DRIVE4RT: LEARNING QUERYABLE 4D SCENE REPRESENTATIONS FOR AUTONOMOUS DRIVING

Anonymous authors
Paper under double-blind review

ABSTRACT

Understanding dynamic driving scenes requires more than reconstructing their instantaneous 3D geometry: a scene representation should also capture how the same physical points correspond across cameras and evolve over time. We present **Drive4RT**, a queryable 4D scene representation for surround-view autonomous driving. Given a short multi-camera video, Drive4RT encodes the observations into a shared 4D scene representation and predicts the metric 3D location of a physical point conditioned on its source observation, target time, and reference camera frame. By varying the query conditions, 3D reconstruction, cross-view geometric reasoning, and temporal point tracking become different readouts of the same underlying representation. Adapting queryable 4D reconstruction to driving scenes introduces two key challenges: heterogeneous surround-view observations and the coexistence of a largely static environment with independently moving traffic participants. Drive4RT addresses these challenges through structured intra-view, cross-view, and cross-frame reasoning, together with dynamic motion modeling that combines scene-level motion grouping and point-level residuals. To learn point-level 4D geometry without dense trajectory annotations, we further construct complementary supervision from standard driving logs: static correspondences are derived from LiDAR and ego poses, while dynamic trajectories are generated from temporally associated 3D object tracks. Across five large-scale autonomous-driving datasets, Drive4RT achieves strong performance in 3D reconstruction, depth estimation, ego-pose estimation and point-level 4D tracking. These results demonstrate that geometry and motion in dynamic driving scenes can be organized around a shared queryable 4D representation rather than modeled as isolated perception outputs.

1 INTRODUCTION

Recovering the three-dimensional structure of the surrounding environment is a fundamental problem in autonomous driving. Over the past few years, vision-centric driving perception has progressively moved from object-level detection toward increasingly dense scene representations, including bird's-eye-view features Li et al. (2024), 3D occupancy Wei et al. (2023); Tian et al. (2023), depth estimation, and dense 3D geometry. In parallel, general-purpose visual geometry models such as DUSt3R Wang et al. (2024), VGGT Wang et al. (2025b), and CUT3R Wang et al. (2025c) have demonstrated that large vision transformers can recover camera parameters, depth, and dense point maps directly from images while learning transferable geometric priors. More recently, driving-specific models such as DVGT Zuo et al. (2026) extend this paradigm to heterogeneous surround-view camera systems and directly reconstruct metric-scale scene geometry. Despite these advances, most existing geometric representations primarily answer a spatial question: *where is the observed scene content in 3D?*

Dynamic driving scenes require a richer representation. For a physical point observed on a moving vehicle, recovering only its current 3D position is insufficient. A useful scene representation should additionally describe how the same physical point corresponds across viewpoints, how it evolves over time, and where it lies when expressed in different coordinate frames. Existing approaches typically address these capabilities through specialized formulations. Point trackers such as TAPIR Doersch et al. (2023) and CoTracker Karaev et al. (2024) establish long-range 2D correspondences, while SpatialTracker Xiao et al. (2024) extends point tracking into 3D space. More recent meth-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

ods jointly reason about scene geometry and motion: ST4RTrack Feng et al. (2025) reconstructs dynamic geometry while tracking points in a consistent world frame, and SpatialTrackerV2 Xiao et al. (2025) jointly estimates scene geometry, camera motion, and pixel-wise 3D motion. These methods substantially advance dynamic scene understanding, but geometry, correspondence, and motion are still often exposed through task-specific outputs rather than through a unified interface to the underlying scene representation.

Recent work on queryable 4D reconstruction provides a promising alternative. D4RT Zhang et al. (2026) encodes a video into a shared latent scene representation and predicts the 3D location of a physical point conditioned on its source observation, target time, and reference coordinate frame. Related work such as 4RC Luo et al. (2026) further explores conditional 4D reconstruction across space and time, while Trace Anything Liu et al. (2026) and SM4RT Lin et al. (2026) investigate continuous or structured representations of scene motion. These approaches suggest a different perspective on dynamic geometry: rather than exhaustively predicting a fixed collection of geometric outputs, a scene can be represented as a latent state that is queried according to the geometric relation of interest. Under this formulation, reconstruction and tracking become different readouts of the same underlying dynamic scene representation.

However, extending such queryable 4D representations to autonomous driving is not a straightforward domain transfer. First, autonomous vehicles observe the environment through *heterogeneous surround-view camera rigs*, where cameras differ substantially in viewpoint, intrinsics, extrinsics, and visual overlap. Existing driving representations have shown that structured cross-view and temporal aggregation is crucial for multi-camera perception Li et al. (2024); Wei et al. (2023); Zuo et al. (2026), but queryable 4D models are largely designed for monocular or relatively homogeneous image sequences Xiao et al. (2025); Feng et al. (2025); Zhang et al. (2026); Luo et al. (2026). A driving-oriented 4D representation must therefore reason jointly within each view, across cameras, and across time.

Second, driving scenes exhibit a strong structural decomposition between the largely static environment and a relatively small number of independently moving traffic participants. Ego motion and object motion jointly induce complex temporal variation, making unconstrained point-wise motion prediction unnecessarily difficult. Existing dynamic reconstruction and tracking approaches model motion explicitly Feng et al. (2025); Xiao et al. (2025); Liu et al. (2026); Lin et al. (2026), yet autonomous-driving scenes provide an additional inductive bias: most points belong to the static world, while foreground motion is concentrated in a small set of traffic actors. Exploiting this structure can simplify dynamic modeling and improve temporal consistency.

Third, standard autonomous-driving datasets do not provide dense point-level 4D correspondences. LiDAR measurements and accurate ego poses provide strong geometric supervision for static regions, but naively transforming observations from different timestamps into a shared world frame implicitly assumes that all scene points are static. This creates incorrect zero-motion constraints for independently moving objects. Although driving datasets commonly provide tracked 3D bounding boxes, such annotations have rarely been exploited to construct dense point-level temporal supervision for queryable 4D representation learning. Learning from ordinary driving logs therefore requires a supervision strategy that treats static and dynamic scene content differently.

To address these challenges, we develop Drive4RT, which makes queryable 4D representation learning practical for heterogeneous surround-view dynamic driving scenes. Rather than treating reconstruction, cross-view correspondence, and temporal motion as separate outputs, Drive4RT organizes them around a shared point-query interface over a unified 4D scene representation. Given a short multi-camera video, Drive4RT encodes the observations into a shared 4D scene representation and exposes scene geometry through a common point-query interface. Each query specifies a source image location, a target time, and a reference camera frame, and the model predicts the metric 3D position of the corresponding physical point. By changing the target time or reference frame while keeping the source observation fixed, the same latent representation supports instantaneous 3D reconstruction, cross-view geometric querying, and temporal point tracking. The source observation determines the identity of the physical point, while temporal and camera conditions determine how this point is read out from the shared representation.

To model heterogeneous surround-view videos efficiently, Drive4RT factorizes information exchange along the three structural axes of driving observations: within-view geometry, cross-view

spatial relations, and temporal evolution. The resulting representation is shared across all point queries. For dynamic scene content, Drive4RT further combines scene-level motion grouping with point-level residual motion, explicitly separating the dominant static background from independently moving traffic participants. This provides a structured motion prior suited to driving scenes rather than predicting unconstrained motion independently for every point.

