STAR-TRACK: Latent Motion Models for End-to-End 3D Object Tracking with Adaptive Spatio-Temporal Appearance Representations

Simon Doll[1,2], Niklas Hanselmann[1,2], Lukas Schneider[1], Richard Schulz[1],
Markus Enzweiler[3] and Hendrik P.A. Lensch[2]

[1]Mercedes-Benz AG
[2]University of Tübingen
[3]Esslingen University of Applied Sciences

Abstract

Following the tracking-by-attention paradigm, this paper introduces an object-centric, transformer-based framework for tracking in 3D. Traditional model-based tracking approaches incorporate the geometric effect of object- and ego motion between frames with a geometric motion model. Inspired by this, we propose STAR-TRACK *which uses a novel latent motion model (*LMM*) to additionally adjust object queries to account for changes in viewing direction and lighting conditions directly in the latent space, while still modeling the geometric motion explicitly. Combined with a novel learnable track embedding that aids in modeling the existence probability of tracks, this results in a generic tracking framework that can be integrated with any query-based detector. Extensive experiments on the nuScenes benchmark demonstrate the benefits of our approach, showing state-of-the-art performance for DETR3D-based trackers while drastically reducing the number of identity switches of tracks at the same time. Project page:* *https://simondoll.github.io/S.T.A.R.-Track/*

1. Introduction

Robust perception and tracking of movable objects in the environment form the basis for safe decision-making in autonomous agents such as self-driving cars. Classical *multi-object tracking* (MOT) pipelines typically follow a *tracking-by-detection* paradigm, using powerful object detectors coupled with greedy matching [21] and state estimators [1, 10] to track objects in time. Building on recent advances in 3D object detection from multi-view camera imagery, transformer-based architectures [44, 8, 22, 21] can yield strong tracking performance [33, 21, 2] using relatively low-cost sensors. However, decoupling the detection and tracking tasks comes with two main drawbacks: (1) the object detection model is optimized towards a detection metric, rather than directly optimizing for the downstream tracking performance, which is prone to compounding errors [18, 13] and (2) it makes it non-trivial to incorporate appearance information, which poses a challenge to consistent association. This in particular can lead to difficulties in handling confusion among object identities in crowded scenarios with many partial object-to-object occlusions [31].

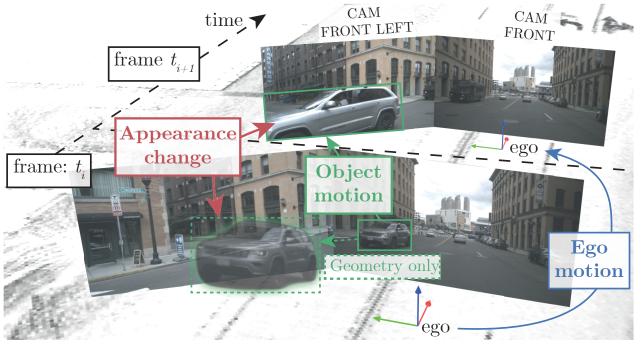

Figure 1. Visualization of a tracked object for two consecutive frames. Due to ego and object motion the object's 3D pose and its appearance in the individual camera images change in scale, viewing angle and lighting condition. We utilize an explicit geometric and a novel latent motion model to compensate for these effects during the prediction step of the tracking pipeline.

Recent work [31, 47] proposes an alternative *tracking-by-attention* paradigm that unifies perception and tracking into a single end-to-end module. Under this paradigm, the rich geometric and semantic information contained in the high-dimensional object queries of query-based detectors can be leveraged for the association of object instances across time via the attention mechanism [42]. As an additional advantage, tracking-by-attention allows for exploiting these queries as priors for detection in the following frames. This requires adjusting them to the expected future object state, analogous to the model-based prediction step in classical state estimator-based trackers [36]. In the case of purely geometric features, this can be done in a straightforward fashion by simply applying the transform corresponding to both ego and estimated object motion. However, this is not possible for latent object queries, as they also encode semantics and appearance in addition to geometric information. MUTR3D [47] sidesteps this issue by anchoring object queries to geometric reference points which can be analytically updated. While this enables some adjustment, only the object translation rather than the full pose is considered and the change in appearance resulting from changes in the relative pose is not modeled. In [39], the authors propose a LiDAR-based tracking method that corrects both geometric and appearance information directly in latent space via a hyper-network [14] to compensate for ego motion. However, this approach forfeits the ability to analytically update geometric information and does not model object motion.

In this paper, we propose to compensate for appearance changes resulting from both ego and object motion via a novel *latent motion model* (LMM) which updates queries in latent space as a function of the geometric motion transform. Paired with analytical updates on geometric reference points for each query, we obtain a transformable **S**patio-**T**emporal geometry and **A**ppearance **R**epresentation for each object that enhances consistency with future observations. Furthermore, we propose learned track embeddings that encode information on the track's lifetime to distinguish tracks from new detections. The resulting tracking framework, termed STAR-TRACK, exhibits improved tracking performance. Specifically, we observe that accounting for appearance changes between frames as well as the improved existence probability modeling eases association, leading to a drastically reduced number of switches in object instance identities.

In summary, we make the following main contributions:

- We propose a *latent motion model* (LMM) to model the appearance change of an object given a geometric transformation encoding ego and object motion.
- We introduce track embeddings to allow for latent, attention-based existence probability modeling.
- The resulting general extension to query-based 3D detectors for tracking achieves *state-of-the-art* (SOTA) tracking performance on the nuScenes [3] dataset for DETR3D [44]-based methods.

