuo Weng, Ye Yuan, and Kris Kitani. Ptp: Parallelized tracking and prediction with graph neural networks and diversity sampling. *RA-L*, 2021.

[36] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *ICIP*, 2017.

[37] Jinrong Yang, En Yu, Zeming Li, Xiaoping Li, and Wenbing Tao. Quality matters: Embracing quality clues for robust 3d multi-object tracking. *arXiv:2208.10976*, 2022.

[38] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Center-based 3d object detection and tracking. In *CVPR*, 2021.

[39] Fangao Zeng, Bin Dong, Yuang Zhang, Tiancai Wang, Xiangyu Zhang, and Yichen Wei. Motr: End-to-end multiple-object tracking with transformer. In *ECCV*, 2022.

[40] Yihan Zeng, Chao Ma, Ming Zhu, Zhiming Fan, and Xiaokang Yang. Cross-modal 3d object detection and tracking for auto-driving. In *IROS*, 2021.

[41] Tianyuan Zhang, Xuanyao Chen, Yue Wang, Yilun Wang, and Hang Zhao. Mutr3d: A multi-camera tracking framework via 3d-to-2d queries. In *CVPR*, 2022.

[42] Yifu Zhang, Peize Sun, Yi Jiang, Dongdong Yu, Fucheng Weng, Zehuan Yuan, Ping Luo, Wenyu Liu, and Xinggang Wang. Bytetrack: Multi-object tracking by associating every detection box. In *ECCV*, 2022.

[43] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Tracking objects as points. In *ECCV*, 2020.

[44] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv:1904.07850*, 2019.

[45] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *ICLR*, 2021.