More importantly, we introduce complementary supervision that enables point-level 4D representation learning directly from standard autonomous-driving logs. For static regions, cross-time correspondences are constructed by lifting LiDAR observations into 3D and transforming them into a common scene coordinate frame using ego poses. For dynamic traffic participants, LiDAR points are associated with temporally tracked 3D bounding boxes and propagated according to object motion. This static–dynamic separation prevents moving objects from being incorrectly constrained by static-world transformations and provides temporally coherent point-level supervision without requiring manually annotated dense trajectories.

We evaluate Drive4RT across five large-scale autonomous-driving datasets with diverse camera configurations and driving environments. The learned representation achieves strong performance in 3D reconstruction, depth estimation, and ego-pose estimation, while substantially improving point-level 3D tracking over the original D4RT formulation. Qualitative results further demonstrate temporally coherent point trajectories for independently moving traffic participants. These results suggest that geometry, correspondence, and motion in dynamic driving scenes can be organized around a shared queryable representation rather than treated as isolated perception outputs.

Our main contributions are summarized as follows:

- We extend **queryable 4D representation learning to heterogeneous surround-view dynamic driving scenes**, enabling reconstruction, cross-view geometric reasoning, and temporal point tracking to be accessed through a shared conditional point-query interface.
- We introduce a **driving-oriented architecture** that adapts pretrained visual geometry representations to heterogeneous multi-camera videos, combining structured spatiotemporal reasoning with source-anchored querying and structured dynamic motion modeling.
- We develop complementary **static–dynamic supervision** from standard driving logs, using LiDAR and ego poses for static geometry and tracked 3D boxes for dynamic point trajectories, enabling point-level 4D learning without dense trajectory annotations.

2 METHOD

As illustrated in Fig. 1, Drive4RT represents a surround-view driving video as a shared, queryable 4D scene representation. Rather than predicting reconstruction, correspondence, and motion with separate task-specific heads, Drive4RT exposes these geometric relations through a common conditional point-query interface. Given a physical point observed at a source image location, the model predicts its metric 3D position under a specified target time and reference camera frame.

2.1 QUERYABLE 4D SCENE REPRESENTATION

Given a surround-view clip $\mathcal{I}$ with T timestamps and V cameras, Drive4RT answers a point query:

$$\hat{\mathbf{x}}_{\text{cam}} = F_\theta(\mathcal{I}, q), \qquad q = (\mathbf{p}, t_s, c_s, t, c_r). \tag{1}$$

The source pixel $\mathbf{p} = (u, v)$, time t_s, and camera c_s identify the physical point. The target time t and reference camera c_r specify when and in which camera frame its metric 3D position is returned. Setting $t = t_s$ gives reconstruction; varying t gives tracking. All target times lie within the observed clip.

Internally, points move in a fixed scene frame, chosen as the ego frame at the first timestamp. Their positions are converted to the requested camera frame using estimated ego poses and rig extrinsics. Thus, a static point has zero scene-frame motion even when the camera moves. Changing the reference camera changes the coordinates and possibly visibility, but not point identity.

2.2 Structured Surround-View Encoding

A surround-view driving video has a highly structured organization: each observation contains local image geometry, multiple cameras jointly describe the same spatial environment, and consecutive frames capture temporal evolution. Applying unrestricted global attention over all image tokens ignores this structure and scales poorly with the number of cameras and frames. We therefore factorize scene encoding along these three axes. We organize feature interaction into:

- **Intra-view interaction**, which captures local appearance and geometric structure within each individual image;
- **Cross-view interaction**, which exchanges information among cameras observed at the same timestamp;
- **Cross-frame interaction**, which propagates temporal information along observations from the same camera stream.

These interactions are applied sequentially, yielding a shared scene representation

$$\mathbf{F} = E_\theta(\mathcal{I}),$$

which is subsequently accessed by all point queries. The structural design preserves the geometric prior of the pretrained backbone while explicitly adapting information flow to the spatial and temporal organization of surround-view driving data.

2.3 Source-Anchored Point Querying

Given the shared scene representation $\mathbf{F}$, Drive4RT uses a lightweight query decoder to retrieve the representation of a specific physical point. A query embeds the source image location together with its source camera and time, target time, and target reference frame. The query interacts with the scene representation through cross-attention,

$$\mathbf{f}_q = D_\theta(\mathbf{F}, q),$$

where $\mathbf{f}_q$ denotes the query-specific point representation.

Importantly, point queries are decoded independently: different queries interact with the shared scene representation but do not attend to one another. The prediction therefore does not depend on the number, ordering, or batching of other queried points. This allows the same representation and decoder to support both sparse point tracking and dense geometric reconstruction.

We adopt a *source-anchored* formulation to preserve point identity across different queries. The source pixel $(\mathbf{p}, t_s, c_s)$ first determines a canonical representation of the physical point. Temporal and reference-frame conditions then modulate how this point is read out, without redefining its

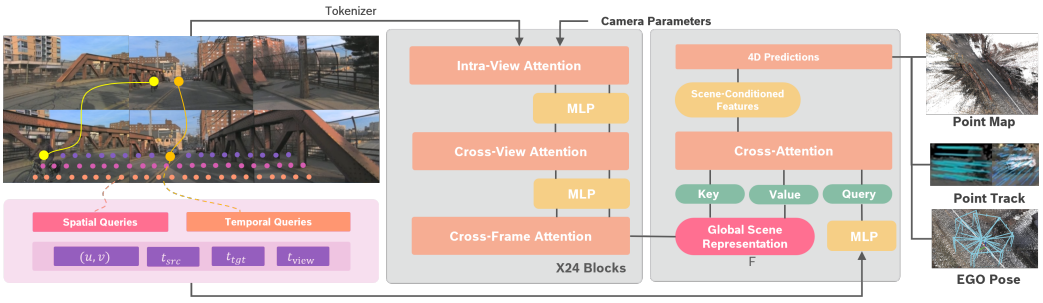


Figure 1: Framework of our Drive4RT for a queryable 4D scene representation for multi-camera driving videos. The framework contains three main components. First, a structured surround-view encoder aggregates information within images, across cameras, and across time. Second, a source-anchored query decoder retrieves the representation of a physical point from the shared scene state. Third, a structured motion module models temporal evolution by combining scene-level motion grouping with point-level residual motion.

identity. Consequently, the same source point can be queried at multiple timestamps or expressed under different camera coordinate systems while sharing a common underlying point representation.

All 3D coordinates are internally represented in a clip-level ego-centric canonical frame defined by the first context frame. Calibrated camera extrinsics are then used to transform queried geometry into the requested reference coordinate system.

2.4 Structured Dynamic Motion Modeling

Driving scenes are dominated by static background, while independent motion is concentrated in a small number of traffic participants. We therefore decompose point motion into shared scene-level motion and point-level residuals.

Given the query feature f_q, Drive4RT predicts one background slot and K foreground motion slots, together with a soft assignment

$$\boldsymbol{\alpha}_q = \mathrm{softmax}(g(f_q)). \tag{2}$$

Each foreground slot s_k predicts a time-conditioned motion prototype $m_k(t) = M(s_k, t)$, yielding

$$\Delta\mathbf{X}_{\mathrm{slot}}(q,t) = \sum_{k=1}^{K} \alpha_{q,k} \left[m_k(t) - m_k(t_s) \right]. \tag{3}$$

A query-specific residual captures local deviations,

$$\Delta\mathbf{X}_{\mathrm{res}}(q,t) = (1 - \alpha_{q,0}) \left[r(f_q, t) - r(f_q, t_s) \right]. \tag{4}$$

The final trajectory is

$$\hat{\mathbf{X}}_q(t) = \hat{\mathbf{X}}_q(t_s) + \beta_s \Delta\mathbf{X}_{\mathrm{slot}} + \beta_r \Delta\mathbf{X}_{\mathrm{res}}. \tag{5}$$

Thus, motion slots capture shared actor-level dynamics, while the residual branch preserves point-level flexibility and source-time consistency.