2. Related Work

Query-based Detection: MOT approaches that follow the tracking-by-detection [21, 1, 10] paradigm require a detector to detect a set of objects in each frame. The pioneering work DETR [4] proposed a way to leverage the transformer architecture for object detection. In contrast to previous approaches, this set-based architecture comes with various desirable properties such as a sparse prediction scheme due to bi-partite matching, a dynamic amount of object hypotheses, and no need for hand-crafted components such as *non-maximum suppression* (NMS) or dense object anchors [37]. Additionally, several refinements to the original DETR architecture have been proposed to improve performance and convergence speed [49, 5, 16]. Furthermore, the concept was generalized to the 3D case as well as to different sensor modalities including LiDAR [2, 9], multi-view camera [44, 8, 22] and multi-modal detection methods [2, 28, 21]. It is noteworthy that such query-based detectors became the de-facto standard in object detection and reach SOTA performance on various benchmarks such as COCO [25], KITTI [12] or nuScenes [3].

Tracking-by-Detection: Tracking methods that rely on the well-established tracking-by-detection paradigm have the benefit of being compatible with any detection framework since the detection per frame and the tracking/association part are not directly linked. A simple greedy association as proposed in [45] is still widely adopted in current SOTA methods on the nuScenes tracking benchmark [21, 28, 2, 33]. As a result of this generic approach, the detector can not make use of previous tracks and the association often relies on geometric cues only. This causes track identity switches in which a tracked object is not associated with the previous track and is reinitialized with a newborn detection instead. Various extensions such as re-ID features [34], [38] and motion models [40, 10] have been proposed to mitigate this effect. Motion models integrate prior knowledge about the physical properties and trajectory of the tracked object while re-ID features allow an association that is not solely based on bounding box geometry but also influenced by other features such as motion cues or the object's appearance.

Tracking-by-Attention: To overcome the independent nature of the detection and tracking modules in a fully differentiable fashion and to implicitly solve the association between frames, the transformer-based approaches in [31, 41] follow the *tracking-by-attention* paradigm. Leveraging the potential of the attention block, tracking and detection are performed jointly by auto-regressive query-based tracking since each detection of the last frame is used as a prior (track-query) for the next frame. The self-attention block, therefore, allows track queries to suppress double

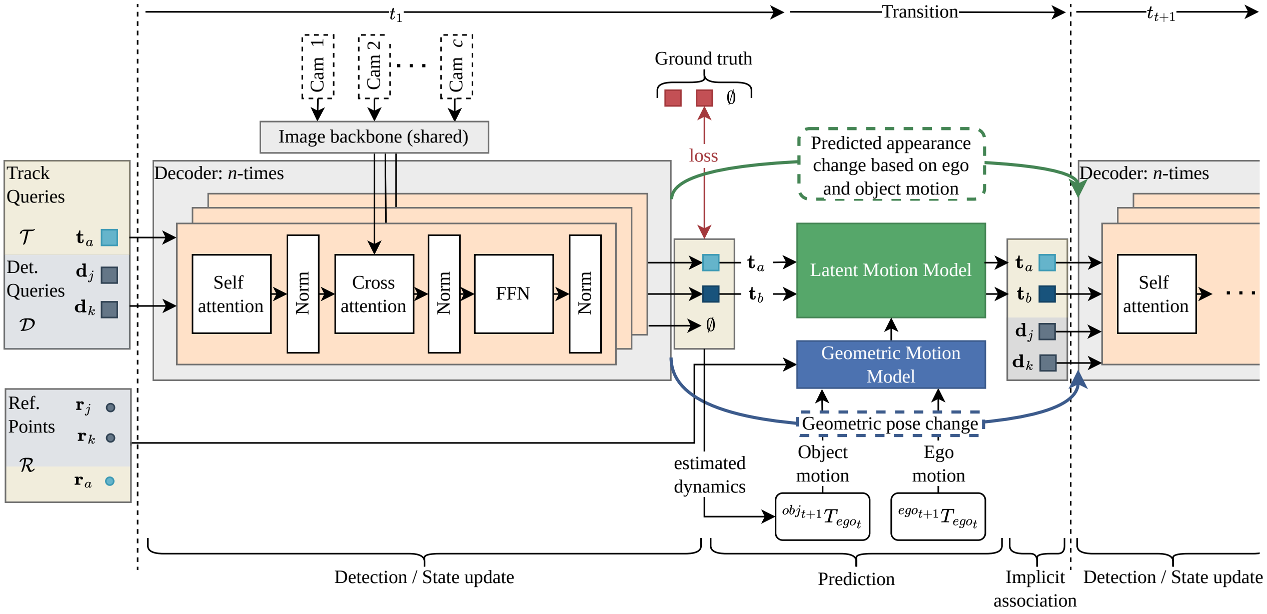

Figure 2. STAR-TRACK architecture. A joint set of time-independent object queries and track queries of the previous frames is used in a stack of decoder layers that utilize self- and cross-attention blocks to detect and re-identify objects in consecutive time steps. This requires predicting the state of each object in the following frame. Combined with any geometric motion model (blue) the newly proposed latent motion model (green) solves this issue by modeling the spatio-temporal change of a track query in the latent and the 3D geometric space jointly, based on the estimated dynamics.

detections, while still allowing to spawn newly appeared objects [31]. A tracking extension of the multi-camera 3D object detection method DETR3D [44] is proposed in MUTR3D [47], which additionally adds a geometric compensation of object and ego-motion. This is done by utilizing a 3D reference point per object that is transformed between consecutive frames while the latent query features remain unchanged. A possibility to account for the appearance change caused by the ego-motion is proposed in [39]. The proposed ego-motion-compensation module models the effect directly in the latent space as a linear function that depends on the estimated transform between the two frames. Similar to the 3D case in which the transform can be represented as a 4×4 homogeneous matrix the transform in latent space is modeled as a full-rank $k \times k$ matrix which is learned from the given ego-motion via a hyper-network [14].

Inspired by the aforementioned previous works we propose a so-called latent motion model to account for the effects of *ego- and object motion* on the latent appearance representation jointly. This allows for keeping the explicit geometric update proposed in [47] while altering the object's learned appearance as a function of the geometric transform to simplify its detection and re-identification in the next frame.

