

Supplementary Material for SynCL

A Implementation Details

A.1 Model Details

We present the details of the model architectures and some experimental design decisions for the MUTR [1] and PF-Track [2] trackers with their corresponding variants, Baseline#1 and Baseline#2.

MUTR. In alignment with MUTR [1], we adopt DETR3D [3] as the detector, employing ResNet-101 [4] as the backbone. An FPN [5] neck module is connected to the ResNet-101 backbone. The detection head comprises six transformer decoder layers. Each decoder layer includes query-to-query self-attention and DETR3D-based query-to-image cross-attention mechanisms. The output of the cross-attention is followed by box regression and classification heads, each consisting of two-layer MLPs. All decoder layers maintain an embedding dimension of 256 and a feed-forward dimension of 512. MUTR chooses seven tracking task categories as ground-truth targets to guide the supervision of model outputs.

PF-Track. Aligned with PF-Track, we adopt PETR [6] as the detector, employing VoVNetV2-99 [7] as the backbone. The detection head follows the MUTR design, with a key modification where the PETR-based query-to-image cross-attention incorporates 3D position embedding. All decoder layers maintain an embedding dimension of 256, while the feed-forward dimension is 2048. Unlike MUTR, PF-Track supervises all ten categories as ground-truth targets for the model’s output. During the inference stage of tracking, seven tracking task categories are selected for evaluation.

Baseline#1. We establish the Baseline#1 of PETRv2 [8] by substituting only the detector of PF-Track. Compared to PETR, PETRv2 combines the temporal information of intermediate frames with key frames and enhances the 3D position embedding for different frames. We keep the refinement module and motion prediction module in the PF-Track [2] framework for Baseline#1.

Baseline#2. We establish the Baseline#2 with StreamPETR detector in MUTR3D framework. StreamPETR uses 644 initialized object queries in each frame and propagates the Top-256 object queries into the next frame. For the tracking task, we prioritize track queries with persistent IDs, selecting them first as the propagated queries for subsequent frames. The remaining top object queries are then chosen to complete the set of 256 propagated queries. During training, we supervise all ten categories, while seven tracking task categories are selected for evaluation during the inference stage of tracking.

A.2 Model Training

Following [3, 6, 8, 9], we employ Focal Loss [10] for classification scores and ℓ_1 -loss for box regression. Each training batch consists of a clip of $T = 3$ consecutive frames. The loss for object queries is computed from the first frame, while the contrastive loss and the prediction loss for track queries are computed from the second frame onward. For brevity, we use the subscript $1 \rightarrow 1$ and $1 \rightarrow m$ to denote the one-to-one and the one-to-many label assignment, while S and C denote S-decoder and C-decoder, respectively. The loss for the entire clip is formulated as:

$$\mathcal{L}_{trk} = \mathcal{L}_{1 \rightarrow 1}^S + \mathcal{L}_{1 \rightarrow 1}^C, \mathcal{L}_{obj} = \mathcal{L}_{1 \rightarrow 1}^S + \mathcal{L}_{1 \rightarrow m}^C \quad (A1)$$

$$\mathcal{L} = \sum_{t=1}^T \lambda_{obj} \mathcal{L}_{obj}^t + \sum_{t=2}^T (\lambda_{trk} \mathcal{L}_{trk}^t + \lambda_{CL} \mathcal{L}_{CL}^t) \quad (A2)$$

where λ_{obj} , λ_{trk} , and λ_{CL} denote the loss coefficients for detection, tracking, and contrastive learning, respectively, all of which are set to 1.0 by default.

A.3 Experimental setup in Fig. 2 and Fig. 3

We provide more implementation details for Fig.2 and Fig.3 as shown in Tab. A1. We claim that the results of Fig. 2 and Fig. 3 are from the model trained without utilizing our proposed SynCL.