2.5 Learning 4D Supervision from Driving Logs

Dense point-level 4D trajectories are rarely available in standard autonomous-driving datasets. However, driving logs typically provide sparse LiDAR measurements, accurate ego poses, and temporally associated 3D object tracks. Drive4RT converts these annotations into complementary static and dynamic point supervision.

Static scene supervision. For a LiDAR point observed at time t, we transform its 3D coordinate into a clip-level canonical frame using the ground-truth ego pose. For static scene content, observations from different timestamps should map to the same physical location in this canonical frame. This provides cross-time geometric correspondences for roads, buildings, vegetation, and other static structures.

Dynamic object supervision. The same transformation is invalid for independently moving traffic participants, as it would incorrectly assign them zero motion in the world frame. We therefore identify LiDAR points belonging to temporally tracked 3D bounding boxes and propagate these points according to the rigid transformation of the corresponding object track. This produces temporally aligned point trajectories for moving actors without requiring manually annotated dense point tracks.

The static and dynamic supervision paths are complementary: ego motion explains the transformation of the static world, while tracked object motion explains independently moving scene content. Their separation prevents dynamic points from receiving contradictory static-world constraints.

Geometric objective. For each query, Drive4RT predicts metric 3D coordinates together with auxiliary geometric quantities including target-view projection, visibility, displacement, surface normal, and confidence. The geometric objective is written as

$$\mathcal{L}_{\mathrm{geo}} = \lambda_{3D}\mathcal{L}_{3D} + \lambda_{\mathrm{proj}}\mathcal{L}_{\mathrm{proj}} + \lambda_{\mathrm{vis}}\mathcal{L}_{\mathrm{vis}} + \lambda_{\mathrm{disp}}\mathcal{L}_{\mathrm{disp}} + \lambda_{\mathrm{normal}}\mathcal{L}_{\mathrm{normal}} + \lambda_{\mathrm{conf}}\mathcal{L}_{\mathrm{conf}}.$$

Following robust geometric regression practice, coordinates are mapped using

$$\phi(x) = \operatorname{sign}(x) \log(1 + |x|),$$

before geometric regression. We use ℓ_1 losses for 3D position, projection, and displacement, binary cross-entropy for visibility, and cosine distance for surface normals. Each loss is applied only where valid supervision is available.

Trajectory objective. The trajectory objective supervises metric point motion and the structured motion representation:

$$\mathcal{L}_{\text{traj}} = \mathcal{L}_{\text{pos}} + \lambda_g \mathcal{L}_{\text{group}} + \lambda_e \mathcal{L}_{\text{ego}} + \lambda_c \mathcal{L}_{\text{cons}} + \lambda_b \mathcal{L}_{\text{bal}} + \lambda_r \mathcal{L}_{\text{res}}. \tag{6}$$

Metric source and target positions are supervised using a Huber loss with $\delta = 1\,\text{m}$. The grouping loss assigns static points to the background slot and encourages points belonging to the same tracked object to share similar foreground-slot assignments. Since foreground-slot identities are defined only within each clip, the objective is permutation-invariant with respect to slot indices. Temporal consistency encourages the same tracked actor to maintain similar slot assignments across source frames, while the balancing term prevents foreground points from collapsing into a single motion slot. The residual regularization discourages the point-level residual branch from explaining all motion independently, encouraging shared actor-level motion to be captured by the foreground slots. Ego motion is supervised using rotation and translation errors over the observed frames.

The complete training objective is

$$\mathcal{L} = \mathcal{L}_{\text{geo}} + \mathcal{L}_{\text{traj}}.$$

3 EXPERIMENTS

Our experiments aim to examine whether a single queryable 4D representation can jointly support geometric reconstruction and temporal point reasoning in surround-view driving scenes. To this end, we evaluate Drive4RT across complementary tasks covering 3D reconstruction, depth estimation, ego-pose estimation, and point-level 3D tracking on diverse autonomous-driving datasets. We further use ablation and qualitative studies to analyze how structured spatiotemporal modeling and static-dynamic supervision contribute to the learned representation.

3.1 Training Setup

We train Drive4RT on five large-scale autonomous-driving datasets: **NuScenes** Caesar et al. (2020), **Waymo Open Dataset** Sun et al. (2020), **OpenScene** Contributors (2023), **DDAD** Guizilini et al. (2020), and **Argoverse 2** Wilson et al. (2023), covering diverse camera rigs, driving environments, and traffic conditions. Drive4RT jointly learns scene geometry and point-level motion using the supervision introduced in Sec. 2.5. Static regions are supervised by LiDAR measurements and ego poses, while dynamic regions are supervised using temporally associated 3D object tracks. Detailed training configurations are provided in the Appendix.

3.2 3D Reconstruction and Depth Estimation

We first evaluate the geometric representation learned by Drive4RT on NuScenes, Waymo, OpenScene, Argoverse V2, and DDAD. We report point-cloud accuracy (Acc) and completeness (Comp) for 3D reconstruction, together with Abs Rel and $\delta < 1.25$ for ray-depth estimation. As shown in Table 1, Drive4RT achieves strong reconstruction performance across heterogeneous camera configurations. It obtains the best Acc and Comp on Waymo, and the best Comp on both OpenScene and DDAD. The improvement is particularly pronounced on Waymo, where Drive4RT reduces Acc from 0.457 to 0.225 and Comp from 0.494 to 0.356 compared with DVGT. These results indicate that structured cross-view and temporal reasoning improves the global consistency and completeness of the reconstructed geometry.

Table 2 reports ray-depth estimation results. Drive4RT achieves the best Abs Rel on Waymo and the best overall depth performance on DDAD, while DVGT remains stronger on NuScenes and OpenScene. Notably, the gain in global 3D reconstruction is more consistent than that in per-ray depth estimation, suggesting that Drive4RT primarily benefits the cross-view and cross-frame organization of scene geometry rather than optimizing individual depth predictions in isolation.

Table 1: **Quantitative 3D reconstruction results across diverse datasets.** * indicates that the predicted 3D point map is aligned with the sparse LiDAR depth using the Umeyama alignment Umeyama (1991) algorithm to recover the metric scale.

