

Supplementary Material of “Standing Between Past and Future: Spatio-Temporal Modeling for Multi-Camera 3D Multi-Object Tracking”

Ziqi Pang¹, Jie Li², Pavel Tokmakov², Dian Chen², Sergey Zagoruyko³, Yu-Xiong Wang¹
University of Illinois Urbana-Champaign¹, Toyota Research Institute², Woven Planet Level-5³

Our appendix describes the additional experimental analysis and implementation details. The catalog is as below:

- (A) **Video demo.** We provide a demo for multi-camera 3D multi-object tracking (MOT) as explained in Sec. A.
- (B) **Implementation details.** We explain the detailed model architecture, procedures for training and inference, and settings for ablation studies in Sec. B.
- (C) **Additional ablations.** We provide more analysis and experimental results in Sec. C.
- (D) **Limitations and future work.** We discuss the limitations of PF-Track and list interesting future directions.
- (E) **Performance Verification.** For checking the results, we provide the screenshot of the test split results for verification in Sec. E.

A. Multi-camera Tracking Video Demo

Please check out our demo video at <https://youtu.be/eJghONb2AGg>. It contains:

- Visualization of 3D MOT results on both surrounding images and Bird’s-eye-view.
- Illustration for addressing occlusions.
- Qualitative results for predicted trajectories.

B. Implementation Details

B.1. Model Architecture

We explain the design choices of PF-Track in the sections below. To provide a high-level view of the model, we enclose the config file in mmdetection3d [4] format in this supplementary material.

Backbone. We use VoVNetV2 [8] as backbone. For the feature pyramid [9], the C5 feature (output of the 5-th stage) is upsampled and fused with C4 feature (output of the 4-th stage). To save GPU memory during training, we adopt the checkpointing trick [3] by default.

Detection head. We follow the design of PETR [10] by setting the region to $[-51.2m, 51.2m]$ on the XY-axis and $[-5m, 3m]$ on the Z-axis. The centers of bounding

boxes are normalized to $[0, 1]$, respectively. The detection head composes of 6 transformer decoder layers [14] and 2 MLP heads for bounding box regression and classification. Each transformer decoder layer has an embedding dimension of 256 and a feedforward dimension of 2048. The dropout probability is 0.1. The MLP heads are both two-layer MLPs. The bounding box regression head predicts the centers (x, y, z) , sizes (l, w, h) , orientation, and velocities of objects, and the classification head returns the logits for every category.

PF-Track configurations. Our PF-Track uses a fixed number of 500 detection queries per frame. As for track queries, the training and inference procedures are different. Training protocol adds queries into the set of track queries once they become a positive match to the ground truth ($< 2.0m$), and a tracking query is kept associated with the ground truth of the same object. The inference protocol initializes the track queries if the corresponding confidence score is larger than 0.4. During the inference time, our model output at most 300 objects (same as [10, 15, 18]) and set a minimum score threshold of 0.2.

Past reasoning. The length of the query queue is 1.5 seconds ($\tau_h = 3$ frames) because each of our training samples has 3 frames. However, we emphasize that PF-Track is able to aggregate historical information from the entire video during the *inference* time because the past reasoning module is recurrent. Concretely, the queries at frame t attend to frames $[t - 2, t]$, and the queries at frame $t - 2$, in turn, attend to frames $[t - 4, t - 2]$, and so on. Thus, the queries at frame t have access to information from all the previous frames.

For cross-frame and cross-frame attention, we employ two transformer decoder layers for them each. Every transformer decoder layer has an embedding dimension of 256, and a feedforward dimension of 2048. Same as the transformer layers in the detection head, these two layers also have a dropout probability of 0.1. The track refinement module uses two separate 2-layer MLP heads for regression

Algorithm A Algorithm for “Query Propagation.”

Input:

$\mathbf{Q}_{t-1}, \mathbf{C}_{t-1}, \mathbf{M}_{t-1:t+\tau_f-1}^{t-1}$: queries, center positions, and motion predictions of objects from frame $t - 1$;
 N : number of queries.