Table A1: Experimental setup in Fig. 2 and Fig. 3

Method	Backbone	Detector	Resolution	w/ SynCL	Dataset	Scene token	frame ID
PF-Track	V2-99	PETR	320×800	No	nuScenes	3927699fa	1874

B Supplemental Experiments

B.1 Runtime Analysis

In Tab. A2, we compare the inference speed across various tracking methods listed in Tab. 1. SynCL employs only the standard decoder during inference stage, thus not adding any extra computation. An end-to-end MOT pipeline typically comprises three stages: model prediction, motion model updating, and instance life cycle management. The model prediction stage performs inference on the GPU, while the remaining stages primarily rely on the CPU. Different tracking frameworks (e.g., MUTR3D and PF-Track) have distinct design across these three stages. FLOPs reflect only the computational complexity during the model prediction phase on the GPU. The FPS is measured from the input images to the final tracking results, providing a more comprehensive consideration of the inference speed of the entire end-to-end pipeline. Based on Baseline#2 with StreamPETR detector in MUTR3D framework, our SynCL exhibits superior performance and inference speed.

Table A2: Runtime and model complexity analysis. The FPS is measured on a single NVIDIA A100 GPU, from the input images to the final tracking results.

Method	Backbone	Detector	Resolution	#Param.	FLOPs	FPS
MUTR3D	R101	DETR3D	900×1600	54.6 M	1016G	4.5
SynCL (ours)	R101	DETR3D	900×1600	54.6 M	1016G	4.5
PF-Track	V2-99	PETR	320×800	91.8 M	539 G	7.5
SynCL (ours)	V2-99	PETR	320×800	91.8 M	539 G	7.5
Baseline#1	V2-99	PETRv2	320×800	95.4 M	1039 G	6.7
SynCL (ours)	V2-99	PETRv2	320×800	95.4 M	1039 G	6.7
Baseline#2	V2-99	Stream	320×800	82.6 M	518 G	9.4
SynCL (ours)	V2-99	Stream	320×800	82.6 M	518 G	9.4

B.2 Extend Analysis of Dynamic Query Filtering

We visualize the assignment cost distributions of the query filtering methods listed in Tab. 4b as shown in Fig. A1, using data from the first 1,000 training iterations of PF-Track at the resolution of 320×800 for clarity. A lower matching cost coincides with a higher number of boxes, indicating the acquisition of more abundant and high-quality matching pairs. Although IoU-based query filtering methods (ATSS and SimOTA) focus on obtaining high-quality matching pairs, they are limited by their relatively low matching quantity. Compared to the cost-based query filtering method DETA, our GMM-based dynamic query filtering method can mine more high-quality positive samples based on the model’s optimization status, considering both the classification score and box quality.

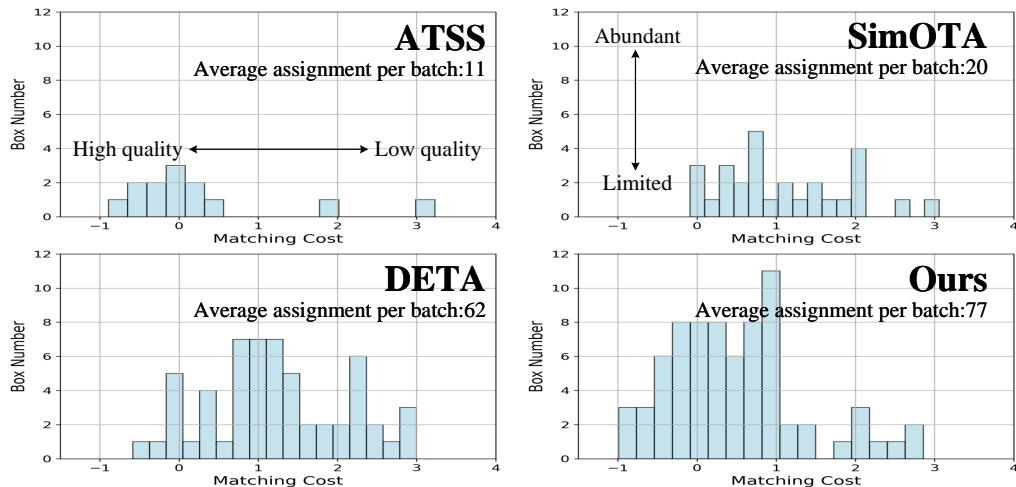


Figure A1: The assignment cost distribution of the query filtering methods within a batch.

