Cross-Modal 3D Object Detection and Tracking for Auto-Driving

Yihan Zeng[1], Chao Ma[1⋆], Ming Zhu[1], Zhiming Fan[1] and Xiaokang Yang[1]

***Abstract*— Detecting and tracking objects in 3D scenes play crucial roles in autonomous driving. Successfully recognizing objects through space and time hinges on a strong detector and a reliable association scheme. Recent 3D detection and tracking approaches widely represent objects as points when associating detection results with trajectories. Despite the demonstrated success, these approaches do not fully exploit the rich appearance information of objects. In this paper, we present a conceptually simple yet effective algorithm, named AlphaTrack, which considers both the location and appearance changes to perform joint 3D object detection and tracking. To achieve this, we propose a cross-modal fusion scheme that fuses camera appearance feature with LiDAR feature to facilitate 3D detection and tracking. We further attach an additional branch to the 3D detector to output instance-aware appearance embedding, which significantly improves tracking performance with our designed association mechanisms. Extensive validations on large-scale autonomous driving dataset demonstrate the effectiveness of the proposed algorithm in comparison with state-of-the-art approaches. Notably, the proposed algorithm ranks first on the nuScenes tracking leaderboard to date.**

I. INTRODUCTION

3D object detection and tracking are two fundamental tasks for autonomous driving. A recent trend in multi-object tracking (MOT) is to convert 3D object detectors into trackers and combine both tasks in the same framework. Typically, this tracking framework represents objects as points when associating detection results with trajectories, assuming that position changes between consecutive frames are in a local region. Despite the demonstrated success, this framework tends to fail in the presence of large motion changes and noisy detection results, due to the lack of rich appearance information of objects, as shown in Fig. 1. In this work, we aim at a conceptually simple yet effective algorithm, which considers both the position and appearance changes for simultaneous 3D object detection and tracking.

When learning instance-aware appearance representation for 3D objects, two main challenges arise. First, current 3D detectors mainly build upon LiDAR points, which are good at distance sensing but lack of texture information. In contrast, camera images provide rich color and texture cues of objects. The progresses on 2DMOT [2]–[4] show that data association greatly benefits from re-identification embedding, suggesting the importance of appearance information in camera images to distinct objects. However, how to exploit camera images with LiDAR points to effectively represent objects in 3D space still remains an open problem. Second, since joint

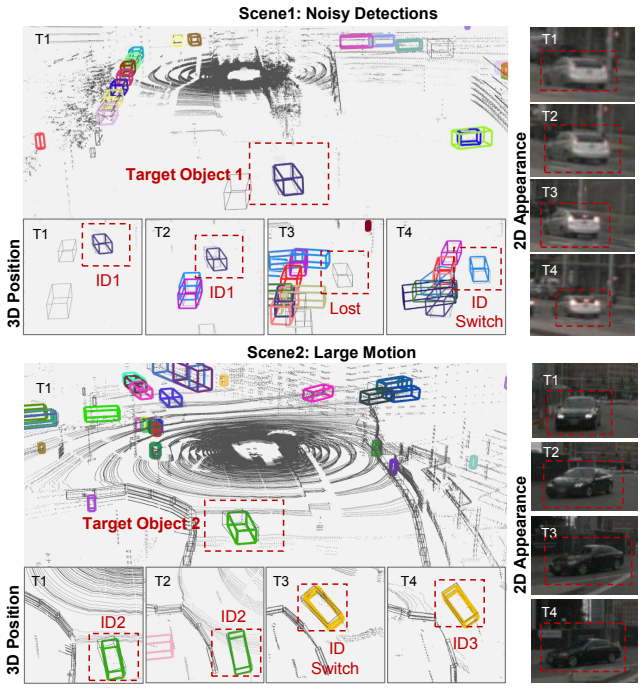

Fig. 1. Tracking results on the nuScenes dataset from the state-of-the-art LiDAR-only detector, CenterPoint [1], which only considers the position changes. The ground truth bounding boxes are labeled in gray and the tracked objects are labeled in colors. When position affinity leads to incorrect tracking results in the presence of noisy detections and large motion changes, the appearance textures from images are able to distinguish the objects. This motivates us to take both position and appearance changes into consideration for robust object tracking.

3D detection and tracking requires high computational efficiency, it is unpractical to train a separate association network to extract instance-aware appearance features per detection like prior trackers [5]–[7]. Moreover, features extracted from existing 3D detectors are instance-agnostic, with a focus on the category-level differences. Therefore, it is of great importance to enable 3D detectors to generate instance-aware feature representation to facilitate 3D tracking association.

To address these issues, we propose to improve the state-of-the-art LiDAR-only 3D detector [1] with a novel cross-modal fusion scheme. We further attach an additional branch to the detector to jointly learn instance-aware embeddings to help tracking association. Our proposed model simultaneously produces the location and appearance embeddings of 3D objects in a single forward pass, which accelerates the association process. Specifically, our cross-modal fusion scheme is motivated by PointPainting [8], which concatenates point-wise segmentation scores with each LiDAR

[1] All the authors are with MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University, China

⋆ C. Ma is the corresponding author. Email: chaoma@sjtu.edu.cn.

point. However, segmentation scores are too semantically abstract to maintain sufficiently detailed appearance information. Thus we explore a better representation by replacing the segmentation scores with image features that contains richer appearance clues. To better handle the modality differences, we further apply separate 3D convolution branches to extract the LiDAR feature and camera feature respectively. Extensive ablation studies show that the above designs of our fusion scheme benefits both detection and tracking. As the detection branch emphasizes category-level differences while the appearance embedding branch focuses on instance nuances, jointly training the detection and embedding branches leads to degraded performance on both tasks. To strike a balance between these two tasks, we alternatively train these two branches. Experiments show that our proposed embedding representation is effective in distinguishing instances without hurting the detection branch.