3. Method

3.1. Overall Architecture

Under the tracking-by-attention paradigm, an object is tracked by updating its unique query feature to be consistent with new observations and other track hypotheses at each point in time via the attention mechanism [31, 47]. Since attention reasons about the affinity between new observations and existing tracks via feature similarity, queries of tracked objects need to be adjusted to account for the changes in relative pose and appearance resulting from both ego- and object motion between frames. To this end, we propose an LMM, an extension to commonly used purely geometric motion models. The LMM adjusts each object's latent query feature to be consistent with its expected state in the next frame, increasing similarity to new observations of the same object and simplifying the association task. The LMM implements a generic query prediction strategy that can be readily coupled and jointly trained with any query-based detector.

An overview of the proposed architecture is presented in Fig. 2. We utilize a decoder-only transformer architecture as in DETR3D [44], where a set of learnable detection queries $\mathcal{D} = \{\mathbf{d}_1, \dots \mathbf{d}_n\}$ is used to represent hypotheses for newly detected objects in the scene. Following the design in [47], the time independent detection queries are concatenated with a set of track queries $\mathcal{T} = \{\mathbf{t}_1, \dots \mathbf{t}_m\}$

that correspond to hypotheses from the previous frames. Then, the decoder refines both the track hypotheses and new detections jointly by applying self- and cross-attention into features extracted from multi-view camera images by a shared image backbone in an alternating fashion. The final bounding attributes are then decoded from the latent queries by a *feed forward network* (FFN). We kindly refer the reader to [47, 46, 4] for further details on the general MOT architecture. Lastly, we carry the objects over to the next frame by applying both the analytical geometric motion transform as well as the LMM.

3.2. Revisiting Multi-Object Tracking

Traditional model-based tracking systems [17, 1] often rely on three sequential steps: (1) Detection / State update, (2) Prediction and (3) Association. This allows for incorporating inductive biases such as geometric constraints into the different parts of the tracking framework while also maintaining a high level of interpretability. In the following, we outline the challenges of each step in this traditional tracking pipeline and how it is possible to keep these properties for high-dimensional latent object hypotheses.

Detection / State update: In each frame a set $\mathcal{D}$ of new detections is used to update the current belief state of tracked objects $\mathcal{T}$ in the scene. This enables rejecting implausible sensor measurements, updating the estimated bounding box and existence probability of each track, and spawning new tracks for newly appeared objects. The transformer-based tracking-by-attention mechanism mirrors this behavior by performing two attention operations per decoder layer utilizing scaled dot product attention as defined in [42]:

$$Attn(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}(\frac{\mathbf{Q} \cdot \mathbf{K}^T}{\sqrt{d_k}}) \cdot \mathbf{V}. \tag{1}$$

Self-attention within the joint set of track queries and newly spawned detection queries models object interactions, integrating new objects and rejecting duplicate proposals. Afterward, *cross-attention* between all object queries and the camera features is used to refine each object proposal by incorporating sensor measurements. The tracking-by-attention framework uses data-independent detection queries as well as track queries of previous time steps as priors for the detection in the next frame. This potentially simplifies the detection of objects that are far away, partially occluded or hardly visible.

Prediction: Given the current ego motion ${}^{ego_{t+1}}\mathbf{T}_{ego_t}$ and estimated object dynamics, e.g. velocity and turn-rate of each tracked object, a traditional geometric tracking framework predicts the object's pose in the next frame. This is typically achieved utilizing a motion model which is a function of the object's state and dynamics.

For a high-dimensional latent object representation the geometric update in terms of the object's pose should be handled similarly to the explicit bounding box representation since the geometric transform can be applied analytically. However, the high-dimensional appearance representation of the object query also needs to be taken into consideration since the ego and object motion might heavily affect an object's appearance and thus its query feature in the next frame, see Fig. 1. This is crucial since the transformer attention relies on a query-key similarity as defined in Eq. (1). Without a latent appearance update the re-identification of a tracked object in the next frame might be impaired. Firstly, track identity switches or track losses can occur if a track query cannot be associated to the next frame's sensor data in the cross-attention blocks. Second, without proper appearance updates, duplicates might spawn, since existing tracks fail to suppress their newly detected counterparts in the self-attention blocks.

Association: To associate detections in the next frame with existing tracks, any similarity metric between object hypotheses can be used. Traditional methods rely on geometry-based metrics [45, 30] or additional re-ID features [34, 6] to form an affinity matrix between tracks and new detections which can be used together with the Hungarian algorithm to find an optimal matching. Auto-regressive query-based tracking methods [46, 41, 31, 47] solve this problem differently since a track query always represents the same object in the scene resulting in an implicit association. During training, this is enforced by matching each track query to its corresponding object in the scene to which it was assigned at first appearance. In case two hypotheses describe the same object, the model needs to distinguish between newly spawned and already tracked objects and favor the latter. This is crucial since confusions between tracks and newborn detections might result in track losses or identity switches between tracks and new detections at inference time.

As a result of the considerations above, two key challenges arise for auto-regressive query-based tracking: (1) The prediction step needs to model the influence of the geometric transform on the object's pose as well as its latent appearance and semantic features. (2) Due to the implicit association mechanism each track query needs a latent existence probability to efficiently suppress newborn duplicate queries that also belong to the tracked object.

3.3. Latent Motion Models

As motivated above, the prediction step in the tracking pipeline aims to estimate the state of an object in the next frame. In our model, the set of tracked objects and newly spawned detections is defined as a set of latent feature vectors $\mathbf{q} \in \mathcal{Q}$. Additionally, each object's query position is defined with respect to a 3D reference point $\mathbf{r} \in \mathcal{R}$ as proposed in [44]. As a result, the geometric effect of the ego motion for a time delta δ_t between two frames at time t and $t+1$ can be described with a 4×4 homogeneous matrix

$$^{ego_{t+1}}\mathbf{T}_{ego_t} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{bmatrix} \tag{2}$$

which combines the rotation matrix $\mathbf{R}$ and the translation vector $\mathbf{t}$.