B.3 Ablation Studies

Gaussian Mixture Model Components Number. As shown in Tab.A3, we compare the performance by varying the number of components in the GMM. Simply dividing object queries into the reliable group and the unreliable group achieves the highest performance. Increasing the number of components tends to lower the dynamic threshold, resulting in more stringent filtering that reduces the supervision of promising candidates. We thus use $N_c = 2$ as the default setting.

Variants of Kernel-based Contrastive Learning. In Tab.A4, we explore several variants within the kernel-based contrastive learning (CL) design and use the experiment without CL supervision as the baseline (①). **1):** We first remove the FFN for object queries. The FFN projection helps align inter-temporal track queries with intra-temporal object queries, thereby enhancing the quality of both representations (②vs.④). **2):** We apply the contrastive learning to the output of each decoder layer, but find that it does not further improve model performance. We infer that too many sources of CL supervision make model training unstable (③vs.④).

Table A3: Ablation of the Gaussian Mixture Model components number N_c .

N_c	Tracking			Detection	
	AMOTA	AMOTP↓	Recall	NDS	mAP
2	44.7%	1.262	56.5%	49.7%	39.6%
3	44.3%	1.286	54.9%	49.2%	39.4%
4	41.9%	1.312	53.2%	49.0%	38.7%

Table A4: Comparison results of different kernel-based instance-aware contrastive learning designs.

Exp	Tracking			Detection	
	AMOTA	AMOTP↓	Recall	NDS	mAP
①	42.6%	1.286	54.6%	49.9%	39.9%
②	44.2%	1.282	55.3%	49.5%	39.2%
③	43.1%	1.303	54.3%	49.7%	39.1%
④	44.7%	1.262	56.5%	49.7%	39.6%

The necessity of removing the self-attention mechanism. As shown in Tab. A5, we apply one-to-many matching to object queries in the standard decoder (S-decoder) without constructing the parallel decoder, but this results in a decrease in model performance (①vs.②). We further try to expand the matching range of object queries to include all GTs while still keeping the self-attention mechanism in the only standard decoder. However, both the detection and tracking performance of the model suffered significantly (②vs.③). This shows that allowing object queries and track queries to match the same ground truth directly in the standard decoder worsens the query competition rather than improving it. Finally, we retain the self-attention mechanism in the parallel decoder. Compared to the baseline (④vs.⑤), this also leads to a severe decrease in model performance, demonstrating that the query competition issues remain with the existence of the self-attention mechanism. These experimental results indicate that it is challenging to address the problems caused by the self-attention mechanism and bipartite matching together without removing the self-attention mechanism.

Table A5: Comparison results of matching variants for object queries in Task-specific Hybrid Matching module. PDAM denotes the parallel decoder’s attention mechanism.

#	Decoder Type	Matching Range	Strategy	PDAM	Tracking		Detection	
					AMOTA	AMOTP↓	NDS	mAP
①	S-decoder	New-born GTs	one-to-one	w/o parallel decoder	40.8%	1.343	47.7%	37.8%
②	S-decoder	New-born GTs	one-to-many	w/o parallel decoder	38.8%	1.357	46.5%	37.1%
③	S-decoder	All GTs	one-to-many	w/o parallel decoder	26.7%	1.470	38.5%	17.3%
④	C-decoder	New-born GTs	one-to-many	cross	44.7%	1.262	49.7%	39.6%
⑤	C-decoder	New-born GTs	one-to-many	self+cross	28.1%	1.458	39.9%	17.8%