We develop a novel tracking pipeline taking both position and appearance changes into consideration for 3D MOT. We observe that these two types of changes often vary across different scenarios, so their reliability should be carefully measured in association mechanisms. Prior methods simply apply weighted summation [6] or fusion with convolution network [5] to exploit these two clues. But those fusion mechanisms do not explicitly complement each other, and may even accumulate noisy association from both sides. This motivates us to explicitly complement the position and appearance clues. Since the vehicles in driving datasets move regularly in most cases, we base the association on position affinity. As appearance features are able to re-identify instances, we use them to filter incorrect matched pairs and re-associate lost objects. Extensive ablation studies show that our presented mechanisms facilitate tracking association by better exploiting position and appearance clues.

We validate our proposed AlphaTrack algorithm on the large-scale autonomous driving dataset nuScenes [9]. Compared to the state-of-the-art approach [1], we achieve +10.69% and +9.55% gains in detection and tracking accuracy. The main contributions are summarized as follows:

- We propose a novel cross-modal fusion scheme that fuses LiDAR points with image features. We further empower our model to jointly learn instance-aware appearance representation together with detection.
- We present novel association mechanisms that explicitly exploit both the motion and appearance changes for 3D object tracking.
- We extensively validate the design choices of the proposed algorithm on the large-scale nuScenes dataset. Our AlphaTrack algorithm ranks first on the leaderboard of the nuScenes tracking challenge to date.

II. RELATED WORK

A. 3D Obejct Detection

Tracking performance hinges on detection results. Though LiDAR point clouds provide a very accurate range view for 3D detection and tracking, they are sparse and lack fine-grained textures when compared with camera images. Recent 3D object detectors wildly exploit cross-modal data fusion to utilizes the best of both worlds. The state-of-the-art LiDAR-only detectors [1], [10], [11] use convolution networks to transfer LiDAR points to the bird's eye view (BEV) and outperform the range-view methods [12], [13]. Hence, the core challenge of fusion lies in consolidating the LiDAR bird's eye view with the camera view.

There are three typical types of fusion scheme. The proposal level fusion methods [14], [15] perform fusion at region proposal level, which brings computation load. The feature level fusion methods [16], [17] perform feature sharing across image and LiDAR backbones, which suffers from the misaligned viewpoint. Instead, point level fusion methods [8], [18] augment LiDAR points with point-wise concatenation. For better representation, our method replaces the concatenated segmentation scores in [8] with the projected image features. Besides, we exploit separate 3D convolution backbones to flatten voxelized LiDAR features and camera features into BEV. Experiments show that our fusion scheme facilities both 3D detection and tracking.

B. Multi-Object Tracking

Existing MOT algorithms typically follow the tracking-by-detection paradigm [19], [20], where the first stage detects objects in bounding boxes and the second stage links the detection results via data association. A proper affinity measure is indispensable in data association. Thanks to the precise distance sensing provided by LiDAR, current 3D MOT methods mainly rely on position distance to measure affinity, including Euclidean distance [1], Mahalanobis distance [21], [22] and the IoU metric [20]. But these trackers are unlikely to handle large motion changes and noisy detection results by solely considering position changes.

As motion estimation in the 2D image is highly challenging, 2D MOT methods wildly explore rich textures in images as re-identification features to greatly improve the affinity measurement. Early works [23]–[27] utilize offline person Re-ID models [27], [28] to extract discriminative features for each object. For computational efficiency, recent methods [2], [3], [29] explore one-shot methods to jointly detect objects and learn instance-level features. The demonstrated success suggests that it is feasible to enable 3D object detectors to learn instance-aware features.

A number of methods [5]–[7] attempt to introduce appearance feature or geometric feature in 3D MOT. However, those methods consider detection and tracking as two separate tasks. They extract instance-level feature independent of detector for association, which takes extra post-processing time after detection. PnPNet [30] presents an end-to-end model to solve detection and feature representation for tracking. However, it is based on LiDAR-only input and unable to utilize appearance features of camera images. Besides, it does not achieve better results than CenterPoint [1] on the nuScenes dataset. Different from prior works, we propose to empower 3D detector [1] to jointly learn instance-aware appearance embedding with the cross-modal fusion scheme. Our method exploits the camera feature without introducing

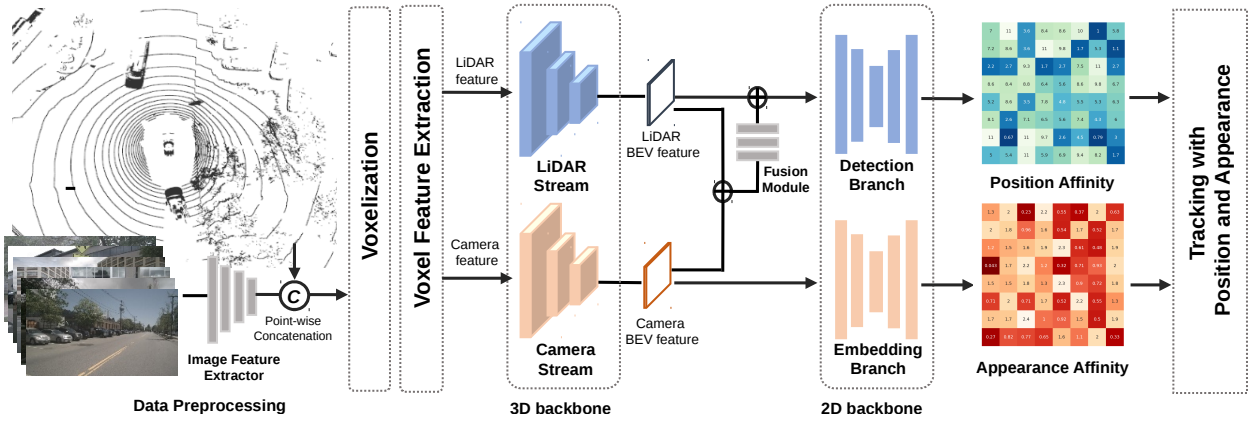

Fig. 2. Architecture overview. First, we fuse each LiDAR point with corresponding point-wise image feature. Second, after voxelization and voxel-wise feature extraction, we exploit separate 3D convolution backbones to flatten LiDAR features and camera features into BEV maps, i.e, LiDAR stream and camera stream. Third, we produce 3D detections and instance-aware appearance embeddings jointly using two 2D convolution networks, i.e, the detection branch and the embedding branch. In the tracking association, we take both position affinity and appearance affinity into consideration.

separate models in tracking association. Experiments show that the proposed embedding representation is effective to distinguish instances and thus significantly improves the tracking performance with our association mechanisms.