Furthermore, the transformer decoder's regression branches predict an estimate of the object's dynamics. These include the estimated velocity $\mathbf{v} = \begin{pmatrix} v_x & v_y \end{pmatrix}$, that is supervised by ground truth data during training, and an optional turn-rate δ_θ for the heading angle θ resulting in

$$^{e'_t}\mathbf{T}_{e_t} = \begin{bmatrix} \cos(\delta_\theta) & -\sin(\delta_\theta) & 0 & v_x \cdot \delta_t \\ \sin(\delta_\theta) & \cos(\delta_\theta) & 0 & v_y \cdot \delta_t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3}$$

For consistent notation, we propose an auxiliary frame e'_t that describes the state of the world after object motion compensation relative to the ego frame at time t. We note that due to the explicit modeling of this transformation, any motion model [30] can be used to constrain the estimated transformations by model-based assumptions.

Hyper-Networks: Besides the explicit geometric update transformation on the object's reference point $\mathbf{r}$ as defined in Eq. (4), an additional update to the object's latent features $\mathbf{q}$ is required to propagate the feature to the next frame. As argued in [39], the effect of the geometric transformation in latent space can by modeled as a linear operator that performs an input-dependent multiplication on the object query in the form of a latent transformation matrix $^b\mathbf{K}_a$ of shape $k \times k$. This matrix is a function of its geometric counterpart $^b\mathbf{T}_a$ and represents an arbitrary transform from frame a to frame b. Geometric and latent information is jointly updated:

$$\mathbf{r}_{e_{t+1}} = {}^{e_{t+1}}\mathbf{T}_{e'_t} \cdot {}^{e'_t}\mathbf{T}_{e_t} \cdot \mathbf{r}_{e_t} \qquad \text{Geometric Update} \tag{4}$$

$$\mathbf{q}_{e_{t+1}} = {}^{e_{t+1}}\mathbf{K}_{e'_t} \cdot {}^{e'_t}\mathbf{K}_{e_t} \cdot \mathbf{q}_{e_t} \qquad \text{Latent Update} \tag{5}$$

We propose a *transformation hyper-network* (TfNet) to estimate the parameters of the $k \times k$ dimensional latent transform matrix $^b\mathbf{K}_a$. This matrix is applied as an input-dependent multiplication with the latent object query $\mathbf{q}$. A latent translational offset is incorporated as an element-wise addition. An overview of the proposed LMM architecture is given in Fig. 3.

Input Representation: The input to the TfNet consists of a rotational and translational part:

$$^b\mathbf{K}_a = \text{TfNet}(^b\mathbf{R}_a, {}^b\mathbf{t}_a), \tag{6}$$

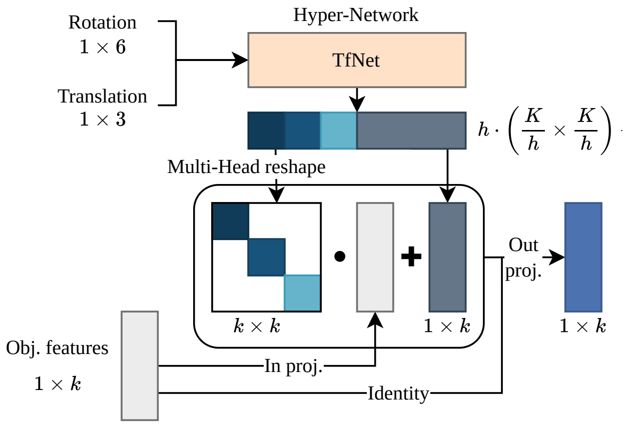

Figure 3. Latent motion model architecture. A geometric transform consisting of a translation and rotation is applied to the high-dimensional object query by using a sparse latent transform matrix K. We estimate the elements of K with a hyper-network (TfNet) and apply the transform as an input dependent multiplication, mimicking the behavior of a homogeneous matrix in 3D. Note the sparse block-diagonal shape of the generated matrix.

whereas $^a\mathbf{R}_b$ describes the rotational component and $^a\mathbf{t}_b$ the translation of $^a\mathbf{T}_b$. While the translation is represented as a 3D vector, as it is done for the geometric operator, the matrix representation for the rotation might not be well-suited as a direct input to the network. Following [48], we instead utilize the 6D rotation representation to model the rotation as a smooth continuous function and to increase the numeric stability.

Sparse Latent Transforms: Since the latent features are typically high-dimensional, for example $k = 256$ [44, 47], a hyper-network that predicts $^b\mathbf{K}_a$ as a full-rank matrix might be over-parameterized or even intractable to train. This is due to the large number of parameters in the output weight matrix $^b\mathbf{K}_a$ that need to be computed per object in each frame. We mitigate this potential issue by adopting the concept of multi-head attention from [42, 4, 44] and propose a sparse multi-head LMM. Here, attention is computed as a combination of h different low-dimensional attention heads that operate on h splits of the feature vector with a dimensionality of $h_{dim} = k/h$ each.

Instead of predicting $k^2 = h^2 \cdot h_{dim}^2$ weights for a full-rank description of K, we propose to only predict $h \cdot h_{dim}^2$ weights for a sparse approximation that drastically reduces the parameter count of the latent transform matrix. Analogously to the attention computation, these are then used as heads along the diagonal of $^b\mathbf{K}_a$ that operate on parts of the k-dimensional latent vector $\mathbf{q}$, see Fig. 3. Since only neighboring dimensions of the feature vector that lie within the same head can influence the latent transform, we follow the attention architecture [42] and incorporate an input and output projection to mitigate this effect.

As a result, with the multi-head LMM the latent transform can be directly applied to the full latent vector in a sparse and numerically more stable fashion, while also streamlining the architecture to follow the layout of the attention blocks that are used in all other parts of the model.