Comparison to 2D detection methods. (1) As we claim that simultaneously addressing the over-duplication of object queries and the self-centric issues of track queries within a parallel decoder remains non-trivial in the main paper, we validate this by directly applying DAC-DETR [11]’s parallel decoder technology to the tracking-by-attention paradigm as shown in ③ of the Tab. A6. (2) Although DETA [12] and YOLOX [13] have designed one-to-many matching strategies (DETA and SimOTA) specifically for 2D detectors, adapting these technologies directly from 2D to 3D MOT yields limited improvements, as shown in ② and ③ of the Tab. A6. (3) The ablation results in Table 4(b) of our main paper incorporate both Hybrid Matching and Instance-aware Contrastive Learning. We combine

these results below (④ and ⑤) for a thorough comparison. In contrast to the incremental adaptation of these 2D techniques to 3D MOT (②vs.④, ③vs.⑤), the consistent improvements by SynCL highlight the novelty of our work and its contribution to 3D MOT.

Table A6: Comparison results among SynCL and other 2D detection methods.

#	Methods	Backbone	Detector	Tracking		Detection	
				AMOTA	AMOTP↓	NDS	mAP
①	PF-Track-S	V2-99	PETR	40.8%	1.343	47.7%	37.8%
②	DAC-DETR (SimOTA)	V2-99	PETR	40.7%	1.345	48.0%	37.8%
③	DAC-DETR (DETA)	V2-99	PETR	41.3%	1.337	48.7%	38.3%
④	SynCL (SimOTA)	V2-99	PETR	42.8%	1.323	49.2%	38.8%
⑤	SynCL (DETA)	V2-99	PETR	43.1%	1.320	49.4%	38.6%
⑥	SynCL (GMM)	V2-99	PETR	44.7%	1.262	49.7%	39.6%

Analysis of IDS degradation. We first summarize and analyze the changes in ID switches before and after integrating our SynCL as follows: (1) In Table 1 of our main paper, PF-Track and Baseline#1 show fewer ID switches compared to MUTR3D and Baseline#2 because the PF-Track framework includes a trajectory prediction module, which prolongs the lifespan of tracklets when occlusions or motion blur result in low-confidence predictions for track queries. (2) With this trajectory prediction module, if the detector performs better, integrating our SynCL can actually reduce ID switches (Baseline#1 vs. SynCL: IDS 173 vs. IDS 170) and significantly improve overall metrics, including AMOTA which offers a holistic evaluation of tracking performance by accounting for IDS, FP, FN, etc.. (3) Conversely, without this trajectory prediction module, even when using a stronger detector (MUTR3D vs. Baseline#2), integrating SynCL will still lead to an increase in ID switches.

We then summarize the possible reasons for this degradation of the IDS metric as follows: (1) Firstly, our Hybrid Matching module has uncovered more high-quality candidates, leading to an increase in the model’s total positive predictions, as shown in Tab. A7. From a probabilistic perspective, this has heightened the likelihood of ID switches occurring between tracklets. (2) Secondly, track queries may generate low-quality predictions during the inter-frame propagation due to factors, e.g., occlusion and motion blur. With the integration of SynCL, object queries can discover and produce high-quality predictions for the same target while suppressing the confidence of the corresponding track query predictions, creating new tracklets. Although this substitution may lead to ID switches, it allows for robust re-detection. After incorporating PF-Track’s trajectory prediction module, the model can retain track queries for a longer duration before re-detecting them, facilitating the identification of truly lost targets outside the field of view. Therefore, the combination of robust re-detection with the trajectory prediction module is essential for addressing the issue of ID switches.

Table A7: Comparison results among SynCL and other 2D detection methods.