III. APPROACH

This section presents the proposed 3D detection and tracking algorithm in detail, which considers both the position and appearance changes of objects across space and time.

We construct our method on the basis of the state-of-the-art LiDAR-only 3D detector, CenterPoint [1], which predicts objects as points and associates objects by position changes. In order to exploit rich appearance information, we equip our model with cross-modal fusion scheme and enable the model to output instance-aware appearance representation to distinct objects. Fig. 2 shows an overview of our method. First, we concatenate LiDAR points with corresponding point-wise images features. Second, we modify the baseline detector to process LiDAR features and camera features for jointly producing 3D detection results and instance-aware appearance embedding. Lastly, we exploit the instance-aware appearance embedding to measure the affinity for tracking with proposed tracking association mechanisms.

A. Problem Context

We phrase 3D detection and tracking as a jointly learning problem. Let the object state be $s = (P, A)$. P denotes the 3D detection results that include center location (x, y, z), bounding box dimension and orientation (h, w, l, o), which locates objects in 3D world coordinate. And A denotes the instance-aware appearance embeddings, which re-identify each object instance. For 3D tracking, we aim to find trajectories for each object in a sequence. Each trajectory T^i links detected object state in sequence $\{s^i_a, s^i_{a+1}, \ldots, s^i_b\}$, which starts at frame a where object appears and ends at frame b where object disappears. Our proposed tracking algorithm considers both position affinity and appearance affinity. For any observed object from adjacent frames t_1 and t_2, the object state should satisfy the following conditions: (i) $D_P(s^i_{t_1}, M_{t_1 \to t_2}(s^i_{t_2})) < D_P(s^i_{t_1}, M_{t_1 \to t_2}(s^j_{t_2}))$. (ii) $D_A(s^i_{t_1}, s^i_{t_2}) < D_A(s^i_{t_1}, s^j_{t_2})$, where trajectory index $i \neq j$, $M_{t_1 \to t_2}(\cdot)$ represents the motion prediction from frame t_1 to frame t_2, $D_P(\cdot)$ represents the distance metric of position and $D_A(\cdot)$ represents the distance metric of appearance embedding.

We obtain the state representation including P and A jointly from our learnable end-to-end model, equipped with cross-modal fusion scheme that exploits both LiDAR points and camera images.

B. Cross-Modal Fusion Scheme

Our proposed cross-modal fusion scheme consists of data preprocessing and architecture modifications. We present a novel representation that fuse camera feature with LiDAR feature. It enables the model to exploit camera appearance information for both detection and tracking.

1) Data Preprocessing: We concatenate the input LiDAR points with point-wise image features. Each LiDAR point originally contains the spatial location (x, y, z), the reflectance ratio r and the relative timestamp of multiple LiDAR sweep t. To augment LiDAR points with corresponding image features, we first extract image feature map by a 2D detector backbone, where the channel number is set to 64 in our implementation. Then we set up the correspondence between the LiDAR coordinate $\mathbf{l}_{xyz}$ with camera coordinates $\mathbf{l}_{uv}$ with a homogeneous transformation $\mathbf{T}$ followed by a projection into the image. Let the camera matrix be $\mathbf{M}$. We project LiDAR point into the image plane as follows:

$$(u, v, 1) = \mathbf{M} \cdot \mathbf{T} \times (x, y, z) \tag{1}$$

We fetch the corresponding point-wise image feature vector f for each LiDAR point. The augmented points, denoted as (x, y, z, r, t, f), are then voxelized for voxel-wise features.

2) 3D backbones: Our architecture follows the widely applied voxel-based pipeline [11] to transfer voxelized features into the BEV map. After voxelization and voxel feature extraction, we modify the architecture to split the

3D convolution backbone into two parallel streams to process LiDAR features and camera features separately, i.e., a LiDAR stream and a camera stream. Considering the large modality gap between LiDAR features and camera features, separate 3D convolution backbones show favorable performance on narrowing the gap. Specifically, the location stream flatten the LiDAR features to the BEV perspective as LiDAR BEV map $M_L \in \mathcal{R}^{W\times H\times C}$, and the feature stream transfers the camera features into camera BEV map $M_C \in \mathcal{R}^{W\times H\times C}$. Note that M_L can be regarded as the same normal representation of LiDAR input as in original CenterPoint, while M_C is the additional BEV representation of the camera images. The two types of BEV feature maps are then fed into 2D convolution branches for detection and appearance representation learning.

C. Joint Learning of Detection and Appearance Embedding

We apply two branches to learn detection and embedding representation jointly in a single forward pass.

1) Detection Branch: We follow the baseline detector to design the detection branch, which consists of a 2D RPN network and regression heads. As CenterPoint only exploits the LiDAR BEV features, our model exploits the fusion module to apply both LiDAR feature and camera feature. Specifically, our fusion module first apply 3 layers of 2D convolutions to fuse features, i.e., $Conv(M_{in}) \rightarrow M_{fuse} \in \mathcal{R}^{W\times H\times 2C}$, where $M_{in} \in \mathcal{R}^{W\times H\times 2C}$ is the concatenation of LiDAR feature M_L and camera feature M_C. Then we concatenate the fused output to LiDAR BEV feature again to retain more location information that is critical for detection regression as $Cat(M_{fuse}, M_L) \in \mathcal{R}^{W\times H\times 3C}$. Finally, the RPN network transfers our concatenated features into BEV feature-map $M_{Det} \in \mathcal{R}^{W\times H\times C^{'}}$, which is further fed into the regression heads to output detection results. The detection losses of regression heads keep the same as CenterPoint, including focal loss, center loss of heatmap, and L1 regression losses for size, offset, rotation, and velocity.