	NuScenes		Waymo		OpenScene		DDAD		Argoverse 2	
Method	Acc ↓	Comp ↓	Acc ↓	Comp ↓	Acc ↓	Comp ↓	Acc ↓	Comp ↓	Acc ↓	Comp ↓
CUT3R*	2.054	2.603	3.391	4.216	1.864	2.258	2.774	4.677	–	–
VGGT*	1.300	1.498	1.641	2.053	1.422	1.496	1.741	2.473	–	–
StreamVGGT*	2.588	2.414	3.630	3.275	2.304	2.098	2.717	2.788	–	–
DVGT	**0.457**	**0.494**	0.457	0.494	1.714	2.216	**0.402**	0.481	–	–
Drive4RT	0.596	0.749	**0.225**	**0.356**	**0.984**	**1.196**	0.700	**0.390**	**0.700**	**0.389**

Table 2: **Quantitative ray depth results across diverse datasets.**

	NuScenes		Waymo		OpenScene		DDAD		Argoverse 2	
Method	AbsRel ↓	δ_1 ↑	AbsRel ↓	δ_1 ↑	AbsRel ↓	δ_1 ↑	AbsRel ↓	δ_1 ↑	AbsRel ↓	δ_1 ↑
CUT3R	0.332	0.547	0.291	0.562	0.278	0.593	0.870	0.315	–	–
VGGT	0.243	0.729	0.176	0.811	0.241	0.719	0.613	0.476	–	–
MapAnything	0.568	0.269	0.507	0.211	0.486	0.240	1.971	0.195	–	–
StreamVGGT	0.412	0.540	0.339	0.584	0.319	0.607	0.838	0.415	–	–
Driv3R	0.189	0.721	0.168	0.770	0.188	0.740	0.185	0.740	–	–
DVGT	**0.069**	**0.953**	0.106	**0.921**	**0.049**	**0.971**	0.152	0.837	–	–
Drive4RT	0.103	0.894	**0.085**	0.916	0.306	0.701	**0.149**	**0.846**	**0.091**	**0.917**

3.3 3D Point Tracking

Beyond static geometry, we evaluate whether the learned representation can preserve point identity and motion over time. Following the TAPVid-3D-style evaluation protocol, we report average 3D position accuracy (APD_{3D}), occlusion accuracy (OA), 3D average Jaccard (AJ_{3D}), and Recall. As shown in Table 3, we compare Drive4RT with both the original D4RT and SpatialTrackerV2, a dedicated 3D point tracking method, under the same evaluation protocol. Drive4RT consistently outperforms both baselines across all evaluated driving datasets and metrics. Compared with D4RT, the improvements are particularly pronounced in APD_{3D} and Recall, demonstrating the importance of adapting queryable 4D representations to heterogeneous surround-view driving scenes. Drive4RT also substantially improves over SpatialTrackerV2, especially in APD_{3D} and Recall, indicating stronger recovery of temporally consistent 3D point correspondences in dynamic driving scenes.

Importantly, point tracking is obtained from the same shared scene representation used for geometric reconstruction, without maintaining a separate task-specific tracking model. These results provide evidence that Drive4RT jointly captures scene geometry and point-level temporal evolution within a unified queryable 4D representation.

Table 3: **Quantitative evaluation of 3D point tracking capability.** All methods are evaluated using the same point queries and TAPVid-3D-style evaluation protocol.

	D4RT*				SpatialTrackerV2				Drive4RT			
Dataset	APD_{3D} ↑	OA ↑	AJ_{3D} ↑	Recall↑	APD_{3D} ↑	OA ↑	AJ_{3D} ↑	Recall↑	APD_{3D} ↑	OA ↑	AJ_{3D} ↑	Recall↑
NuScenes	0.050	0.377	0.023	0.005	0.031	0.579	0.041	0.002	0.280	0.836	0.230	0.322
Waymo	0.028	0.326	0.015	0.019	0.022	0.639	0.022	0.000	0.310	0.783	0.125	0.344
OpenScene	0.032	0.385	0.008	0.051	0.018	0.673	0.037	0.010	0.134	0.769	0.106	0.409
Argoverse 2	0.040	0.361	0.018	0.024	0.021	0.413	0.049	0.008	0.221	0.742	0.247	0.362

3.4 Ego-Pose Estimation

We additionally evaluate whether the learned representation captures global camera motion using AUC@30, following the relative-pose evaluation protocol of DVGT. As shown in Table 4, Drive4RT achieves the best performance across all four datasets. It reaches 92.9 on NuScenes and 92.6 on

Table 4: **Quantitative ego pose results across diverse datasets.**

Method	NuScenes AUC@30 ↑	Waymo AUC@30 ↑	OpenScene AUC@30 ↑	DDAD AUC@30 ↑	AV2 AUC@30 ↑
CUT3R	43.5	50.1	34.7	48.6	-
VGGT	87.8	87.7	66.3	92.8	-
MapAnything	85.0	82.8	65.6	87.0	-
StreamVGGT	86.2	85.6	74.1	91.9	-
DVGT	86.5	86.4	74.7	95.1	-
Drive4RT	**92.9**	**92.6**	**79.8**	**95.2**	**91.5**

Table 5: **Cross-view point querying on Waymo.** Queries are constructed from LiDAR points visible in two overlapping cameras.

Method	3D Err. (m) ↓	Reproj. Err. (px) ↓
D4RT	0.438	25.4
Drive4RT w/o CV	0.407	18.6
Drive4RT	**0.113**	**5.4**

Waymo, improving over VGGT by 5.1 and 4.9 points, respectively. On OpenScene, Drive4RT improves the previous best result from 74.7 to 79.8, while achieving 95.2 on DDAD. These results complement the reconstruction and tracking experiments: the shared representation captures not only local point geometry and dynamic motion, but also the global ego-motion structure of the scene.

Cross-view point querying. We directly evaluate whether a source-anchored query preserves point identity across cameras. For each LiDAR point visible in two overlapping cameras at the same timestamp, we use its projection in one camera as the source query and request its 3D coordinate in the other camera frame. We evaluate the predicted target-frame 3D position and its reprojection into the target image. This setting fixes the target time to the source time, thereby isolating cross-view reasoning from temporal tracking.

3.5 Zero-shot Generalization on KITTI.

To further evaluate cross-dataset generalization, we directly evaluate Drive4RT on KITTI Geiger et al. (2013) without any fine-tuning. As shown in Table 6, Drive4RT achieves strong zero-shot performance across geometry, depth, and ego-pose estimation, indicating that the learned 4D representation transfers well to unseen driving environments and camera configurations. Despite using no KITTI fine-tuning, Drive4RT achieves better 3D reconstruction performance than the KITTI-trained results officially reported by DVGT.

Table 6: **Additional evaluation on KITTI*(Zero-Shot).** KITTI results are obtained in a zero-shot setting without any fine-tuning.

Model	Acc	Comp	Abs Rel	$\delta < 1.25$	AUC@30	APD_{3D}	OA	AJ_{3D}	Recall
DVGT	0.846	1.468	0.136	0.849	87.6	-	-	-	-
Drive4RT*	0.720	0.440	0.088	0.958	87.7	0.281	0.773	0.103	0.494

3.6 Ablation Study

Ablation on structured spatiotemporal modeling. Table 7 shows complementary effects of cross-view (CV) and cross-frame (CF) reasoning: CV primarily improves reconstruction and depth, whereas CF benefits ego-pose estimation. Their combination, together with TPE, further improves temporal point tracking. Motion slots (MS) substantially enhance dynamic tracking, increasing APD_{3D} from 0.183 to 0.293 and AJ_{3D} from 0.094 to 0.145. Trajectory supervision (TS) further improves most metrics, with a small decrease in AJ_{3D} from 0.145 to 0.125.

3.7 Qualitative Analysis

Figure 2 shows representative 4D reconstruction results in diverse urban scenes. Drive4RT jointly recovers coherent scene geometry and point-level motion from short surround-view sequences, preserving static structures while capturing the temporal evolution of dynamic traffic participants. The results demonstrate consistent geometry and motion estimation under viewpoint changes, occlusions, and multiple moving objects.