Methods	Backbone	Detector	IDS↓	TP	FP↓	Total Positive Predictions
MUTR3D	R101	DETR3D	474	57595	15269	72864
SynCL	R101	DETR3D	588	60569	14311	74880
PF-Track-S	V2-99	PETR	166	60879	16331	77210
SynCL	V2-99	PETR	203	64893	15344	80237
Baseline#1	V2-99	PETRv2	173	64659	14106	78765
SynCL	V2-99	PETRv2	170	65923	13411	79334
Baseline#2	V2-99	Stream	411	66257	13962	80219
SynCL	V2-99	Stream	540	67689	13639	81328

B.4 Visualization Results

We provide visualization results comparing the original predictions of the tracker with the predictions after adopting our proposed SynCL approach. As shown in Fig.A2, trackers with SynCL successfully capture potential candidates previously overlooked by self-attention mechanisms and enhance the cross-frame localization of track queries.

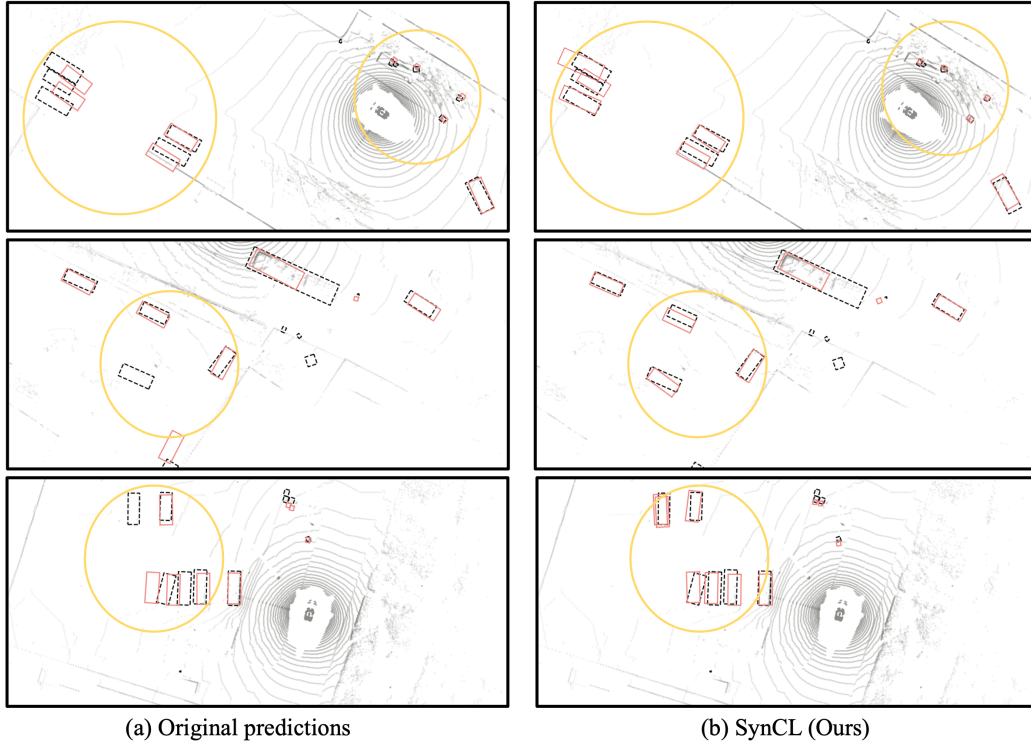


Figure A2: Visualization on the nuScenes *val* set. The black dashed boxes represent ground-truths, while the pink boxes denote predictions.