2) Appearance Embedding Branch: We attach an additional embedding branch parallel to the detection branch. The appearance branch consists of a 2D RPN network and convolution heads as well. The appearance representation is based on the camera feature map M_C. The 2D RPN network first generates feature map $M_{Emb} \in \mathcal{R}^{W\times H\times C^{'}}$. The regression head then predicts appearance embedding on top of the RPN network output. In particular, the regression head includes one layer of deformable convolution and two layers of 3×3 convolution with 512 kernels. The output appearance representation is in BEV map view $E \in \mathcal{R}^{H\times W\times 512}$. As the outputs of the appearance and detection branches share the same range in the BEV map, the detection properties and appearance representation for each object can be easily aligned and jointly produced. We implement the training procedure of appearance representation as a classification task. During the training process, we extract identical appearance feature vector e^i of groundtruth box b^i at its central location in the heatmap and map the vector to the class distribution vector $p^i(k)$. Let the one-hot representation of the ground truth class label be $L^i(k)$. We compute the softmax loss as:

$$L_{embedding} = -\sum_{i=1}^{N}\sum_{k=1}^{K} L^i(k)\log(p^i(k)), \tag{2}$$

where K is the number of classes.

The straightforward uniformly training of both branches will lead to degraded performance on both tasks. Because detection emphasizes category information for object classification, whereas the embedding branch requires instance-aware information to distinguish instances. We apply an alternatively training strategy to strike a balance between both sides. We first train the detection branch for 20 epochs. Then we freeze the detection branch and train the appearance embedding branch for another 10 epochs. Finally, we jointly train the two branches together for one more epoch.

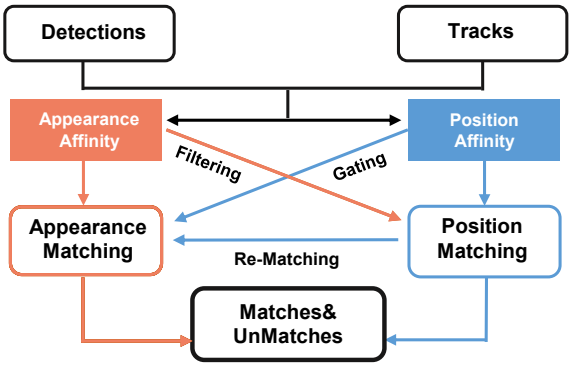

Fig. 3. Appearance affinity and position affinity are complementary in our tracking pipeline. On one hand, appearance affinity filters the incorrectly matched pairs due to fault position estimate. On the other hand, the appearance affinity re-matches the lost pairs with a gating threshold of position distance.

D. Tracking Association

With the detection result and the corresponding appearance embedding for each 3D object, we perform tracking association by computing position affinity and appearance affinity, as shown in Fig. 3. A favorable strategy has to take the reliability of both position and appearance changes into consideration. We propose a novel tracking association pipeline to explicitly exploit these two clues to complement each other, including a filtering mechanism and a re-matching mechanism. Based on the observation that vehicles in driving datasets move regularly in most cases, we weigh more on position affinity. As appearance features are able to re-identify instances, we use them to filter incorrect matched pairs and re-detect lost objects.

The position affinity is measured by the position distance D_P between pairs of detections and tracks. As in original CenterPoint [1], position distance can be calculated by the center point distance D_P^{vel} of 3D bounding boxes via velocity prediction model as following:

$$D_P^{vel}(T_i^{t-1}, \hat{D}_j^t) = ||\mathbf{p}_j^{t-1}, \hat{\mathbf{p}}_i^t - \hat{\mathbf{v}}||_2, \tag{3}$$

where $\mathbf{p}_j^{t-1}$ is the center location of track T_i^{t-1}, $\hat{\mathbf{p}}_i^t$ and $\hat{\mathbf{v}}$ are the center location and regressed velocity of detection $\hat{D}_j^t$.

TABLE I. Ablation studies on cross-modal fusion scheme and appearance embedding branch on the nuScenes [9] validation set. FI: fusion information applied to augment LiDAR points, including Seg (segmentation scores) and Feat (image features). FM: fusion mechanism, including EF (early fusion mechanism as point concatenation) and LF (late fusion mechanism at BEV map). AE: appearance embedding branch, which trained with uniformly strategy or alternatively strategy. We report both detection accuracy and tracking accuracy (mAP% / AMOTA%) for each tracking class.

	FI	FM	AE	Bicycle	Bus	Car	Motorcycle	Pedestrian	Trailer	Truck	mAP↑/AMOTA↑
(a)	-	-	-	35.92 / 40.89	67.23 / 79.86	84.73 / 82.93	57.41 / 54.59	82.85 / 73.61	35.30 / 48.85	54.83 / 65.20	59.75 / 63.72
(b)	Seg	EF	-	52.76 / 51.74	69.21 / 79.60	85.50 / 82.81	61.42 / 62.12	85.49 / 74.57	39.90 / 48.56	56.98 / 64.59	64.47 / 66.28
(c)	Feat	EF	-	57.02 / 58.27	71.75 / 82.17	86.84 / 83.85	72.25 / 77.21	86.77 / 74.26	41.93 / 51.57	59.36 / 67.84	67.99 / 70.74
(d)	Feat	LF	-	62.09 / 62.61	**74.56** / 83.03	87.50 / 84.41	**75.78** / **78.18**	86.96 / 73.71	**43.86** / 53.97	61.57 / 70.41	70.33 / 72.33
(e)	Feat	LF	Uniform	56.94 / 59.13	70.02 / 80.07	85.96 / 82.77	70.26 / 74.67	85.86 / 72.66	38.17 / 46.57	59.13 / 68.44	67.99 / 69.19
(f)	Feat	LF	Alter	**64.26** / **65.86**	74.05 / **83.67**	**87.60** / **85.27**	74.94 / **78.18**	**87.15** / **74.83**	43.32 / **54.64**	**61.78** / **70.49**	**70.44** / **73.27**
Gains from a to f				+28.14 / +24.91	+6.82 / +3.81	+2.87 / +2.34	+17.53 / +23.59	+4.30 / +1.22	+8.02 / +5.79	+6.95 / +5.29	+10.69 / +9.55

TABLE II. Ablation studies on tracking association mechanisms on the nuScenes validation set. Our mechanisms including two components (Fiter and Re-Match) to apply appearance affinity based on position affinity, comparing with two simple mechanism (Sum and Conv) that fuse appearance affinity with position affinity. We evaluate the tracking performance in terms of AMOTA and IDS based on two kinds of motion models.

