Motion Cues from Image-based Point Tracking for LiDAR Scene Flow Estimation

Youngdong Jang[*,1] **Gyeongrok Oh**[*,1] **Jong Wook Kim**[1] **Hyunju Ryu**[1]
Hyung-gun Chi[2] **SeungHyeon Kim**[1] **Seungryong Kim**[3] **Jonghyun Choi**[4] **Sangpil Kim**[†,1]

[1] Korea University [2] Purdue University [3] KAIST [4] Hyundai Motor Company

Abstract

Self-supervised LiDAR scene flow approaches rely on pseudo-labels that distinguish static and dynamic points, providing targeted supervision. These labels are typically derived from LiDAR geometry, making them unreliable in sparse or occluded regions. The resulting noise misguides motion supervision and degrades scene flow learning, yet existing methods remain largely dependent on the same geometric evidence. To address this, we introduce TRACKCUE, a tracking-guided framework that transfers dense image trajectories into reliable static-dynamic supervision. TRACKCUE anchors tracking queries to projected LiDAR points, yielding image-space trajectories that provide motion evidence beyond sparse LiDAR geometry. Since image-space displacement also reflects ego-vehicle motion, we present a visually consistent motion compensation strategy that compares each tracked trajectory with the rigid trajectory induced by ego-motion alone. It admits motion evidence only where both trajectories are valid and the deviation persists across frames, isolating true object motion while suppressing transient tracking errors. Finally, we perform visual motion cue lifting, which associates image-space object motion cues with LiDAR points for label refinement. As a result, TRACKCUE substantially improves the precision of dynamic pseudo-labels, providing more reliable supervision and leading to better self-supervised scene flow estimation.

1 Introduction

LiDAR scene flow estimation [1, 2, 3, 4] is essential for autonomous driving, as it provides point-level 3D motion from consecutive point clouds. However, accurate scene flow learning is challenging due to the high cost of ground-truth annotations. To reduce this reliance on manual labels, self-supervised approaches learn scene flow directly from point cloud sequences. Recent self-supervised approaches [5, 6, 7, 8] can be divided into optimization-based and feed-forward methods. In this work, we focus on feed-forward methods [9, 7, 10], which are more suitable for practical deployment than optimization-based methods by avoiding costly per-scene optimization at test time.

A key challenge in this setting is the severe imbalance between static and dynamic points. Most LiDAR points in driving scenes are static. Without accounting for this imbalance, self-supervision biases the model toward predicting zero flow. To address this, existing approaches [8, 7] first separate static and dynamic points, applying different supervision to each. This static-dynamic separation typically relies on ray-casting-based dynamic-awareness mapping, such as DUFOMap [11], which is constructed only from LiDAR point clouds. Specifically, ray-casting identifies regions traversed by rays as voids and classifies points later observed in those regions as dynamic. However, this procedure depends on dense and consistent LiDAR observations. Sparse or occluded observations yield unreliable void estimates, causing static points to be mislabeled as dynamic and moving points

* Equal contribution. † Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

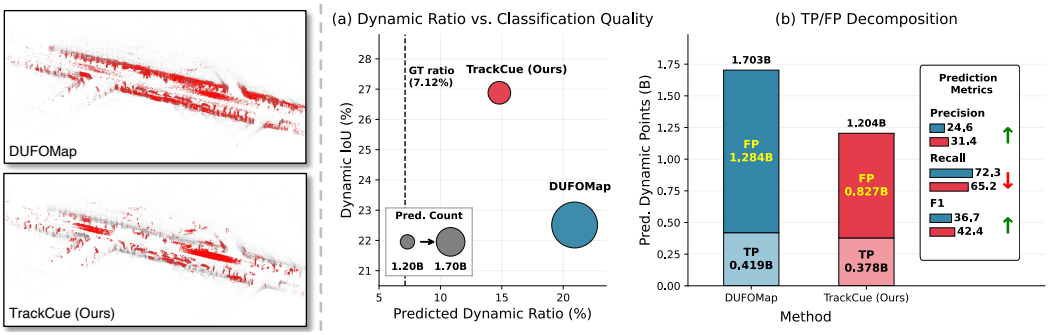


Figure 1: Analysis of dynamic auto-label quality for self-supervised LiDAR scene flow. (Left) TRACKCUE suppresses static regions while preserving true object motion. (Right) Compared with DUFOMap [11], TRACKCUE reduces dynamic over-prediction and false positives, improving Dynamic IoU, precision, and F1. The dashed line denotes the ground-truth dynamic ratio.