Table 7: **Ablation study on structured spatiotemporal modeling on Waymo. CV and CF denote cross-view and cross-frame interactions.**

Ablation Setting	Acc	Comp	Abs Rel	$\delta < 1.25$	AUC@30	APD_{3D}	OA	AJ_{3D}	Recall
GA	2.963	4.378	0.544	0.382	68.0	0.040	0.334	0.016	0.094
CF-Only	0.305	0.759	0.215	0.849	89.4	0.087	0.378	0.027	0.102
CV-Only	0.590	0.550	0.120	0.904	71.6	0.103	0.405	0.045	0.150
+ CF	0.430	0.458	0.109	0.905	86.3	0.163	0.408	0.078	0.245
+ TPE	0.312	0.408	0.099	0.902	92.2	0.183	0.413	0.094	0.293
+ MS	0.301	0.396	0.096	0.907	92.1	0.293	0.541	0.145	0.314
+ TS	**0.225**	**0.356**	**0.085**	**0.916**	**92.6**	**0.310**	**0.783**	**0.125**	**0.344**

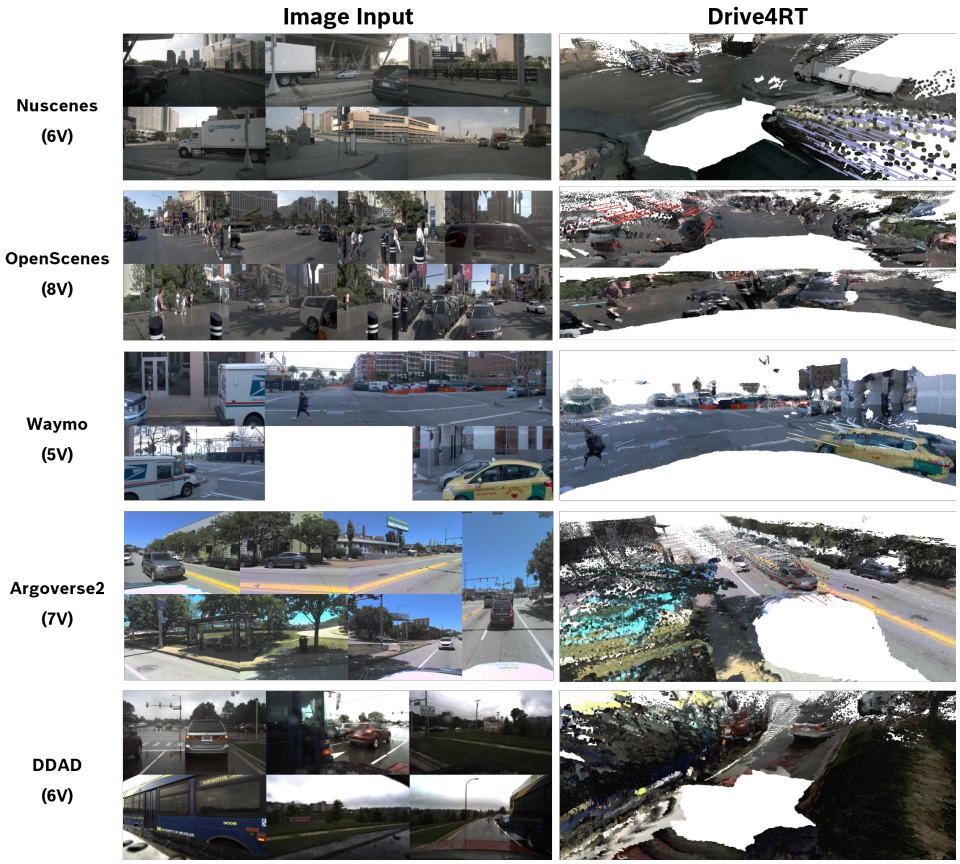


Figure 2: **Visualization of our Drive4RT on 4D point map reconstruction**.

4 Conclusion

We introduced **Drive4RT**, a queryable 4D scene representation for surround-view autonomous driving. By formulating 3D reconstruction, cross-view geometric reasoning, and temporal point tracking as conditional queries over a shared scene representation, Drive4RT provides a unified interface for modeling scene geometry and motion. Structured surround-view encoding and complementary static–dynamic supervision further enable learning from standard driving logs without dense 4D trajectory annotations. Experiments across five datasets demonstrate strong performance in reconstruction, depth estimation, ego-pose estimation, and point-level 3D tracking. Overall, our results suggest that dynamic driving geometry can be represented as a shared queryable 4D scene state rather than a collection of task-specific outputs.

AI USE STATEMENT

In this work, we used generative AI tools to assist with translation, refinement of methodological and mathematical descriptions, feedback on experimental design, and interpretation of results. We did not use generative AI tools to generate synthetic datasets, implement the reported methods, or produce experimental measurements. Additionally, generative AI tools were used to help draft and edit portions of the manuscript for clarity and readability.

All AI-assisted content was reviewed and verified by the authors against the actual implementation and experimental results. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide the main architectural design, supervision construction, training objectives, and evaluation protocols in the main paper. Detailed data preprocessing, clip construction, point-query sampling, augmentation, optimization settings, and metric definitions are provided in the Appendix. These details are intended to support reproduction of both the training pipeline and the reported evaluations.

REFERENCES

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 *2020 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 11618–11628. IEEE, 2020.

OpenScene Contributors. Openscene: The largest up-to-date 3d occupancy prediction benchmark in autonomous driving. `https://github.com/OpenDriveLab/OpenScene`, 2023.

Carl Doersch, Yi Yang, Mel Vecerik, Dilara Gokay, Ankush Gupta, Yusuf Aytar, Joao Carreira, and Andrew Zisserman. Tapir: Tracking any point with per-frame initialization and temporal refinement. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 10027–10038. IEEE, 2023.

David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. *Advances in neural information processing systems*, 27, 2014.

Haiwen Feng, Junyi Zhang, Qianqian Wang, Yufei Ye, Pengcheng Yu, Michael J Black, Trevor Darrell, and Angjoo Kanazawa. St4rtrack: Simultaneous 4d reconstruction and tracking in the world. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 8503–8513. IEEE, 2025.

Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The international journal of robotics research*, 32(11):1231–1237, 2013.

Vitor Guizilini, Rares Ambrus, Sudeep Pillai, Allan Raventos, and Adrien Gaidon. 3d packing for self-supervised monocular depth estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2485–2494, 2020.

Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker: It is better to track together. In *European conference on computer vision*, pp. 18–35. Springer, 2024.

Skanda Koppula, Ignacio Rocco, Yi Yang, Joe Heyward, João Carreira, Andrew Zisserman, Gabriel Brostow, and Carl Doersch. Tapvid-3d: A benchmark for tracking any point in 3d, 2024. URL `https://arxiv.org/abs/2407.05921`.

Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Qiao Yu, and Jifeng Dai. Bevformer: learning bird's-eye-view representation from lidar-camera via spatiotemporal transformers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(3):2020–2036, 2024.

Shing Ho J Lin, Wenzhao Zheng, Dong Zhuo, Yuqi Wu, Jie Zhou, and Jiwen Lu. Sm4rt: Learning structured motion geometry for 4d reconstruction. *arXiv preprint arXiv:2607.22534*, 2026.