	Motion	Sum	Conv	Filter	Re-Match	AMOTA↑(%)	IDS↓
(a1)	Kalman					68.73	1021
(b1)	Kalman	✓				69.12	967
(c1)	Kalman		✓			67.68	1152
(d1)	Kalman			✓		68.53	3432
(e1)	Kalman			✓	✓	**70.00**	**929**
(a2)	Velocity					72.39	642
(b2)	Velocity	✓				72.77	639
(c2)	Velocity		✓			70.76	994
(d2)	Velocity			✓		73.21	715
(e2)	Velocity			✓	✓	**73.27**	**575**

The regressed velocity is trained with groundtruth, which is not always available. So we also calculate position distance based on the Kalman filter motion model as in [21], which applies the Mahalanobis distance D_P^{kal} as following:

$$D_P^{kal}(T_i^{t-1}, \hat{D}_j^t) = \sqrt{(\mathbf{o}_i^t - \mathbf{H}\mathbf{A}\mu_j^{t-1})\mathbf{S}_t^{-1}, (\mathbf{o}_i^t - \mathbf{H}\mathbf{A}\mu_j^{t-1})}, \quad (4)$$

where $\hat{\mathbf{o}}_i^t$ denote the detection attribute of $\hat{D}_j^t$, μ_j^{t-1} denotes the estimated mean of state of track T_i^{t-1}, $\mathbf{H}$ and $\mathbf{A}$ denote the linear observation model and state transition matrix that makes prediction for tracks, $\mathbf{S}_t$ is the innovation covariance.

The appearance affinity is measured by the appearance distance D_A between pairs of detections and tracks. We compute the cosine distance between high-dimensional appearance vectors as following:

$$D_A(T_i^{t-1}, \hat{D}_j^t) = \frac{\mathbf{e}_i^{t-1} \cdot \mathbf{e}_j^t}{\{||\mathbf{e}_i^{t-1}||_2\}\{||\mathbf{e}_j^t||_2\}}, \quad (5)$$

where $\mathbf{e}_i^{t-1}$ and $\mathbf{e}_j^t$ denotes the appearance embedding features of track T_i^{t-1} and detection $\hat{D}_j^t$ respectively.

As illustrated in Fig. 3, for each frame in a video clip, the tracking association involves two stages. In the first stage, we compute both the position affinity and appearance affinity between the detections and the predicted tracks in the track store. We run greedy bipartite matching on position affinity and remove the matched pairs whose appearance affinity ranks beyond the top rate λ (λ is set to 0.4) among all pairs, namely, filtering the pairs close in position but different in appearance. Compared with using a constant threshold of appearance similarity, the proposed distance ranking is more robust. In the second stage, we re-match those unmatched detections and tracks based on the appearance affinity in a gating range provided by position distance. In general, our proposed tracking strategy facilitates original position-only tracking association to apply appearance clues, which explicitly reduces mismatches caused by position ambiguity.

TABLE III. Comparison with other appearance representations on the nuScenes validation set. To be fair, we apply our appearance embedding, the state-of-the-art 2D feature extractor [28] and 3D extractor [6] based on the same LiDAR-only detector [1]. We compare the discriminative power of appearance embedding in ATPR and the tracking performance in AMOTA.

APP	Det	ATPR↑(%)	AMOTA↑(%)
-	CenterPoint	-	63.72
AlignedReID [28]	CenterPoint	66.92	54.56
PointNet [6]	CenterPoint	41.94	51.82
AlphaTrack (ours)	CenterPoint	**92.68**	**64.93**

IV. Experiments

In this section, we describe the dataset, implementation details and evaluation results of our proposed method.

A. Dataset

We evaluate our method on the large-scale driving benchmark dataset nuScenes [9]. It is annotated with 3D bounding boxes for 1000 20-second scenes at 2Hz, resulting in 28130 samples for training, 6019 samples for validation, and 6008 samples for testing. Among the full autonomous vehicle data suite, we exploit the LiDAR point clouds and RGB images from all 6 cameras. In addition to assigning the bounding box annotations for the detection training, we assign instance IDs to object individuals for appearance representation training. As the 3D MOT challenge evaluates the performance of 7 classes: cars, trucks, buses, trailers, pedestrians, motorcycles, and bicycles, the overall instance numbers are too large to cover. Further, there exists a severe class imbalance. To deal with those problems, we follow the class-balanced grouping strategy proposed by CBGS [31]. Specifically, we split 8 classes into 5 groups: (Car), (Truck), (Bus, Trailer), (Motorcycle, Bicycle), (Pedestrian), putting the classes of discrepant shapes or sizes together. We assign the ground truth ID labels to each instance in each group respectively.

B. Implementation Details

As for the input size, we set the detection range to $[-51.2m, 51.2m]$ for the X and Y axis, $[-5m, 3m]$ for Z axis, following the nuScenes evaluation guideline. To extract point-wise image features, we employ the pre-trained DLA-34 in [32] as extractor. The input image is resized to 1600×900 and the feature map has a resolution of 400×225.

TABLE IV. Evaluation results on the nuScenes test set. Our AlphaTrack achieves remarkable performance gains over the state-of-the-art 3D MOT trackers, which ranks first on the nuScenes tracking learderboard to date.

Method	Bicycle	Bus	Car	Motor	Ped	Trailer	Truck	AMOTA↑(%)	AMOTP↓(%)	FP↓	FN↓	IDS↓
StanfordIPRL-TRI [21]	25.5	64.1	71.9	48.1	74.5	49.5	51.3	55.0	79.8	17353	33216	950
CenterPoint-single [1]	32.1	71.1	82.9	59.1	76.7	65.1	59.9	63.8	55.5	18612	22928	760
EagerMOT	**58.3**	74.1	81.0	62.5	74.4	63.6	59.7	67.7	**55.0**	17705	24925	1156
Octopus-Traker	41.2	74.5	83.2	69.4	**79.0**	64.5	63.5	67.9	56.2	**16971**	22272	781
AlphaTrack (ours)	47.1	**74.9**	**84.2**	**74.2**	78.3	**70.1**	**64.2**	**70.4**	57.5	18247	**21126**	**718**

TABLE V. Comparison with other tracking methods on the nuScenes validation set. All trackers listed above attempt to consider both motion and appearance in tracking association with different models. We compare with their published AMOTA, where [30] only provides result of car and [7] only provides overall result.