to be missed. The resulting noise misguides motion supervision and degrades scene flow learning, while refinements based solely on LiDAR geometry [7] remain vulnerable to sparsity and occlusion.

A natural approach is to track points in RGB images, where object motion remains visually continuous despite sparse or occluded LiDAR. However, naive point tracking is insufficient. Image motion mixes object and ego-induced motion, and tracking suffers from transient correspondence and visibility errors. Even with reliable trajectories, image-space motion cues do not directly transfer to sparse LiDAR observations.

To address these issues, we introduce TRACKCUE, a tracking-guided framework that transfers temporally consistent image trajectories into reliable static-dynamic supervision. TRACKCUE anchors tracking queries to projected LiDAR points, maintaining correspondence between image-space trajectories and 3D observations. As shown in Figure 3, static points exhibit substantial image-space displacement under ego-vehicle motion. Since this displacement alone cannot distinguish object motion from ego-motion, we present a visually consistent motion compensation strategy. Specifically, TRACKCUE resolves this ambiguity by comparing each tracked trajectory with the rigid trajectory of the same point under ego-motion alone, and identifies points that consistently deviate from it as independently moving. A joint visibility mask with temporal voting then restricts this comparison to frames where both the tracked trajectory and its rigid projection are valid, and requires the deviation to persist across frames, suppressing transient tracking errors. Finally, TRACKCUE performs visual motion cue lifting, transferring the ego-compensated cues back to the LiDAR domain by associating their image-space locations with projected LiDAR points through nearest-neighbor search. The resulting labels complement LiDAR-only observations and provide more reliable supervision for self-supervised scene flow learning.

Figure 1 shows that TRACKCUE produces more reliable pseudo-labels than DUFOMap [11]. It reduces false-positive dynamic labels by 35.6%, while improving precision from 24.6% to 31.4% and F1 score from 36.7% to 42.4%. Our results show that TRACKCUE improves dynamic label reliability and scene flow performance. Our contributions are summarized as follows:

- We introduce TRACKCUE, a tracking-guided framework that leverages dense image trajectories to complement sparse LiDAR geometry for reliable static-dynamic supervision.
- We propose a visually consistent motion compensation strategy with joint visibility masking and temporal voting to distinguish object-motion cues from ego-induced apparent motion while suppressing transient tracking errors.
- We present visual motion cue lifting, which transfers reliable image-space motion evidence back to the LiDAR domain for static-dynamic label refinement, leading to improved pseudo-label quality and downstream scene flow performance.

2 Related Works

Scene Flow Estimation aims to estimate the 3D motion vector of moving objects in dynamic scenes. In autonomous driving, existing works have explored both supervised [3, 4, 12, 13, 14, 15]

and self-supervised learning [16, 7, 8, 9, 10, 5, 6, 17, 1, 2] paradigms. Recently, self-supervised approaches have gained increasing attention, as dense 3D annotations are costly and difficult to scale. One prominent line of work [5, 6, 1, 2] performs per-scene optimization at test time. While these approaches exploit long-term temporal context to estimate accurate flow fields, their computational overhead limits real-time deployment. Another line of work [16, 7, 8, 9, 10] uses feed-forward models pretrained on large-scale unlabeled data, enabling flow prediction in a single forward pass. Importantly, the self-supervised approaches highly depend on pseudo-labels based on dynamic object clustering. DUFOMap [11] is an efficient dynamic-awareness mapping framework that separates moving objects from the static environment. It enables reliable flow supervision across consecutive point clouds, even without manually annotated labels. However, inherent limitations of LiDAR, including sparse observations, temporally varying point measurements, and occlusions, can lead to dynamic object misclassifications. Such erroneous dynamic labels disrupt training stability and ultimately degrade the model performance. To this end, our proposed TRACKCUE refines dynamic object clustering informed by dense visual cues from RGB images.