B.5 Test Split Screenshot

We include a screenshot of the submission to the nuScenes *test* set in Fig. A3, which serves as a supplement to the test set results presented in Tab. 3 of the main paper.

```

Calculating metrics...
Saving metrics to: /tmp/tmpwc58c9ul/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   0.495   1.061   0.580   0.789 2186   0.454   0.464  81  72   19.5 1258 266  919  9   4   0.50  0.78
bus        0.471   1.133   0.581   0.712 1701   0.413   0.626  38  40   19.1  986  284  712  3  10   1.27  1.85
car        0.716   0.724   0.705   0.822 68518   0.638   0.432 2816 1151 162.4 53241 9498 14758 519 378   0.57  0.89
motorcy    0.599   0.955   0.678   0.765 1945   0.510   0.499  74  43   23.5 1295 304  626  24  16   0.77  1.15
pedestr    0.672   0.938   0.715   0.818 34010   0.574   0.659 1104 524  87.7 23856 4348 9691 463 336   0.80  1.34
trailer    0.622   1.006   0.689   0.753 2566   0.514   0.666  74  41   43.9 1753 433  798  15  11   0.79  1.58
truck      0.538   1.015   0.668   0.645 8639   0.427   0.550  236 165   50.2 5713 2028 2868  58  49   0.98  1.50

Aggregated results:
AMOTA 0.588
AMOTP 0.976
RECALL 0.671
MOTAR 0.758
GT 17080
MOTA 0.504
MOTP 0.556
MT 4423
ML 2036
FAF 58.1
TP 88102
FP 17161
FN 30372
IDS 1091
FRAG 804
TID 0.81
LGD 1.30
Eval time: 5984.8s

Completed evaluation for test phase

```

Figure A3: Screenshot of the test set results, which serves as a supplement to our results in Tab. 3 of the main paper.

References

- [1] Tianyuan Zhang, Xuanyao Chen, Yue Wang, Yilun Wang, and Hang Zhao. MUTR3D: A multi-camera tracking framework via 3d-to-2d queries. In *CVPRW*, pages 4536–4545, 2022.
- [2] 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. In *CVPR*, pages 17928–17938, 2023.

- 136 [3] Yue Wang, Vitor Guizilini, Tianyuan Zhang, Yilun Wang, Hang Zhao, and Justin Solomon. DETR3D: 3d
137 object detection from multi-view images via 3d-to-2d queries. In *Conference on Robot Learning*, pages
138 180–191. PMLR, 2021.
- 139 [4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition.
140 In *CVPR*, pages 770–778, 2016.
- 141 [5] Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie.
142 Feature pyramid networks for object detection. In *CVPR*, pages 936–944, 2017.
- 143 [6] Yingfei Liu, Tiancai Wang, Xiangyu Zhang, and Jian Sun. PETR: position embedding transformation for
144 multi-view 3d object detection. In *ECCV*, pages 531–548, 2022.
- 145 [7] Youngwan Lee, Joong-Won Hwang, Sangrok Lee, Yuseok Bae, and Jongyoul Park. An energy and
146 gpu-computation efficient backbone network for real-time object detection. In *CVPRW*, pages 752–760,
147 2019.
- 148 [8] Yingfei Liu, Junjie Yan, Fan Jia, Shuailin Li, Aqi Gao, Tiancai Wang, and Xiangyu Zhang. Petrv2: A
149 unified framework for 3d perception from multi-camera images. In *ICCV*, pages 3239–3249, 2023.
- 150 [9] Shihao Wang, Yingfei Liu, Tiancai Wang, Ying Li, and Xiangyu Zhang. Exploring object-centric temporal
151 modeling for efficient multi-view 3d object detection. In *ICCV*, pages 3598–3608, 2023.
- 152 [10] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object
153 detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, pages 318–327, 2020.
- 154 [11] Zhengdong Hu, Yifan Sun, Jingdong Wang, and Yi Yang. DAC-DETR: divide the attention layers and
155 conquer. In *NeurIPS*, 2023.
- 156 [12] Jeffrey Zhang, Jang Hyun Cho, Xingyi Zhou, and Philipp Krähenbühl. NMS strikes back. *CoRR*,
157 abs/2212.06137, 2022.
- 158 [13] Zheng Ge, Songtao Liu, Feng Wang, Zeming Li, and Jian Sun. YOLOX: exceeding YOLO series in 2021.
159 *CoRR*, abs/2107.08430, 2021.