Method	Modality	Tracking Association	Overall	Car
Uncertainty [6]	3D	Motion+Appearance	59.4	75.0
PnPNet [30]	3D	Motion+Appearance	-	81.5
GNN3DMOT [7]	2D + 3D	Motion+Appearance	29.9	-
Probabilistric [5]	2D + 3D	Motion+Appearance	68.7	84.3
AlphaTrack(ours)	2D + 3D	Motion+Appearance	**73.3**	**85.3**

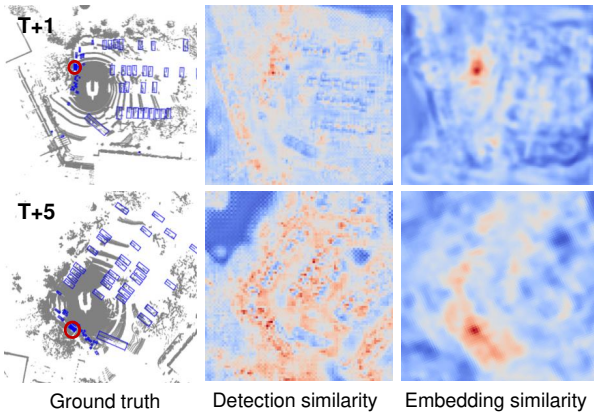

Fig. 4. Visualization of the retrieval performance of the detection feature map and the appearance embedding feature map in bird-eye-view. The target object is located in red circle, which is specified in Frame T and searched in the following two sample frames. The appearance embedding feature map is aware of the instance-level difference to distinct the target object, while the detection feature map is object-agnostic so that its response map is unable to distinct instances.

We set the stride of 3D backbone to 8, producing the output resolution of 128×128. We limit the maximum number of voxels to $60k$ for sparse convolution. We freeze the parameters of DLA-34 backbone during the training process. Following [1], we optimize the model using the adam [33] optimizer with the one-cycle learning rate policy [34], with max learning rate $1e-3$, weight decay 0.01 and momentum 0.85 to 0.95. For association details, we apply the greedy algorithm and set an upper bound threshold to solve the bipartite matching problem. In the velocity-based method, we set center distance threshold $\tau_p = 5$. In the Kalman filter-based method, we set Mahalanobis distance threshold $\tau_p = 11$. We set appearance rank rate threshold as $\lambda_{rank} = 0.4$ for filtering and set the appearance distance threshold $\tau_a = 0.4$ for re-matching. The running time is on average in total 254ms, 250ms for the network inference and 4ms for the tracking association, on the GeForce GTX 1080 Ti with a single thread i7-9700 CPU@3.60GHz.

C. Metrics

We evaluate three aspects of performance for our method: detection accuracy, the discriminatory ability of the produced appearance embedding, and the tracking performance. To evaluate detection accuracy, we compute the average precision (mAP). To evaluate the discriminative power, we extract appearance embedding of all ground truth instances over the whole validation sets of nuScenes, then apply 1 to N retrieval among all these instances and report the true positive rate at false accept rate 0.1 (TPR@FAR=0.1). For the tracking performance, we follow the evaluation tool in nuScenes Tracking Challenge [9], which applies average multi-object tracking accuracy (AMOTA) as the main evaluation metric and includes other standard metrics.

D. Ablations Studies

We conduct ablation studies to evaluate the three key components of our method: cross-modal fusion scheme, joint embedding representation and tracking association mechanisms. For generalization, our experiments based on two different kinds of motion prediction models: velocity model and Kalman filter model. The velocity model tends to provide more accurate prediction by regressed output, but it needs groudtruth for training and not always available. By contrast, the Kalman filter model is applicable to all datasets, which predicts and updates the tracks with a linear motion model.

We evaluate the detection and tracking accuracy for each tracking class on the nuScenes validation set in Table I, where a and e respectively show the results of baseline LiDAR-only method [1] and the final results of our proposed method. For simplicity, we only compare the tracking results based on velocity-based model in Table I. Comparing b and c, replacing the segmentation scores that proposed by [8] with our image features achieves overall gains of +3.52% mAP and +4.56% AMOTA. It illustrates that the fine-grained image features provide richer appearance information. Comparing c and d, our late fusion mechanism achieves gains of +2.34% mAP and +1.59% AMOTA by flattening LiDAR feature and camera feature to BEV maps separately instead of simple early concatenation. This demonstrates the superiority of our late fusion method that considers the modality differences. Comparing e and f, the simply uniformly training with both tasks will hurt both detection and tracking performance, while our alternative training strategy enables the jointly appearance embedding representation without hurting