3.4. Track Embeddings for Latent Existence Probability Modeling

As discussed in Section 3.2, the self-attention blocks serve the purpose of allowing for object interactions as well as suppressing newborn detections that belong to an already tracked object. Although it might be sufficient to distinguish between tracks and new detections in this case, the track queries in general require a consistent integration of the track history to account for short-term occlusions and deliver robust existence probability estimates.

Since learned embeddings have been used successfully to incorporate inductive biases in attention-based detectors [2, 8], we propose to use a learned latent *track embedding* to address the aforementioned issues. Using a single shared track embedding $\mathbf{e}$ and a FFN we update all active tracks $\mathcal{T}$ of the current time step using

$$\mathbf{t_i}' = \mathbf{t_i} + \text{FFN}([\mathbf{t_i}, \mathbf{e}]) \qquad \forall \mathbf{t_i} \in \mathcal{T}. \tag{7}$$

This way, the model is flexible to integrate the track embedding to the current latent state of an object and to model the desired distinction between tracks and new detections. As a result, we obtain more consistent existence probabilities and improved track losses, track fragmentations and identity switches, as our experiments in Section 4.2 show.

4. Experiments

We evaluate the performance of STAR-TRACK on the tracking task [33] of the well-established nuScenes dataset [3]. Additionally, we provide extensive ablation studies to evaluate the effects of different LMM configurations, latent track embeddings and transform representations, as well as qualitative results.

4.1. Experimental Setup

Dataset: All experiments are performed on the large-scale nuScenes dataset [3] that consists of 1000 scenes with a length of 20 s each and annotated key frames with a frequency of 2 Hz. We use the official train-, val- and test-set split and train on the seven object classes used in the tracking benchmark [33] as in previous work [47].

Metrics: We report performance using the standard tracking metrics as defined in the nuScenes benchmark [33]: These include the *average multi object tracking accuracy* (AMOTA) as well as the *average multi object tracking precision* (AMOTP) as the two main metrics. Additionally, we report the *number of identity switches* (IDS), *number of track fragmentations* (FRAG) and *number of mostly tracked trajectories* (MT) as secondary metrics. For the full metric definitions and further details, we refer to [3, 33].

Training Configuration: To increase comparability and reproducibility, we closely follow the settings proposed in DETR3D [44] and MUTR3D [47]. Each training sample consists of three consecutive frames. The geometric and latent motion models assume a constant velocity and no turn-rate transformation for each object, as used in [47]. We leave the integration of more complex dynamics models to future work. As in previous works [4, 44, 47], bi-partite matching and the Hungarian algorithm are used to match tracked objects of the current frame with the ground truth. We use Focal-Loss [24] as classification loss and L1-Loss for bounding box regression, see [47] for details. In the training phase, previously matched track queries are always matched to their corresponding ground truth objects. As in [46, 47], we randomly drop tracked queries with a probability $p_{drop} = 0.1$ and spawn false positive tracks with a probability of $p_{fp} = 0.3$. During inference, non-confirmed tracks are kept as inactive for a duration of five frames to handle full occlusions over multiple time steps.

We train all models for 24 epochs with the same random seed on four NVIDIA-V100 GPUs with 16 GB RAM using a batch size of four and a ResNet-101 backbone [15] with a *feature pyramid network* (FPN) [23] as in [47, 44]. As proposed in [47], the transformer decoder utilizes $l = 6$ decoder layers, $q = 300$ detection queries for each frame and a latent dimension of $d_l = 256$ spread over $h = 8$ heads of dimension $d_h = d_l/h = 32$. This is also used as configuration of the proposed multi-head LMM. All experiments use the training schedule proposed in DETR3D [44] that utilizes a learning rate of $2e^{-4}$, a cosine annealing learning rate schedule and AdamW [29] as optimizer.

We initialize the model with an already trained MUTR3D checkpoint to avoid retraining and keep the image backbone and FPN fixed. To initialize the newly introduced LMM, we propose a simple yet effective pretraining scheme: For each sample in the dataset we store the tracking results, consisting of latent queries as well as decoded object proposals from MUTR3D [47] and train the LMM to predict the state of the latent object query vectors of the next frame.

4.2. Comparison to Existing Works

We compare STAR-TRACK to state-of-the-art methods for 3D MOT on multi-view camera images. To control for the effects of different detection algorithms on the overall tracking performance, we present our main comparison in terms of DETR3D-based frameworks, which are well-established and widely used [8, 26, 27, 43]. This allows for a fair assessment of our contributions. As shown

Table 1. Comparison of state-of-the-art methods on the nuScenes benchmark on the validation set. For a fair comparison all methods utilize DETR3D [44] as detector with different image backbone configurations. DETR3D† utilizes the greedy tracking approach proposed in [45]. Due to a potential evaluation error in MUTR3D [47, 32] we add a customized MUTR3D$^+$ baseline. The version of our model that only uses the LMM and no learned track embedding is denoted by *.

Name	Backbone	AMOTA↑	AMOTP↓	RECALL↑	MOTA↑	MT↑	FRAG↓	IDS↓
DETR3D [44]†	ResNet101	0.327	1.372	0.463	0.291	2039	2372	2712
MUTR3D [47]	ResNet101	0.294	1.498	0.427	0.267	-	-	3822
MUTR3D [47]$^+$	ResNet101	0.360	1.411	0.487	0.341	2368	1232	522
CC-3DT [11]	ResNet101	0.359	1.361	0.498	0.326	-	-	2152
PF-Track [35]	VovNet-V2-99	0.362	1.363	-	-	-	-	**300**
STAR-TRACK*	ResNet101	0.378	1.365	0.497	0.354	2467	1241	439
STAR-TRACK	ResNet101	**0.379**	**1.358**	**0.501**	**0.360**	**2468**	**1109**	372

in Table 1, our tracking framework STAR-TRACK that utilizes the novel LMM and track embedding achieves the best performance in all key metrics on the nuScenes benchmark [33] for DETR3D-based [44] tracking algorithms.