Multi-modal Motion Understanding has become a widely adopted paradigm in safety-critical vision systems, particularly in autonomous driving perception. By combining the accurate geometric localization of LiDAR with the dense appearance and semantic cues of RGB images, camera–LiDAR fusion [18, 19, 20] has shown strong effectiveness in various 3D perception tasks, including object detection [21, 22], semantic segmentation [23, 24], and occupancy prediction [25, 26]. Furthermore, cross-modal distillation [27, 28, 29] has also been explored to transfer complementary knowledge across modalities, thereby improving feature representation for downstream tasks. Despite these advances in 3D perception, multi-modal reasoning remains relatively underexplored in LiDAR scene flow estimation [30, 3], especially under self-supervised settings. Inspired by these developments, our work pioneers a new perspective on scene flow learning, which improves the reliability of self-supervised dynamic labels.

Point Tracker in videos has made tremendous progress in recent years. It demonstrates a remarkable ability to understand the temporal evolution of scenes while capturing the physical dynamics of moving instances. Originally, the task of tracking any point [31, 32] was first formalized by the classical optical flow estimation problem, consistently generating point trajectories with extended temporal window instead of 2-frames. Afterwards, the series of TAP [32, 33, 34, 35, 36, 37] and CoTracker [38, 39] are built upon modern transformer architectures and trained on large-scale video datasets, including both labeled and unlabeled, real and synthetic data. Beyond these representative works, a growing number of studies [40, 41] have further advanced this direction, exploring more robust and scalable point tracking frameworks and broadening their applicability [42, 43, 44, 45]. In this paper, we argue that foundation point trackers can serve as a powerful source of visual motion cues for LiDAR scene flow estimation, complementing sparse geometric observations with dense and temporally consistent RGB-based trajectories.

3 Method

3.1 Preliminary

Problem Statement. LiDAR scene flow estimation aims to predict a 3D motion vector $\mathbf{f}_{i,t} \in \mathbb{R}^3$ for each LiDAR point, describing its motion to the next frame. Given consecutive LiDAR point clouds $\{\mathcal{P}_t, \mathcal{P}_{t+1}, \dots\}$, $\mathcal{P}_t = \{\mathbf{p}_{i,t}\}_{i=1}^{M_t}$ denotes the point cloud at time t with M_t points. For all points in $\mathcal{P}_t$, the predicted scene flow is represented as $\mathcal{F}_t = \{\mathbf{f}_{i,t}\}_{i=1}^{M_t}$ from $\mathcal{P}_t$ to $\mathcal{P}_{t+1}$. Since autonomous driving scenes are observed from a moving vehicle, the scene flow is decomposed into ego-motion flow $\mathcal{F}_t^{\text{ego}}$ and residual flow $\Delta\mathcal{F}_t$ as follows:

$$\mathcal{F}_t = \mathcal{F}_t^{\text{ego}} + \Delta\mathcal{F}_t. \tag{1}$$

While $\mathcal{F}^{\text{ego}}$ can be derived from vehicle odometry, the network is trained to estimate $\Delta\mathcal{F}_t$.

Static-Dynamic Self-Supervision. To derive supervision without labeled data, self-supervised approaches first divide points into static and dynamic sets, denoted as $\mathcal{P}_t^{\text{s}}$ and $\mathcal{P}_t^{\text{d}}$, respectively. This classification allows different supervision signals to be applied according to the motion type of each point. The overall self-supervised objective is formulated by combining static and dynamic losses:

$$\mathcal{L}_{\text{ssl}} = \mathcal{L}_{\text{s}}(\mathcal{P}_t^{\text{s}}) + \mathcal{L}_{\text{d}}(\mathcal{P}_t^{\text{d}}), \tag{2}$$