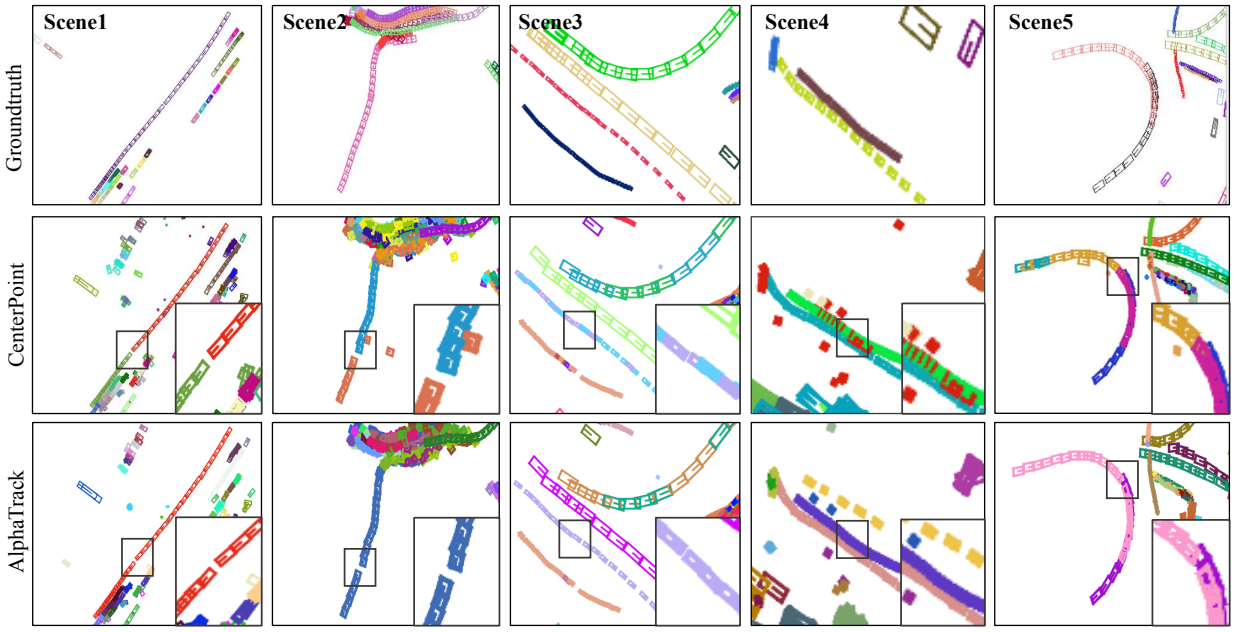

Fig. 5. Visualization of the tracking results in bird's eye view. Results show that our tracking association makes full use of appearance information to distinct objects, achieving much smaller ID switches than CenterPoint does. To make fair comparison on the association effect, we enhance the original CenterPoint detector by using our cross-modal fusion scheme but leaving the embedding branch.

detection and achieves gains of +0.11% mAP and +0.94% AMOTA. The tracking improvement mainly comes from our proposed tracking association mechanisms that applies the appearance affinity with original position affinity.

We further ablate the tracking mechanisms in Table II to show how our method works explicitly. We compare our proposed mechanism with two simple fusion mechanisms: summation mechanism and convolutional fusion mechanism. The summation mechanism sums up the position affinity and appearance affinity as final affinity by weights to associate, which slightly improves the tracking performance. We implement convolutional fusion mechanism by apply a convolutional layer that fuse embedding feature with position vector to compute affinity, which fails to enhance the tracking performance. That demonstrate that those fusion methods can not apply the complementary information from position and appearance reasonably. Our method exploits appearance clues explicitly with two mechanisms, i.e., filtering mechanism and re-matching mechanism. The pure filtering mechanism deletes false matching but do harm to IDS since the tracks are fragmented as shown in $d1$ and $d2$. The re-matching mechanism improves both AMOTA and IDS by adding the lost pairs and enhancing continuity of the trajectory, as shown in $e1$ and $e2$. The cooperation of these two mechanisms improves the overall performance greatly.

Notably, the performance gain is related to the position association baseline that differs in two motion models. Generally, Kalman filter makes coarse motion prediction, resulting in lower baseline accuracy. So it achieves greater performance gain, i.e., +5.00% AMOTA increase for car and +2.81% increase for pedestrian. However, the velocity motion model tends to fail in some categories such as Bicycle, whose velocity are irregular, and our tracking algorithm achieves +2.42% AMOTA increase. We only report the overall performance without expansion of each class due to the space limitation.

E. Quantative Results

We evaluate the tracking results on the nuScenes test set and report the first place performance to date, as shown in Table IV. Compared with our baseline model CenterPoint [1], our method achieves substantial gains in all categories of AMOTA and remarkable drops in the IDS. We also outperform in metrics including FP and FN that indicate detection accuracy. Our method surpasses the second place method on metrics including AMOTA and IDS, although inferior in FP. Our algorithm thus achieves more robust tracking.

We fairly compare our proposed embedding representation with other feature representation methods in Table III. The method [28] extracts camera feature from image patches and [6] extract LiDAR feature from LiDAR points cropped from bounding boxes. Although these approaches apply extra models to produce instance-level feature for each detection, their representations show worse discriminative power than our joint representation and fail to enhance the tracking association. Besides, our embeddings are jointly produced with detections, which takes only 4ms for tracking association, while [28] and [6] take 77ms and 24ms association time respectively for extracting the instance-aware features.

Prior methods have also tried to consider both motion and appearance changes in 3D MOT. We compare with them in Table V. Results show that our method stands out by a large gain. Note that [21] also applies CenterPoint [1] as detector, but it introduces 2D camera feature with a separate model independent of detection. Our method introduces 2D camera

feature to enhance both detection and tracking with a jointly learnable model, achieving greater performance gains.

F. Qualitative Results

In Fig. 5, we showcase some qualitative tracking results to visualize how our proposed appearance embedding assist tracking association. We compare our method with augmented CenterPoint, namely a position-only tracker equipped with the same data fusion scheme as ours. So the comparison fairly illustrates the validation of our tracking pipeline that enhanced by proposed embedding. In scene 1 and scene 2, the position-only tracker fails to associate the same object due to the missed detections, our method succeeds to recover the lost objects. In scene 3 and scene 4, the position-only tracker is challenged by the noisy detections, resulting in unstable tracks, while our method eliminates the continuous ID switches. In scene 5, our method overcomes the ID switch, where the car is under a sharp turn.

We further visualize the retrieval performance of the detection feature map and the embedding feature map in bird's eye view in Fig. 4. We select the target object in Frame T and search it in the next few frames by computing the cosine similarity between the target feature and the searched object feature maps. The output response maps demonstrate that our proposed appearance embedding is capable to distinguish objects, while the detection feature map is instance-agnostic.