Xinhang Liu, Yuxi Xiao, Donny Y Chen, Jiashi Feng, Yu-Wing Tai, Chi-Keung Tang, and Bingyi Kang. Trace anything: Representing any video in 4d via trajectory fields. In *International Conference on Learning Representations*, volume 2026, pp. 37738–37767, 2026.

Yihang Luo, Shangchen Zhou, Yushi Lan, Xingang Pan, and Chen Change Loy. 4rc: 4d reconstruction via conditional querying anytime and anywhere. *arXiv preprint arXiv:2602.10094*, 2026.

Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2443–2451. IEEE, 2020.

Xiaoyu Tian, Tao Jiang, Longfei Yun, Yucheng Mao, Huitong Yang, Yue Wang, Yilun Wang, and Hang Zhao. Occ3d: A large-scale 3d occupancy prediction benchmark for autonomous driving. *Advances in neural information processing systems*, 36:64318–64330, 2023.

Shinji Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on pattern analysis and machine intelligence*, 13(4):376–380, 1991.

Chengyu Wang, Isabella Bennett, Henry Scott, Liang Zhang, Mei Chen, Hao Li, and Rui Zhao. Matdecompsdf: High-fidelity 3d shape and pbr material decomposition from multi-view images, 2025a. URL https://arxiv.org/abs/2507.04749.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5294–5306. IEEE, 2025b.

Qianqian Wang, Yifei Zhang, Aleksander Holynski, Alexei A Efros, and Angjoo Kanazawa. Continuous 3d perception model with persistent state. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10510–10522. IEEE, 2025c.

Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20697–20709. IEEE, 2024.

Yi Wei, Linqing Zhao, Wenzhao Zheng, Zheng Zhu, Jie Zhou, and Jiwen Lu. Surroundocc: Multi-camera 3d occupancy prediction for autonomous driving. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 21672–21683. IEEE, 2023.

Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal, Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemodel Pontes, et al. Argoverse 2: Next generation datasets for self-driving perception and forecasting. *arXiv preprint arXiv:2301.00493*, 2023.

Yuxi Xiao, Qianqian Wang, Shangzhan Zhang, Nan Xue, Sida Peng, Yujun Shen, and Xiaowei Zhou. Spatialtracker: Tracking any 2d pixels in 3d space. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20406–20417. IEEE, 2024.

Yuxi Xiao, Jianyuan Wang, Nan Xue, Nikita Karaev, Yuri Makarov, Bingyi Kang, Xing Zhu, Hujun Bao, Yujun Shen, and Xiaowei Zhou. Spatialtrackerv2: 3d point tracking made easy. *arXiv preprint arXiv:2507.12462*, 2025.

Chuhan Zhang, Guillaume Le Moing, Skanda Koppula, Ignacio Rocco, Liliane Momeni, Junyu Xie, Shuyang Sun, Rahul Sukthankar, Joëlle K Barral, Raia Hadsell, et al. Efficiently reconstructing dynamic scenes one d4rt at a time. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7382–7392, 2026.

Sicheng Zuo, Zixun Xie, Wenzhao Zheng, Shaoqing Xu, Fang Li, Shengyin Jiang, Long Chen, Zhi-Xin Yang, and Jiwen Lu. Dvgt: Driving visual geometry transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14658–14668, 2026.

Table 8: **Detailed statistics of the datasets used in our experiments.** All temporal statistics are reported at a 2 Hz sampling rate.

Dataset	Train Scenes	Test Scenes	Num. Views
nuScenes	700	150	6
OpenScene	19,376	2,026	8
Waymo	798	202	5
DDAD	150	50	6
Argoverse V2	100	100	7
KITTI(test only)	–	13	2

A Appendix-Training and Implementation Details

A.1 Training Data and Clip Construction

We train Drive4RT on five large-scale autonomous-driving datasets: **Waymo Open Dataset**, **nuScenes**, **OpenScene**, **Argoverse 2**, and **DDAD**. We follow the same train/test splits in Table. 8. Each training sample contains $T = 6$ surround-view frames from all available cameras, corresponding to 5, 6, 8, 7, and 6 cameras, respectively. The base temporal sampling rate is 2 Hz, and all images are resized to 256×384.

Geometric supervision is constructed from LiDAR observations. Static correspondences are generated using ground-truth ego poses, while projected 3D bounding boxes exclude dynamic regions from static supervision. For tracked actors, LiDAR points are represented in object-local coordinates and propagated across time using tracked box poses. Precomputed sky masks further remove invalid regions.

A.2 Data Augmentation

We apply clip-consistent augmentation, sharing the same parameters across all cameras and timestamps to preserve geometric consistency. The temporal stride is randomly sampled from $\{1, 2\}$, corresponding to effective rates of approximately 2 Hz and 1 Hz. Brightness, contrast, and saturation are sampled from $[0.8, 1.2]$, with hue shifted within $[-0.05, 0.05]$. Grayscale conversion and Gaussian blur are applied with probabilities 0.05 and 0.1, respectively. No spatial crop or horizontal flip is used, and all augmentation is disabled during validation.

A.3 Point Query Sampling

We sample up to $Q = 4096$ point queries per clip. When dynamic LiDAR points are available, 25% of the queries are sampled from tracked actors and the remaining 75% from static regions; otherwise, all queries are static. Static queries are sampled from valid LiDAR pixels after masking dynamic objects and sky regions, with 20% preferentially sampled near depth discontinuities.

Source and target timestamps are sampled within the input clip. The target camera matches the source camera with probability 0.85 and is otherwise sampled from cameras with overlapping frustums. For dynamic queries, the same object-local LiDAR point is transformed across time using tracked 3D box poses to supervise 3D position, displacement, and trajectory.

A.4 Optimization

The visual encoder is initialized from the pretrained VGGT-1B checkpoint and fine-tuned jointly with the remaining Drive4RT modules. We use AdamW with weight decay 0.03 and global gradient clipping at 10. The learning rate is linearly warmed up for 1k iterations to 2.4×10^{-5} and then cosine-decayed to 4×10^{-7}.

Training uses FP16 mixed precision and distributed data parallelism on 48 NVIDIA A100 GPUs, with one clip per GPU, yielding a global batch size of 48 clips and approximately 196k point queries per step. The random seed is fixed to 42, and the learning-rate schedule is configured for up to 300k optimization steps.

B MORE EXPERIMENTS

High-Resolution Decoding We further visualize Drive4RT under dense querying (from 0.25x(64x96) to 4x(1024x1536)) in Fig. 3. By densely querying image locations, Drive4RT produces spatially coherent dense geometry while preserving fine structures and sharp depth discontinuities. In particular, thin structures, vehicle contours, and object boundaries are recovered accurately across diverse daytime and nighttime driving scenes.

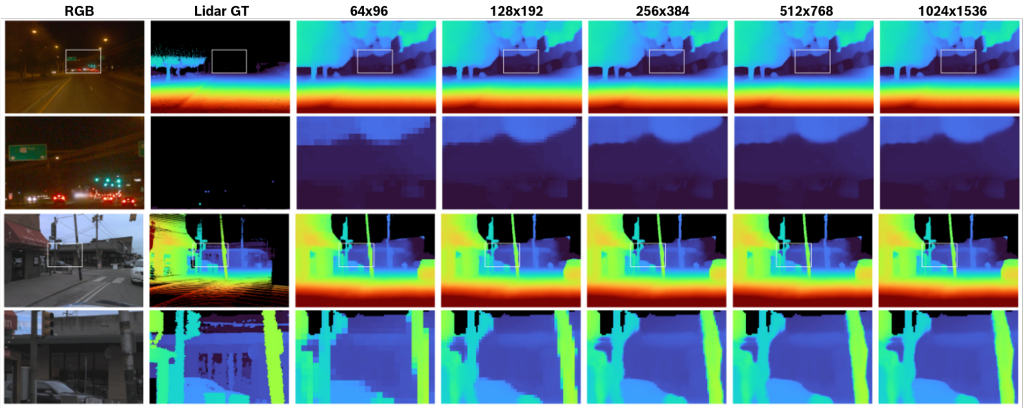


Figure 3: **Visualizing dense-query detail recovery**.