In comparison to the greedy tracking DETR3D baseline that uses a purely geometry-based prediction and association [45], our framework improves the main metric AMOTA substantially by 5.2 %. The optimized version of MUTR3D [47, 32] is outperformed by 1.9 %, highlighting the crucial role of the LMM. In particular, we observe a drastic reduction of IDS by 86.2 % compared to the greedy version and by 28.7 % compared to MUTR3D, see Table 1. We address this fact to the spatially and temporally consistent appearance representations provided by the LMM and our proposed track embedding. This benefits the association over time resulting in less track fragmentations (FRAG) and a higher amount of mostly tracked trajectories (MT).

Additionally, STAR-TRACK also outperforms the concurrent works PF-Track [35] by 1.7 % and CC-3DT [11] by 2 % AMOTA, respectively. The former employs both advanced query refinement operations for temporal consistency and a stronger VovNet-V2-99 [20] image backbone and the latter proposes a learned motion model that is based on an LSTM [11].

Evaluating our model with a VovNet-V2-99 trained on both the train and validation set on the nuScenes test set results in 43.9 % AMOTA, 1.256 AMOTP and 607 IDS. This improves over MUTR3D [47] by 16.9 % in AMOTA and even outperforms concurrent work that utilizes stronger detection algorithms [35, 11].

4.3. Ablation and Analysis

Qualitative results: A qualitative example of two consecutive time steps of the nuScenes [3] validation set is shown in Fig. 4. STAR-TRACK is particularly strong in handling large appearance changes, e.g. due to different lighting conditions and tracking road participants under strong object-object occlusions. We provide additional videos of the tracking performance in the supplementary.

Table 2. Effect of training time. For a fair comparison we fine-tune our version of MUTR3D [47] with and without an LMM indicated by w/LMM. Runs denoted by w/Init use a pretrained MUTR3D instead of a pretrained DETR3D [44] checkpoint.

w/LMM	w/Init	AMOTA↑	AMOTP↓	IDS↓
✗	✗	0.338	1.425	531
✗	✓	0.358	1.382	492
✓	✓	**0.378**	**1.365**	**439**

Table 3. Effect of different LMM architectures. w/LMM indicates whether an LMM is used, multi-head (w/MH) denotes a sparse latent transform matrix ${}^{b}\mathbf{K}_{a}$ instead of a full-rank version. The head / matrix size is denoted by $|\ K\ |$.

w/LMM	w/MH	$\mid K \mid$	AMOTA↑	IDS↓
✗	-	-	0.358	492
✓	✗	32^2	0.372	432
✓	✗	96^2	0.370	**402**
✓	✓	$16 \cdot 16^2$	0.374	517
✓	✓	$4 \cdot 64^2$	0.373	434
✓	✓	$8 \cdot 32^2$	**0.378**	439

Effect of Training Time: The effect of longer training schedules is shown in Table 2. MUTR3D [47] gains a performance boost of 2.0 % in AMOTA and 7.3 % in IDS by further fine-tuning. Adding the proposed LMM yields 2 % AMOTA and improves the IDS by 10.7 % as compared to the equally long trained model. This clearly indicates that the use of our LMM results in more consistent tracks with a reduced number of identity switches.

Effect of LMM Architecture: The performance of different LMM architectures is shown in Table 3. Using the proposed sparse multi-head LMM instead of a full-rank representation of the latent motion matrix ${}^{b}\mathbf{K}_{a}$ does not only align the architecture to the multi-head attention blocks but

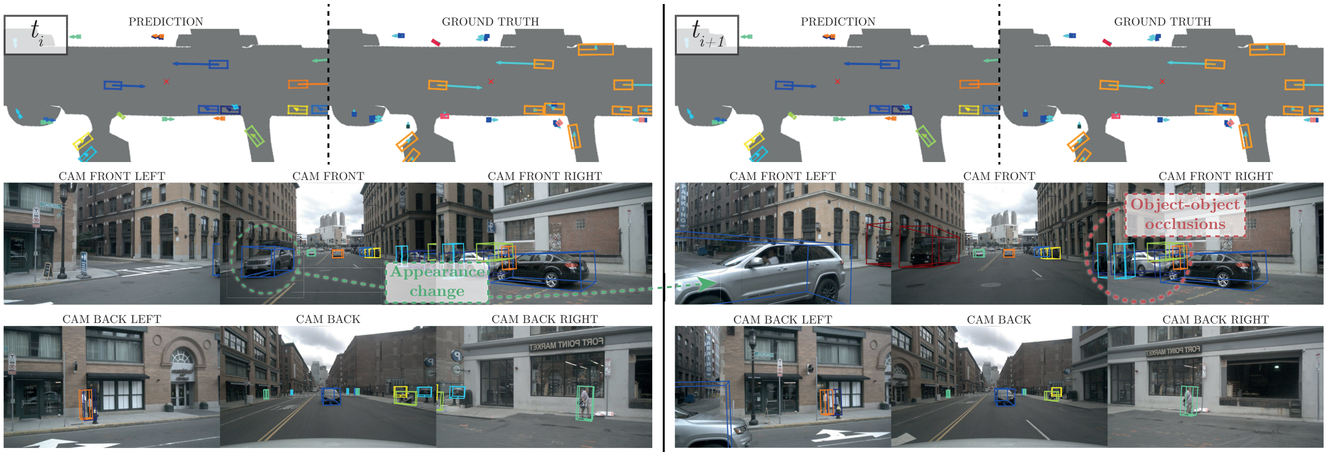

Figure 4. Qualitative results for two consecutive frames on the nuScenes [3] validation set. Upper row shows predictions and ground truth in top view. Different colors of the predicted objects indicate different object ids. The bottom row shows the predictions projected to the multi-view camera images.