Output:

$\hat{\mathbf{Q}}_{t-1}, \hat{\mathbf{C}}_{t-1}, \hat{\mathbf{M}}_{t:t+\tau_f}^{t-1}$: propagated queries, center positions, and motion predictions of objects from frame $t - 1$ to frame t .

```

1: for  $i = 0$  to  $N - 1$  do
2:    $\hat{\mathbf{C}}_{t-1}^i \leftarrow \mathbf{C}_{t-1}^i + \mathbf{M}_{t-1:t}^{t-1,i}$ 
3:    $\hat{\mathbf{Q}}_{t-1}^i \leftarrow \mathbf{Q}_{t-1}^i$ 
4:    $\hat{\mathbf{M}}_{t:t+\tau_f}^{t-1} \leftarrow \text{Padding}(\mathbf{M}_{t:t+\tau_f-1}^{t-1})$ 
5: end for
6: return  $\hat{\mathbf{Q}}_{t-1}, \hat{\mathbf{C}}_{t-1}, \hat{\mathbf{M}}_{t:t+\tau_f}^{t-1}$ 

```

and classification.

Future reasoning. The future reasoning module predicts the movements to future 4.0 seconds ($\tau_f = 8$ frames). It first uses the same cross-frame attention to generate the future features, then applies a 2-layer MLP to translate features into the movements on the XY plane. For track extension, we pick the extension length that maximizes the AMOTA on the validation split, which are 2.0s (4 frames) for the small resolution model and 2.5s (5 frames) for the full resolution model.

Loss weights. In Sec. 3.4 of the main paper, we use coefficients to balance the loss terms. The bounding box regression loss is $\lambda_{box}^D = 0.25$, and the classification focal loss is $\lambda_{cls}^D = 2.0$, which are the same as [10]. For the track refinement part, we adopt the same loss weights for bounding box regression and classification, respectively: $\lambda_{box}^R = 0.25$ and $\lambda_{cls}^R = 2.0$. The weight for motion prediction in future reasoning is $\lambda_f = 0.5$.

B.2. Algorithm for “Track Extension”

We clarify the detailed steps for the track extension algorithms described in Sec. 3.3 (main paper). For best clarity, we rigorously describe query propagation in Algorithm A first and then introduce track extension in Algorithm B.

As in Algorithm A, the propagated center positions $\hat{\mathbf{C}}_{t-1}$ come from adding current states \mathbf{C}_{t-1} with motion predictions $\mathbf{M}_{t-1:t}^{t-1}$. The propagation of queries $\hat{\mathbf{Q}}_{t-1}$ directly reuse the latest results \mathbf{Q}_{t-1} . The propagation of motion predictions can be simple padding functions, as our track extension does not exceed the length of motion prediction.

Then we illustrate the algorithm after adding track extension in Algorithm B. The major difference lies in address-

Algorithm B Algorithm for “Track Extension.”

Input:

$\hat{\mathbf{Q}}_{t-1}, \hat{\mathbf{C}}_{t-1}, \hat{\mathbf{M}}_{t:t+\tau_f}^{t-1}$: propagated queries, center positions, and motion predictions from frame $t - 1$;
 \mathbf{F}_t : image features on frame t ;
 L_t : how many frames have the queries been extended continuously;
 τ_e : maximum frames for extension;
 N : number of queries.

Output:

$\hat{\mathbf{Q}}_t, \hat{\mathbf{C}}_t, \hat{\mathbf{M}}_{t+1:t+\tau_f+1}^t$: propagated queries, center positions, and motion predictions of tracked objects from frame t to frame $t + 1$.

```

1:  $\mathbf{Q}_t^R, \mathbf{C}_t^R, \mathbf{M}_{t:t+\tau_f}^t \leftarrow \text{PF-Track}(\mathbf{F}_t, \mathbf{Q}_t, \mathbf{C}_t)$ 
2: for  $i = 0$  to  $N - 1$  do
3:   if confidence score  $S_t^{R,i}$  is below a threshold then
4:     if  $i$  is an active object  $L_t^i < \tau_e$  then
5:       Then use previous information by Line 6-8:
6:        $\hat{\mathbf{C}}_t^i \leftarrow \hat{\mathbf{C}}_{t-1}^i + \hat{\mathbf{M}}_{t:t+1}^{t-1,i}$ 
7:        $\hat{\mathbf{Q}}_t^i \leftarrow \hat{\mathbf{Q}}_{t-1}^i$ 
8:        $\hat{\mathbf{M}}_{t+1:t+\tau_f+1}^t \leftarrow \text{Padding}(\hat{\mathbf{M}}_{t+1:t+\tau_f}^{t-1})$ 
9:       Record the extension:  $L_t^i \leftarrow L_{t-1}^i + 1$ 
10:    else
11:      Terminate the track of the  $i$ -th object.
12:      Remove it from memory.
13:    end if
14:  else
15:    Propagate normally as Line 15-18
16:     $\hat{\mathbf{C}}_t^i \leftarrow \mathbf{C}_t^{R,i} + \mathbf{M}_{t:t+1}^{t,i}$ 
17:     $\hat{\mathbf{Q}}_t^i \leftarrow \mathbf{Q}_t^{R,i}$ 
18:     $\hat{\mathbf{M}}_{t+1:t+\tau_f+1}^t \leftarrow \text{Padding}(\mathbf{M}_{t+1:t+\tau_f}^t)$ 
19:    Zero the continuous extension:  $L_t^i \leftarrow 0$ 
20:  end if
21: end for
22: return  $\hat{\mathbf{Q}}_t, \hat{\mathbf{C}}_t, \hat{\mathbf{M}}_{t+1:t+\tau_f+1}^t$ 

```

ing the low-confidence objects (lines 3-14), which might be noisy. Instead of always updating according to the latest results, track extension relies more on the results coming from previous frames for better fidelity (lines 5-8). Notably, if an object does not have confident results in τ_e continuous frames, we follow the common practice of terminating this track (lines 10-13). Please note that we set τ_e smaller than the prediction horizon $\tau_f = 8$ frames (4.0s) by default. As in Tab. 2, track extension effectively improves both AMOTA and IDS.

B.3. Model Training

Every training sample composes of three adjacent frames. PF-Track is trained with AdamW optimizer [7, 12] with an weight decay of 0.01. The learning rate starts from 2.0×10^{-4} and is scheduled according to cosine annealing [11]. The above process is identical to image-based 3D detection methods [10, 15]. However, as image data augmentation could break the motion models of objects, we disable the data augmentation during the training of the tracker. The total training epochs follow the settings discussed in Sec. 4.2 (main paper). The full resolution setting takes 3 days on $8 \times A100$ GPUs, and the small resolution setting takes 1 day on $8 \times A100$ GPUs.

B.4. “Tracking by Detection” Experiments

The experiments of “tracking by detection” baselines in Sec. 4.4 (main paper) tune the detection score threshold according to AMOTA on the validation split. Eventually, we set a minimum score threshold of 0.2 for output bounding boxes, which is also the same as PF-Track. According to Tab. 4 (main paper), our hyper-parameter tuning significantly improves the performance of “tracking by detection” baselines and enables a fair comparison. The reason is that AB3DMOT [16], CenterPoint [17], and SimpleTrack [13] designed their trackers for LiDAR-based 3D detection and used low (*e.g.*, 0.01 in SimpleTrack) or no score thresholds. However, image-based detection contains more false positives and requires stronger filtering.

B.5. Motion Prediction from Abstract Object States Settings

This section explains the details of the “Prediction from Query Features” in Sec. 4.4 (main paper).

Dataset construction. We construct the motion prediction dataset by recalling the true positive tracks from the output of PF-Track. Specifically, we perform Hungarian matching between the predicted bounding boxes and ground truth, and the positive matches are determined by less than 2.0m from the ground truth. We use the training/validation split of nuScenes [1] dataset and remove the frames without positive match. Eventually, we have 27,960 and 5,879 frames, and 422,167 and 69,804 tracks in the training and validation set for motion prediction.