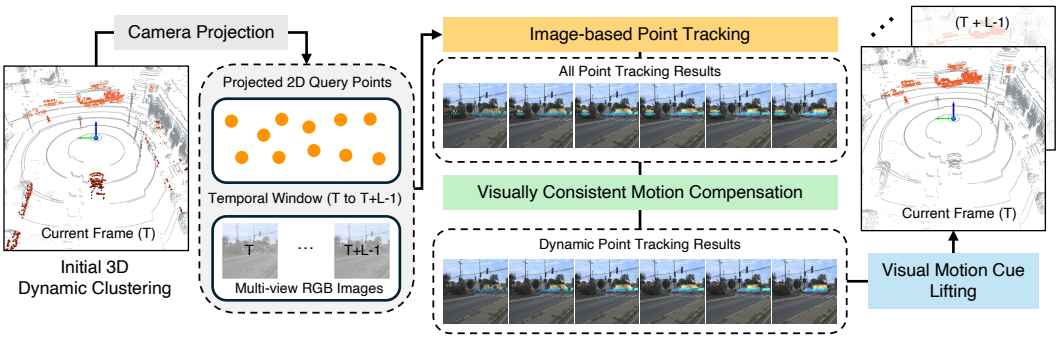


Figure 2: Overview of the TRACKCUE framework.

where $\mathcal{L}_s$ constrains static points to have near-zero residual motion, whereas $\mathcal{L}_d$ supervises dynamic points through Chamfer distance and dynamic clustering terms. Therefore, reliable static-dynamic classification is crucial, as it determines whether each point receives appropriate supervision. More detailed descriptions of the labeling process are provided in Appendix B.

3.2 Overview

TRACKCUE improves static-dynamic labeling by transferring motion evidence from RGB images into the LiDAR domain. As shown in Figure 2, TRACKCUE tracks projected LiDAR points in RGB images, compensates for ego-induced motion, suppresses unreliable trajectories, and lifts the resulting motion cues back to the LiDAR domain. The pipeline consists of three main components:

- **Image-based Point Tracking:** Anchors queries to projected LiDAR points and extracts temporally consistent trajectories from multi-frame images. These trajectories provide temporally consistent visual motion cues for identifying dynamic points that are ambiguous from LiDAR alone.
- **Visually Consistent Motion Compensation:** Resolves motion ambiguity and trajectory uncertainty. It constructs the rigid trajectory of each query under ego-motion alone and measures the residual against the observed track. A joint visibility mask removes invalid frames, while temporal voting retains only deviations that persist across frames.
- **Visual Motion Cue Lifting:** Bridges the image-to-LiDAR correspondence gap. It lifts ego-compensated image motion cues to LiDAR points through nearest-neighbor search, refining static-dynamic labels with image-space motion evidence.

3.3 Image-based Point Tracking

We propose an Image-based Point Tracking module to obtain temporally consistent motion cues from RGB images. While prior self-supervised methods derive static-dynamic labels from LiDAR geometry alone, our module exploits temporally continuous visual cues to complement sparse or occluded observations.

LiDAR-to-Image Projection. To initialize the image-based tracking, we first project the LiDAR points labeled by DUFOMap [11] onto the image plane. For each LiDAR point $\mathbf{p}_{i,t} \in \mathcal{P}_t^{\text{dufo}}$, its pixel coordinate $\mathbf{u}_{i,t} = (u_{i,t}, v_{i,t})^\top$ is derived by applying the extrinsic transformation $\mathbf{T}$ and the camera intrinsic matrix $\mathbf{K}$:

$$\begin{bmatrix} \mathbf{p}_{i,t}^{\text{c}} \\ 1 \end{bmatrix} = \mathbf{T} \cdot \begin{bmatrix} \mathbf{p}_{i,t} \\ 1 \end{bmatrix}, \quad \mathbf{u}_{i,t} = \pi\left(\mathbf{K} \cdot \mathbf{p}_{i,t}^{\text{c}}\right), \tag{3}$$