Table 4. LMM transform representation. Models that apply object and ego motion separately in a consecutive fashion are denoted with w/Separate. w/Share indicates models that use shared parameters for both transforms and w/Feats denotes LMMs that utilize the query feature additionally as input to the TfNet hyper-network.

w/Separate	w/Share	w/Feats	AMOTA↑	IDS↓
✗	✓	✗	0.370	492
✗	✓	✓	0.371	446
✓	✗	✗	0.377	**411**
✓	✗	✓	0.366	464
✓	✓	✓	0.370	426
✓	✓	✗	**0.378**	439

also reduces the amount of output parameters of the hypernetwork. This is key to scale the latent transform matrix to the full latent space dimensions. Using the same configuration as the attention blocks for the multi-head LMM results in an boost in AMOTA of $0.8\,\%$ over a full-rank LMM.

Effect of Transform Representation: The effect of different strategies to apply the transformation modeled by the LMM is shown in Table 4. We do not observe a performance increase when the latent query feature is used as an additional input to the TfNet. This is in line with our general design paradigm to compute the latent motion matrix solely from its geometric counterpart. Although it is beneficial to apply the LMM twice instead of merging the object and ego motion to a single transform, using shared parameters for the object and ego motion compensation cuts the number of parameters in half and does not cause any ill-effects. This supports the general design to model any geometric transform with the LMM without creating an explicit distinction between object and ego motion.

5. Conclusion

This paper presented STAR-TRACK, a novel approach for 3D object tracking-by-attention that is compatible with any query-based object detector. We transferred the concept of motion models from traditional geometry-based trackers to the tracking-by-attention paradigm in terms of latent motion models that predict the spatio-temporal appearance change of objects between two frames. This allowed for a prediction step that models a geometric transform in an analytical way and applies this transform in the latent space with a learned motion matrix at the same time. An additional latent track embedding improved the latent existence probability of tracks. In our experimental evaluation, the integrated system demonstrated significant improvements in all relevant tracking metrics. Increased track consistency was observed as a particular strength evident from significantly decreased identity switches and track fragmentations.

We hope that this work serves as a foundation for future 3D MOT research with the aim of integrating model-based assumptions to end-to-end tracking approaches. While the potential of this has been clearly demonstrated in this work, limitations and opportunities for improvement have also been identified.

Limitations: The implicit association used in the tracking-by-attention scheme falls short in cases with poor motion estimates, since the resulting prediction might impair the re-identification performance in the next frame. This could lead to errors in object position or track losses. In future work, multi-hypothesis tracking [19, 7] could be adopted to model uncertainty in object dynamics and to relax the one-to-one relation of track queries between frames. Additionally, the implicit assignment results in a discrepancy between training and inference time, since the ground truth matching only assigns the correct ground truth object to a single query during training. This could be solved with a non-strict matching approach similar to the 2D detection and tracking case for DETR-like architectures [16]. The novel idea of track embeddings is a promising research direction that could be extended to model the uncertainty distribution of each tracked object explicitly.

Acknowledgement: The research leading to these results is funded by the German Federal Ministry for Economic Affairs and Climate Action within the project "KI Delta Learning" (Förderkennzeichen 19A19013A). The authors would like to thank the consortium for the successful cooperation.

References

[1] Claudine Badue, Rânik Guidolini, Raphael Vivacqua Carneiro, Pedro Azevedo, Vinicius B Cardoso, Avelino Forechi, Luan Jesus, Rodrigo Berriel, Thiago M Paixao, Filipe Mutz, et al. Self-Driving Cars: A Survey. *Expert Systems with Applications*, 2021. 1, 2, 4

[2] 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 *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1, 2, 6

[3] 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 *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 6, 7, 8

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-End Object Detection with Transformers. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2020. 2, 4, 5, 6

[5] Qiang Chen, Jian Wang, Chuchu Han, Shan Zhang, Zexian Li, Xiaokang Chen, Jiahui Chen, Xiaodi Wang, Shuming Han, Gang Zhang, et al. Group DETR v2: Strong Object Detector with Encoder-Decoder Pretraining. *arXiv.org*, arXiv:2211.03594, 2022. 2

[6] Gioele Ciaparrone, Francisco Luque Sánchez, Siham Tabik, Luigi Troiano, Roberto Tagliaferri, and Francisco Herrera. Deep learning in video multi-object tracking: A survey. *Neurocomputing*, 2020. 4

[7] Pierre Del Moral. Nonlinear Filtering: Interacting Particle Resolution. *Comptes Rendus de l'Académie des Sciences-Series I-Mathematics*, 1997. 8

[8] Simon Doll, Richard Schulz, Lukas Schneider, Viviane Benzin, Markus Enzweiler, and Hendrik PA Lensch. SpatialDETR: Robust Scalable Transformer-Based 3D Object Detection from Multi-View Camera Images with Global Cross-Sensor Attention. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2022. 1, 2, 6

[9] Gopi Krishna Erabati and Helder Araujo. Li3DeTr: A LiDAR based 3D Detection Transformer. In *Proc. of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2023. 2

[10] Andreas Ess, Konrad Schindler, Bastian Leibe, and Luc Van Gool. Object Detection and Tracking for Autonomous Navigation in Dynamic Environments. *International Journal of Robotics Research (IJRR)*, 2010. 1, 2

[11] Tobias Fischer, Yung-Hsu Yang, Suryansh Kumar, Min Sun, and Fisher Yu. Cc-3dt: Panoramic 3d object tracking via cross-camera fusion. *Proc. Conf. on Robot Learning (CoRL)*, arXiv:2212.01247, 2022. 7

[12] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2012. 2

[13] Junru Gu, Chenxu Hu, Tianyuan Zhang, Xuanyao Chen, Yilun Wang, Yue Wang, and Hang Zhao. ViP3D: End-to-end Visual Trajectory Prediction via 3D Agent Queries. *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2023. 1