V. Conclusion

In this paper, we propose a conceptually simple yet effective algorithm, named AlphaTrack, which performs 3D object detection and tracking jointly by considering both position and appearance changes through space and time. We propose to equip the state-of-the-art LiDAR-only 3D detector with cross-modal fusion scheme and instance-aware appearance embedding. We further propose a novel tracking pipeline to fully exploit the appearance clues with position clues. We demonstrate the new state-of-the-art 3D tracking results of AlphaTrack on the nuScenes benchmark. Our method not only provides favorable detection predictions but also improves the tracking performance with the jointly learned appearance feature embedding. As the associated trajectories have the potential to smooth the unstable detections and re-detect the lost objects, we consider feeding the associated results back for detection learning as a promising future work.

Acknowledgements. This work was supported by NSFC (61906119, U19B2035), Shanghai Municipal Science and Technology Major Project (2021SHZDZX0102), and Shanghai Pujiang Program.

References

[1] T. Yin, X. Zhou, and P. Krähenbühl, "Center-based 3d object detection and tracking," *CoRR*, 2020.
[2] Z. Wang, L. Zheng, Y. Liu, and S. Wang, "Towards real-time multi-object tracking," *CoRR*, 2019.
[3] Y. Zhang, C. Wang, X. Wang, W. Zeng, and W. Liu, "A simple baseline for multi-object tracking," *CoRR*, 2020.
[4] Z. Lu, V. Rathod, R. Votel, and J. Huang, "Retinatrack: Online single stage joint detection and tracking," in *CVPR*, 2020.
[5] H.-k. Chiu, J. Li, R. Ambrus, and J. Bohg, "Probabilistic 3d multi-modal, multi-object tracking for autonomous driving," *CoRR*, 2020.
[6] J. Wang, S. Ancha, Y. Chen, and D. Held, "Uncertainty-aware self-supervised 3d data association," *CoRR*, 2020.
[7] X. Weng, Y. Wang, Y. Man, and K. M. Kitani, "GNN3DMOT: graph neural network for 3d multi-object tracking with 2d-3d multi-feature learning," in *CVPR*, 2020.
[8] S. Vora, A. H. Lang, B. Helou, and O. Beijbom, "Pointpainting: Sequential fusion for 3d object detection," in *CVPR*, 2020.
[9] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *CVPR*, 2020.
[10] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," in *CVPR*, 2019.
[11] Y. Zhou and O. Tuzel, "Voxelnet: End-to-end learning for point cloud based 3d object detection," in *CVPR*, 2018.
[12] G. P. Meyer, A. Laddha, E. Kee, C. Vallespi-Gonzalez, and C. K. Wellington, "Lasernet: An efficient probabilistic 3d object detector for autonomous driving," in *CVPR*, 2019.
[13] Y. Wang, W. Chao, D. Garg, B. Hariharan, M. E. Campbell, and K. Q. Weinberger, "Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving," in *CVPR*, 2019.
[14] N. Mahmoudi, S. M. Ahadi, and M. Rahmati, "Multi-target tracking using cnn-based features: CNNMTT," *Multim. Tools Appl.*, 2019.
[15] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander, "Joint 3d proposal generation and object detection from view aggregation," in *IROS*, 2018.
[16] M. Liang, B. Yang, S. Wang, and R. Urtasun, "Deep continuous fusion for multi-sensor 3d object detection," in *ECCV*, 2018.
[17] Z. Wang, W. Zhan, and M. Tomizuka, "Fusing bird's eye view LIDAR point cloud and front view camera image for 3d object detection," in *IVS*, 2018.
[18] V. A. Sindagi, Y. Zhou, and O. Tuzel, "Mvx-net: Multimodal voxelnet for 3d object detection," in *ICRA*, 2019.
[19] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. V. Gool, "Online multiperson tracking-by-detection from a single, uncalibrated camera," *TPAMI*, 2011.
[20] X. Weng, J. Wang, D. Held, and K. Kitani, "AB3DMOT: A baseline for 3d multi-object tracking and new evaluation metrics," *CoRR*, 2020.
[21] H. Chiu, A. Prioletti, J. Li, and J. Bohg, "Probabilistic 3d multi-object tracking for autonomous driving," *CoRR*, 2020.
[22] Y. Wang, S. Chen, L. Huang, R. Ge, Y. Hu, Z. Ding, and J. Liao, "1st place solutions for waymo open dataset challenges - 2d and 3d tracking," *CoRR*, 2020.
[23] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in *ICIP*, 2017.
[24] F. Yu, W. Li, Q. Li, Y. Liu, X. Shi, and J. Yan, "POI: multiple object tracking with high performance detection and appearance feature," in *ECCV*, 2016.
[25] Z. Zhou, J. Xing, M. Zhang, and W. Hu, "Online multi-target tracking with tensor-based high-order graph matching," in *ICPR*, 2018.
[26] P. Bergmann, T. Meinhardt, and L. Leal-Taixé, "Tracking without bells and whistles," in *ICCV*, 2019.
[27] E. Ristani and C. Tomasi, "Features for multi-target multi-camera tracking and re-identification," in *CVPR*, 2018.
[28] X. Zhang, H. Luo, X. Fan, W. Xiang, Y. Sun, Q. Xiao, W. Jiang, C. Zhang, and J. Sun, "Alignedreid: Surpassing human-level performance in person re-identification," *CoRR*, 2017.
[29] P. Voigtlaender, M. Krause, A. Osep, J. Luiten, B. B. G. Sekar, A. Geiger, and B. Leibe, "MOTS: multi-object tracking and segmentation," in *CVPR*, 2019.
[30] M. Liang, B. Yang, W. Zeng, Y. Chen, R. Hu, S. Casas, and R. Urtasun, "Pnpnet: End-to-end perception and prediction with tracking in the loop," in *CVPR*, 2020.
[31] B. Zhu, Z. Jiang, X. Zhou, Z. Li, and G. Yu, "Class-balanced grouping and sampling for point cloud 3d object detection," *CoRR*, 2019.
[32] X. Zhou, V. Koltun, and P. Krähenbühl, "Tracking objects as points," in *ECCV*, A. Vedaldi, H. Bischof, T. Brox, and J. Frahm, Eds., 2020.
[33] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *ICLR*, 2019.
[34] L. N. Smith, "A disciplined approach to neural network hyper-parameters: Part 1 - learning rate, batch size, momentum, and weight decay," 2018.