C EVALUATION METRICS

Accuracy and completeness. Following standard point-cloud reconstruction evaluation Wang et al. (2025a), we report accuracy (Acc) and completeness (Comp). For each camera c and frame t, Acc measures the mean nearest-neighbour distance from the aligned prediction $\tilde{\mathcal{P}}_{c,t}$ to the ground truth $\mathcal{G}_{c,t}$, while Comp measures the reverse direction:

$$\text{Acc} = \frac{1}{VT}\sum_{c,t} \text{mean}_{\tilde{\mathbf{g}}\in\tilde{\mathcal{P}}_{c,t}} \min_{\mathbf{g}\in\mathcal{G}_{c,t}} \|\tilde{\mathbf{g}} - \mathbf{g}\|, \tag{7}$$

$$\text{Comp} = \frac{1}{VT}\sum_{c,t} \text{mean}_{\mathbf{g}\in\mathcal{G}_{c,t}} \min_{\tilde{\mathbf{g}}\in\tilde{\mathcal{P}}_{c,t}} \|\mathbf{g} - \tilde{\mathbf{g}}\|. \tag{8}$$

Ray depth. We report the standard AbsRel and $\delta < 1.25$ depth metrics Eigen et al. (2014). In our evaluation, depth is defined as the ray length from the frame-wise ego origin rather than the camera center: $r = \|E_t^{-1}T_{0,0}\mathbf{g}\|$ and $\hat{r} = \|E_t^{-1}T_{0,0}\tilde{\mathbf{g}}\|$. Thus,

$$\text{AbsRel} = \frac{1}{N}\sum_i \frac{|\hat{r}_i - r_i|}{r_i}, \qquad \delta_{1.25} = \frac{1}{N}\sum_i \mathbb{I}\left[\max\left(\frac{\hat{r}_i}{r_i}, \frac{r_i}{\hat{r}_i}\right) < 1.25\right]. \tag{9}$$

Ego-pose estimation. Following DVGT Zuo et al. (2026), we evaluate ego-pose estimation using AUC@30. For each valid frame pair (i, j), we compute the relative rotation error (RRA) as the geodesic distance between the predicted and ground-truth relative rotations, and the relative translation error (RTA) as the angular deviation between their translation vectors. The pose error is defined as

$$e_{ij} = \max\left(e_{ij}^R, e_{ij}^\tau\right). \tag{10}$$

The pose accuracy at threshold τ is

$$\text{Acc}(\tau) = \frac{1}{|\mathcal{Q}|}\sum_{(i,j)\in\mathcal{Q}} \mathbb{I}[e_{ij} < \tau], \tag{11}$$

where $\mathcal{Q}$ denotes the set of valid frame pairs. AUC@30 is then computed as

$$\text{AUC@30} = \frac{1}{30}\int_0^{30} \text{Acc}(\tau)\, d\tau. \tag{12}$$

3D point tracking. We follow the TAPVid-3D protocol Koppula et al. (2024) and report average 3D position accuracy (APD3D), occlusion accuracy (OA), and 3D average Jaccard (AJ3D) and Recall. For a target with GT depth z^{gt}, correctness is evaluated at depth-adaptive thresholds

$$\delta_k = k\, z^{\mathrm{gt}}/f, \qquad k \in \{1, 2, 4, 8, 16\}, \quad f = \sqrt{f_x f_y}. \tag{13}$$

APD3D averages the fraction of visible points within these thresholds, OA measures visibility classification accuracy, and AJ3D jointly evaluates position and visibility.

Since autonomous-driving datasets do not provide manually annotated dense 3D point trajectories, we derive point-level tracking targets from the official temporally associated 3D box annotations. For each object k, the annotated box pose $B_{k,t}$ is first transformed to the world coordinate system and temporally aligned with the camera timestamps. Translation is linearly interpolated and rotation is interpolated in quaternion space when annotations and image timestamps are asynchronous.

We classify an object as dynamic if the velocity of its box center in the scene-0 coordinate system is at least $1\,\mathrm{m/s}$. For a LiDAR point observed inside a dynamic source box at time t_s (with a $5\,\mathrm{cm}$ box enlargement), we record its object-local coordinate $\mathbf{l}$. Assuming rigid object motion, its position at target time t is obtained as

$$\mathbf{w}_t = B_{k,t}\mathbf{l}. \tag{14}$$

The point is then transformed either to the source-camera frame or the target-time camera frame to obtain the corresponding world-frame and camera-frame tracking targets. Visibility is determined using camera-frustum validity together with LiDAR-based occlusion testing.

C.1 Motion Decomposition and Training Objectives

We provide additional implementation details for the structured motion decomposition and trajectory objective introduced in Sec. 2.4 and Sec. 2.5.

Motion-slot assignment. Given the query feature $\mathbf{f}_q$, the motion assignment head predicts a categorical distribution over one background slot and K foreground motion slots:

$$\boldsymbol{\alpha}_q = \mathrm{softmax}(g(\mathbf{f}_q)) = [\alpha_{q,0}, \alpha_{q,1}, \ldots, \alpha_{q,K}] \in \mathbb{R}^{K+1}, \tag{15}$$

where

$$\sum_{k=0}^{K} \alpha_{q,k} = 1. \tag{16}$$

Here, $\alpha_{q,0}$ denotes the probability that query q belongs to the static background, while $\alpha_{q,k}$, $k = 1, \ldots, K$, denotes its assignment probability to the k-th foreground motion slot. We uses K=16 actor motion slots, plus one static background slot and one ego token, for 18 learned slot queries in total.

Each foreground slot $\mathbf{s}_k$ predicts a time-conditioned motion prototype

$$\mathbf{m}_k(t) = M(\mathbf{s}_k, t). \tag{17}$$

The shared foreground-slot displacement between source time t_s and target time t is

$$\Delta\mathbf{X}_{\mathrm{slot}}(q, t) = \sum_{k=1}^{K} \alpha_{q,k} \left[\mathbf{m}_k(t) - \mathbf{m}_k(t_s)\right]. \tag{18}$$

A query-specific residual branch captures local motion that cannot be fully represented by the shared motion slots:

$$\Delta\mathbf{X}_{\mathrm{res}}(q, t) = (1 - \alpha_{q,0}) \left[\mathbf{r}(\mathbf{f}_q, t) - \mathbf{r}(\mathbf{f}_q, t_s)\right]. \tag{19}$$

The foreground gate $(1 - \alpha_{q,0})$ suppresses query-specific residual motion for points assigned to the static background.