Model training and inference. Our model architectures are the same as the LSTM model from the Argoverse [2] and VectorNet [6]. We provide 2.0s of history and require the model to predict up to 4.0s, which is the same as our PF-Track, for a fair comparison. As every frame contains multiple tracks, our implementation normalizes the coordinates with respect to the positions of ego-vehicle, following [5]. The inputs to the models include the bounding box

Model	Extension	AMOTA \uparrow	AMOTP \downarrow	IDS \downarrow
PF-Track	✓	0.391 0.408	1.387 1.343	471 166
wo/ Cross-frame	✓	0.385 0.395	1.402 1.373	410 171
wo/ Cross-object	✓	0.383 0.399	1.390 1.352	481 155

Table A. **Cross-frame and Cross-object attention.** “Extension” denotes using track extension or not. Removing either cross-frame or cross-object from the “Query Refinement” module of past reasoning (Sec. 3.2 of main paper) negatively impacts the model performance.

Model	Extension	AMOTA \uparrow	AMOTP \downarrow	IDS \downarrow
PF-Track	✓	0.391 0.408	1.387 1.343	471 166
Velo Prop	✓	0.374 0.382	1.389 1.359	478 192

Table B. **Velocity for Query Propagation.** “Extension” means using track extension. Using velocities instead of learned trajectories for query propagation in future reasoning (Sec. 3.3 of main paper) negatively impacts the model performance.

centers and velocities on the XY plane. For missing observations, we pad the input to the full length with the closest observation and indicate padding with a 0-1 mask. We train the models for 24 epochs on our custom motion prediction dataset with AdamW [7, 12] optimizer and an initial learning rate of 1.0×10^{-3} . The learning rate drops by 0.1 on the 16-th and 20-th epochs.

C. Supplemental Ablation Studies

C.1. Cross-frame and Cross-object Attention in Past Reasoning.

We analyze the effect of cross-frame and cross-object attention for “Query Refinement” in past reasoning (Sec. 3.2 of main paper). As in Tab. A, if either of them is removed from the final PF-Track, the performance decreases, especially in AMOTA. Although PF-Track has a slightly larger ID-Switch, we argue that it is due to a higher AMOTA. Therefore, both cross-frame and cross-object attentions are useful for multi-camera 3D MOT.

C.2. Trajectories or Velocity for Query Propagation

To verify the necessity of learning a trajectory prediction for query propagation, we experiment with using the velocities for propagation in Tab. B. Specifically, we train a model identical to PF-Track except using velocities to transform the positions of queries across frames. As in Tab. B, using learned trajectories has a better performance compared with the variant of using velocities. Therefore, we conclude that learning long-term trajectory prediction is necessary for robust and accurate 3D MOT.

λ_f	Extension	AMOTA \uparrow	AMOTP \downarrow	IDS \downarrow
0.25	✓	0.389 0.396	1.382 1.353	539 185
0.50	✓	0.391 0.408	1.387 1.343	471 166
1.00	✓	0.377 0.397	1.395 1.346	489 183

Table C. **Loss weights of motion prediction.** “ λ_f ” denotes the loss weight λ_f for motion prediction (Sec. 3.4 in main paper), “Extension” denotes using track extension or not. This table demonstrates the sensitivity of motion prediction and calls for the attention of future work in better multi-task learning strategies.

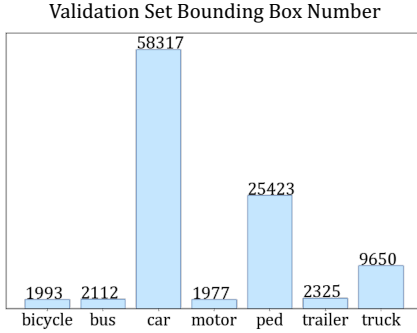


Figure A. Distribution of per-category instance numbers in nuScenes validation set. “motor” denotes “motorcycle” and “ped” denotes “pedestrian.” As clearly demonstrated, the data distribution is imbalanced on nuScenes.