[14] David Ha, Andrew Dai, and Quoc V Le. HyperNetworks. *Proc. of the International Conf. on Learning Representations (ICLR)*, 2017. 2, 3

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016. 6

[16] Ding Jia, Yuhui Yuan, Haodi He, Xiaopei Wu, Haojun Yu, Weihong Lin, Lei Sun, Chao Zhang, and Han Hu. DETRs with Hybrid Matching. *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2, 9

[17] Rudolph Emil Kalman. A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME–Journal of Basic Engineering*, 1960. 4

[18] Peter Karkus, Boris Ivanovic, Shie Mannor, and Marco Pavone. DiffStack: A Differentiable and Modular Control Stack for Autonomous Vehicles. In *Proc. Conf. on Robot Learning (CoRL)*, 2022. 1

[19] Chanho Kim, Fuxin Li, Arridhana Ciptadi, and James M Rehg. Multiple Hypothesis Tracking Revisited. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2015. 8

[20] Youngwan Lee, Joong-won Hwang, Sangrok Lee, Yuseok Bae, and Jongyoul Park. An Energy and GPU-Computation Efficient Backbone Network for Real-Time Object Detection. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2019. 7

[21] Yanwei Li, Yilun Chen, Xiaojuan Qi, Zeming Li, Jian Sun, and Jiaya Jia. Unifying Voxel-based Representation with Transformer for 3D Object Detection. In *Advances in Neural Information Processing Systems (NIPS)*, 2022. 1, 2

[22] Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Yu Qiao, and Jifeng Dai. BEVFormer: Learning Bird's-Eye-View Representation from Multi-Camera Images via Spatiotemporal Transformers. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2022. 1, 2

[23] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature Pyramid

Networks for Object Detection. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. 6

[24] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal Loss for Dense Object Detection. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. 6

[25] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2014. 2

[26] Yingfei Liu, Tiancai Wang, Xiangyu Zhang, and Jian Sun. PETR: Position Embedding Transformation for Multi-View 3D Object Detection. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2022. 6

[27] 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.org*, arXiv:2206.01256, 2022. 6

[28] Zhijian Liu, Haotian Tang, Alexander Amini, Xinyu Yang, Huizi Mao, Daniela Rus, and Song Han. BEVFusion: Multi-Task Multi-Sensor Fusion with Unified Bird's-Eye View Representation. *arXiv.org*, arXiv:2205.13542, 2022. 2

[29] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. *Proc. of the International Conf. on Learning Representations (ICLR)*, 2019. 6

[30] Wenhan Luo, Junliang Xing, Anton Milan, Xiaoqin Zhang, Wei Liu, and Tae-Kyun Kim. Multiple Object Tracking: A Literature Review. *Artificial Intelligence (AI)*, 2021. 4, 5

[31] Tim Meinhardt, Alexander Kirillov, Laura Leal-Taixe, and Christoph Feichtenhofer. TrackFormer: Multi-Object Tracking with Transformers. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1, 2, 3, 4

[32] MUTR3D Evaluation issue #15. https://github.com/a1600012888/MUTR3D/issues/15. Accessed:03.03.23. 7

[33] nuScenes Tracking Task. https://nuscenes.org/tracking. Accessed: 22.02.23. 1, 2, 6, 7

[34] Jiangmiao Pang, Linlu Qiu, Xia Li, Haofeng Chen, Qi Li, Trevor Darrell, and Fisher Yu. Quasi-Dense Similarity Learning for Multiple Object Tracking. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2, 4

[35] 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.org*, arXiv:2302.03802, 2023. 7

[36] Ziqi Pang, Zhichao Li, and Naiyan Wang. SimpleTrack: Understanding and Rethinking 3D Multi-object Tracking. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2023. 2

[37] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2015. 2

[38] Ergys Ristani and Carlo Tomasi. Features for Multi-Target Multi-Camera Tracking and Re-Identification. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[39] Felicia Ruppel, Florian Faion, Claudius Gläser, and Klaus Dietmayer. Transformers for Multi-Object Tracking on Point Clouds. In *Proc. IEEE Intelligent Vehicles Symposium (IV)*, 2022. 2, 3, 5

[40] Robin Schubert, Eric Richter, and Gerd Wanielik. Comparison and evaluation of advanced motion models for vehicle tracking. In *IEEE International Conference on Information Fusion*, 2008. 2

[41] Peize Sun, Jinkun Cao, Yi Jiang, Rufeng Zhang, Enze Xie, Zehuan Yuan, Changhu Wang, and Ping Luo. TransTrack: Multiple Object Tracking with Transformer. *arXiv.org*, arXiv:2012.15460, 2020. 2, 4

[42] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. In *Advances in Neural Information Processing Systems (NIPS)*, 2017. 2, 4, 5

[43] Shihao Wang, Xiaohui Jiang, and Ying Li. Focal-PETR: Embracing Foreground for Efficient Multi-Camera 3D Object Detection. *arXiv.org*, arXiv:2212.05505, 2022. 6

[44] 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 *Proc. Conf. on Robot Learning (CoRL)*, 2022. 1, 2, 3, 5, 6, 7

[45] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Center-based 3D Object Detection and Tracking. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2, 4, 7

[46] Fangao Zeng, Bin Dong, Yuang Zhang, Tiancai Wang, Xiangyu Zhang, and Yichen Wei. MOTR: End-to-End Multiple-Object Tracking with Transformer. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2022. 4, 6

[47] Tianyuan Zhang, Xuanyao Chen, Yue Wang, Yilun Wang, and Hang Zhao. MUTR3D: A Multi-camera Tracking Framework via 3D-to-2D Queries. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2022. 1, 2, 3, 4, 5, 6, 7

[48] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the Continuity of Rotation Representations in Neural Networks. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 5

[49] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable DETR: Deformable Transformers for End-to-End Object Detection. In *Proc. of the International Conf. on Learning Representations (ICLR)*, volume arXiv:2010.04159, 2021. 2