The final queried trajectory is computed as

$$\hat{\mathbf{X}}_q(t) = \hat{\mathbf{X}}_q(t_s) + \beta_s \Delta\mathbf{X}_{\mathrm{slot}}(q, t) + \beta_r \Delta\mathbf{X}_{\mathrm{res}}(q, t). \tag{20}$$

Here, β_s and β_r are fixed scalar coefficients controlling the relative contributions of the shared slot-motion and query-specific residual branches. They are shared across all queries, scenes, and datasets, and are not learned or conditioned on the query. We fix $\beta_s = \beta_r = 1$ throughout training and inference; thus, the slot-motion and residual branches are combined without additional learned rescaling.

Valid trajectory supervision. Let $\mathcal{Q}_{\text{sta}}$ and $\mathcal{Q}_{\text{dyn}}$ denote the sets of static and dynamic queries, respectively. For each query q and target time t, we define a validity mask $M_{q,t} \in \{0, 1\}$ indicating whether a valid trajectory target is available. All trajectory-related losses are evaluated only on valid query–time pairs.

For a dynamic query, we further define its normalized foreground-slot distribution as

$$\tilde{\boldsymbol{\alpha}}_q = \frac{\boldsymbol{\alpha}_{q,1:K}}{1 - \alpha_{q,0} + \epsilon}, \tag{21}$$

where ϵ is a small constant for numerical stability.

For a tracked object o observed at source time t, we define the object-level foreground-slot assignment as

$$\boldsymbol{\mu}_{o,t} = \frac{1}{|\mathcal{Q}_{o,t}|} \sum_{q \in \mathcal{Q}_{o,t}} \tilde{\boldsymbol{\alpha}}_q, \tag{22}$$

where $\mathcal{Q}_{o,t}$ contains all valid dynamic queries associated with object o at time t.

Trajectory-position loss. Let $\mathbf{X}_q^*(t)$ denote the target metric 3D position of query q at target time t. We supervise the predicted point trajectory using the Huber loss:

$$\mathcal{L}_{\text{pos}} = \frac{\sum_{q,t} M_{q,t} \, \text{Huber}_\delta \left(\hat{\mathbf{X}}_q(t) - \mathbf{X}_q^*(t) \right)}{\sum_{q,t} M_{q,t} + \epsilon}, \qquad \delta = 1 \text{ m}. \tag{23}$$

Motion-grouping loss. The grouping objective encourages static queries to select the background slot and queries belonging to the same tracked actor to share a consistent foreground-slot assignment:

$$\begin{aligned} \mathcal{L}_{\text{group}} = & -\frac{1}{|\mathcal{Q}_{\text{sta}}|} \sum_{q \in \mathcal{Q}_{\text{sta}}} \log \alpha_{q,0} \\ & + \frac{1}{|\mathcal{Q}_{\text{dyn}}|} \sum_{q \in \mathcal{Q}_{\text{dyn}}} \left\| \tilde{\boldsymbol{\alpha}}_q - \boldsymbol{\mu}_{o(q),t_s(q)} \right\|_1 . \end{aligned} \tag{24}$$

No semantic identity is assigned to a particular foreground slot. Applying the same permutation to all foreground-slot indices therefore leaves the objective unchanged, making the grouping objective permutation-invariant without requiring a predefined slot-to-object correspondence.

Temporal assignment consistency. A tracked actor should maintain a consistent motion-slot assignment across different source timestamps. We therefore define

$$\mathcal{L}_{\text{cons}} = \frac{1}{|\mathcal{P}|} \sum_{(o,t,t') \in \mathcal{P}} \left\| \boldsymbol{\mu}_{o,t} - \boldsymbol{\mu}_{o,t'} \right\|_1 , \tag{25}$$

where $\mathcal{P}$ contains valid pairs of observations corresponding to the same tracked actor at different timestamps.

Slot-balancing loss. To prevent all foreground actors from collapsing into a single motion slot, we regularize the average foreground-slot utilization within each clip. Let

$$\bar{\boldsymbol{\mu}} = \frac{1}{|\mathcal{O}|} \sum_{(o,t) \in \mathcal{O}} \boldsymbol{\mu}_{o,t}, \tag{26}$$

where $\mathcal{O}$ denotes the set of valid object–time instances in the clip. The balancing objective is

$$\mathcal{L}_{\text{bal}} = \sum_{k=1}^{K} \left(\bar{\mu}_k - \frac{1}{K} \right)^2 . \tag{27}$$

Residual-motion regularization. The residual branch is intended to model local deviations rather than absorb the complete object motion. We therefore penalize its displacement magnitude:

$$\mathcal{L}_{\text{res}} = \frac{\sum_{q,t} M_{q,t} \left\| \Delta \mathbf{X}_{\text{res}}(q,t) \right\|_1}{\sum_{q,t} M_{q,t} + \epsilon}. \tag{28}$$

This regularization encourages motion shared by points on the same actor to be represented by the foreground motion slots whenever possible.

Ego-motion loss. Let $(\hat{\mathbf{R}}_t, \hat{\mathbf{t}}_t)$ and $(\mathbf{R}_t^*, \mathbf{t}_t^*)$ denote the predicted and ground-truth ego poses at time t, respectively. Ego motion is supervised using rotation and translation errors:

$$\mathcal{L}_{\text{ego}} = \frac{1}{T} \sum_{t=1}^{T} \left[d_{\text{SO(3)}} \left(\hat{\mathbf{R}}_t, \mathbf{R}_t^* \right) + \lambda_{\text{trans}} \left\| \hat{\mathbf{t}}_t - \mathbf{t}_t^* \right\|_1 \right], \tag{29}$$

where $d_{\text{SO(3)}}$ denotes the geodesic rotation distance and $\lambda_{\text{trans}} = 1$ in all experiments.

Complete trajectory objective. Combining the above terms, the trajectory objective is

$$\mathcal{L}_{\text{traj}} = \mathcal{L}_{\text{pos}} + \lambda_g \mathcal{L}_{\text{group}} + \lambda_e \mathcal{L}_{\text{ego}} + \lambda_c \mathcal{L}_{\text{cons}} + \lambda_b \mathcal{L}_{\text{bal}} + \lambda_r \mathcal{L}_{\text{res}}. \tag{30}$$

We use

$$\lambda_g = \mathbf{0.1}, \quad \lambda_e = \mathbf{0.1}, \quad \lambda_c = \mathbf{0.1}, \quad \lambda_b = \mathbf{0.05}, \quad \lambda_r = \mathbf{0.01} \tag{31}$$

for all datasets and experiments.

Together with the geometric objective introduced in the main paper, the complete training loss is

$$\mathcal{L} = \mathcal{L}_{\text{geo}} + \mathcal{L}_{\text{traj}}. \tag{32}$$

Geometric loss weights. The geometric training objective is defined as

$$\mathcal{L}_{\text{geo}} = \lambda_{\text{3D}} \mathcal{L}_{\text{3D}} + \lambda_{\text{proj}} \mathcal{L}_{\text{proj}} + \lambda_{\text{vis}} \mathcal{L}_{\text{vis}} + \lambda_{\text{disp}} \mathcal{L}_{\text{disp}} + \lambda_{\text{normal}} \mathcal{L}_{\text{normal}} + \lambda_{\text{conf}} \mathcal{L}_{\text{conf}},$$

where we use

$$\lambda_{\text{3D}} = 1.0, \quad \lambda_{\text{proj}} = 0.1, \quad \lambda_{\text{vis}} = 0.1, \quad \lambda_{\text{disp}} = 0.1, \quad \lambda_{\text{normal}} = 0.4, \quad \lambda_{\text{conf}} = 0.2.$$