C.3. Weights of Motion Prediction in Future Reasoning

We analyze the tracking performance with respect to the loss weights of motion prediction. The discovery is the sensitivity of tracking performance to the weight of the loss for motion prediction λ_f (Sec. 3.4 in main paper). In Tab. C, we vary the weight for motion prediction with $\lambda_f = [0.25, 0.50, 1.00]$, and they could cause variation in the tracking performance. This requires the attention of future works or better multi-task learning strategies

C.4. Category-level Analysis

We compare the category-level performance before and after using our past and future reasoning. To provide the context of nuScenes dataset [1], we start by visualizing the data distribution across categories in Fig. A. As clearly illustrated, nuScenes exhibits an imbalanced category distribution, and the types of “car” and “pedestrian” take up most of the objects.

Then in Tab. D, we compare the category-level performance of the three major types: car, pedestrian, and truck. Compared to the baseline of not using any past or future reasoning (same baseline as Tab. 1 of the main paper), our proposed method significantly improves upon the baseline over individual categories.

Metrics	PF-Track	Car	Pedestrian	Truck	All
AMOTA \uparrow	✓	0.562 0.579	0.372 0.415	0.374 0.403	0.358 0.408
AMOTP \downarrow	✓	1.058 1.021	1.431 1.362	1.341 1.288	1.419 1.343
IDS \downarrow	✓	368 67	154 83	25 7	507 166

Table D. **Per-category tracking metrics analysis.** “All” denotes the averaged metric numbers over all seven categories. On the three major categories on nuScenes, our PF-Track achieves significant and steady improvement over the baseline that does not involve past or future reasoning.

Metrics	PF-Track	Car	Pedestrian	Truck	All
AMOTA \uparrow	✓	0.583 0.600	0.370 0.431	0.305 0.310	0.344 0.362
AMOTP \downarrow	✓	1.038 1.001	1.436 1.338	1.399 1.366	1.419 1.363
IDS \downarrow	✓	394 172	222 116	32 5	680 300

Table E. **PF-Track for DETR3D detection head.** “All” denotes the averaged metric numbers over all seven categories, “PF-Track” denotes using the past and future reasoning from PF-Track. On the three major categories and average metrics on nuScenes, PF-Track is able to enhance the tracking performance.

C.5. PF-Track with DETR3D

We provide supplemental analysis of applying PF-Track to a different 3D detector: DETR3D [15].

Experiment setup. Following DETR3D [15], we use VoVNetV2 [8] as the backbone and fuse the features from C2-C5 with FPN [9] as the image features. As DETR3D only has a full-resolution setting, we first train the backbone with single-frame detection for 24 Epochs, then fix the backbone and train the whole tracker with 3-frame tracking for another 24 Epochs. A cycle of training takes 3 days on $8 \times$ A100 GPUs.

Efficacy of PF-Track. We analyze the performance without or with the joint past and future reasoning in PF-Track in Tab. E. We include the results for top-3 major categories and the average of all categories. Our past and future reasoning also significantly improves the AMOTA and decreases ID-Switches. This result indicates the generalizability of PF-Track on other query-based detectors.

D. Limitations and Future Works

Processing HD-Map for End-to-end Forecasting. The main focus of PF-Track is 3D MOT, and it is not a full-fledged motion prediction pipeline because of not consider HD-Maps. However, with the improved track quality, our past and future reasoning could be beneficial to downstream motion prediction. Therefore, potential future work is to

```

Calculating metrics...
Saving metrics to: /tmp/tmp_bimu3cd/nuscenes-metrics

### Final results ###

Per-class results:

```

	AMOTA	AMOTP	RECALL	MOTAR	GT	MOTA	MOTP	MT	ML	FAF	TP	FP	FN	IDS	FRAG	TID	LGD
bicycle	0.322	1.442	0.387	0.762	2186	0.295	0.635	46	102	14.7	845	201	1341	0	8	1.48	1.87
bus	0.408	1.257	0.544	0.605	1701	0.329	0.720	36	33	25.3	924	365	776	1	21	1.92	2.48
car	0.622	0.916	0.719	0.777	68518	0.558	0.520	2443	1371	186.6	49194	10956	19225	99	415	0.87	1.19
motorcy	0.448	1.245	0.457	0.840	1945	0.384	0.561	41	87	11.5	888	142	1057	0	5	1.31	1.61
pedestr	0.451	1.279	0.541	0.736	34010	0.395	0.738	734	873	96.9	18250	4827	15618	142	273	1.52	2.16
trailer	0.380	1.421	0.528	0.636	2566	0.334	0.884	57	64	48.7	1350	492	1212	4	23	0.99	1.61
truck	0.405	1.206	0.592	0.596	8639	0.352	0.658	191	178	51.1	5107	2065	3529	3	94	1.11	2.07

```

Aggregated results:
AMOTA 0.434
AMOTP 1.252
RECALL 0.538
MOTAR 0.707
GT 17080
MOTA 0.378
MOTP 0.674
MT 3548
ML 2708
FAF 62.1
TP 76558
FP 19048
FN 42758
IDS 249
FRAG 839
TID 1.32
LGD 1.86
Eval time: 5146.9s

Completed evaluation for test phase

```

Figure B. **Screenshot of test set result.** This is in supplementary for our results in Tab. 1 (main paper).

include HD-Map as an optional input to our method and explored end-to-end motion forecasting.

Sensor Modalities. Our current PF-Track is a general query-based framework. Therefore, we could extend PF-Track beyond the multi-camera setting and include LiDAR or Radar into our framework.

E. Test Split Screenshot

We provide the screenshot of the submission to nuScenes test set in Fig. B, as supplementary to the test set results in Tab. 1 (main paper).

References

- [1] 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 multi-modal dataset for autonomous driving. In *CVPR*, 2020. 3, 4
- [2] Ming-Fang Chang, John Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, and James Hays. Argoverse: 3D tracking and forecasting with rich maps. In *CVPR*, 2019. 3
- [3] Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. Training deep nets with sublinear memory cost. *arXiv preprint arXiv:1604.06174*, 2016. 1
- [4] MMDetection3D Contributors. MMDetection3D: OpenMMLab next-generation platform for general 3D object detection. <https://github.com/open-mmlab/mmdetection3d>, 2020. 1
- [5] Scott Ettinger, Shuyang Cheng, Benjamin Caine, Chenxi Liu, Hang Zhao, Sabeek Pradhan, Yuning Chai, Ben Sapp, Charles R Qi, Yin Zhou, et al. Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset. In *ICCV*, 2021. 3
- [6] Jiyang Gao, Chen Sun, Hang Zhao, Yi Shen, Dragomir Anguelov, Congcong Li, and Cordelia Schmid. VectorNet: Encoding HD maps and agent dynamics from vectorized representation. In *CVPR*, 2020. 3
- [7] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 3
- [8] 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 *CVPRW*, 2019. 1, 4
- [9] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017. 1, 4
- [10] Yingfei Liu, Tiancai Wang, Xiangyu Zhang, and Jian Sun. PETR: Position embedding transformation for multi-view 3D object detection. In *ECCV*, 2022. 1, 2, 3
- [11] Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. 3
- [12] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 3
- [13] Ziqi Pang, Zhichao Li, and Naiyan Wang. Simpletrack: Understanding and rethinking 3D multi-object tracking. *arXiv preprint arXiv:2111.09621*, 2021. 3
- [14] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NeurIPS*, 2017. 1

- [15] Yue Wang, Vitor Campagnolo Guizilini, Tianyuan Zhang, Yilun Wang, Hang Zhao, and Justin Solomon. DETR3D: 3D object detection from multi-view images via 3D-to-2D queries. In *CoRL*, 2022. [1](#), [3](#), [4](#)
- [16] Xinshuo Weng, Jianren Wang, David Held, and Kris Kitani. 3D multi-object tracking: A baseline and new evaluation metrics. In *IROS*, 2020. [3](#)
- [17] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Center-based 3D object detection and tracking. In *CVPR*, 2021. [3](#)
- [18] Tianyuan Zhang, Xuanyao Chen, Yue Wang, Yilun Wang, and Hang Zhao. MUTR3D: A multi-camera tracking framework via 3D-to-2D queries. In *CVPRW*, 2022. [1](#)