where $\mathbf{p}_{i,t}^{\text{c}} = (x_{i,t}^{\text{c}}, y_{i,t}^{\text{c}}, z_{i,t}^{\text{c}})^\top$ and $\pi(\cdot)$ denote the camera coordinate and the standard perspective projection, respectively. To ensure reliable tracking initialization, we retain only projected points that satisfy the following constraints:

$$z_{i,t}^{\text{c}} > d_{\min}, \quad b \le u_{i,t} < W - b, \quad b \le v_{i,t} < H - b, \tag{4}$$

where $z_{i,t}^{\text{c}}$, (W, H), and b denote the camera-frame depth, resolution, and boundary margin, respectively. The retained projections form the query set $\mathcal{Q}_t = \{\mathbf{q}_{i,t}\}_{i=1}^{N_t}$.

Point Tracking. Given $\mathcal{Q}_t$, we estimate trajectories over a clip of length L using a pre-trained point tracker [39, 40]. Unlike conventional optical flow, point trackers maintain long-term temporal consistency and remain robust under occlusion. For each initial query $\mathbf{q}_{i,t}$, the point tracker predicts its corresponding image position in each subsequent frame. We denote the tracked trajectory as:

$$\hat{\mathcal{T}}_i = \{\hat{\mathbf{u}}_{i,k}\}_{k=t}^{t+L-1}, \tag{5}$$

where $\hat{\mathbf{u}}_{i,k}$ is the estimated pixel coordinate at frame k. The tracker also predicts a visibility mask $w_{i,k}^{\text{tracker}} \in \{0,1\}$, where $w_{i,k}^{\text{tracker}} = 1$ indicates that the tracked point is predicted to be visible at frame k. Since tracking operates in the image domain, $\hat{\mathcal{T}}_i$ can remain continuous across frames even when the corresponding LiDAR observations are sparse or absent.

3.4 Visually Consistent Motion Compensation

We present a visually consistent motion compensation to distinguish true object motion from ego-induced apparent motion in image-space trajectories. Although a point tracker provides robust trajectories, its predictions capture image-space motion, which contains both independent object motion and ego-induced apparent motion. Thus, motion compensation is required to separate true object motion from ego-induced apparent motion in the image-space trajectories. To ensure that the motion evidence is reliable, a joint visibility mask filters out frames where either trajectory is invalid, while temporal voting retains only deviations that persist across multiple frames.

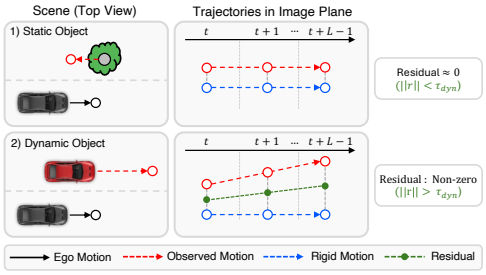


Figure 3: Illustration of our motion compensation.

Rigid Trajectory and Residual. Assuming each query $\mathbf{q}_{i,t}$ corresponds to a LiDAR point in the static scene, we construct a rigid trajectory $\hat{\mathcal{T}}_i^{\text{rigid}} = \{\hat{\mathbf{u}}_{i,k}^{\text{rigid}}\}_{k=t}^{t+L-1}$ for that point. This trajectory represents image-space displacement induced solely by ego-motion. Using the global ego-poses $\mathbf{G} \in SE(3)$, the rigid pixel coordinate $\hat{\mathbf{u}}_{i,k}^{\text{rigid}}$ at frame k is obtained by applying the relative ego-motion from frame t to k and projecting the transformed point onto the target image plane:

$$\hat{\mathbf{u}}_{i,k}^{\text{rigid}} = \pi\left(\mathbf{K} \cdot \mathbf{T} \cdot \mathbf{G}_k^{-1} \cdot \mathbf{G}_t \cdot \begin{bmatrix}\mathbf{p}_{i,t} \\ 1\end{bmatrix}\right), \tag{6}$$

where $\mathbf{G}_t$ and $\mathbf{G}_k$ represent the global ego-poses at the reference and target frames, respectively. The term $\mathbf{G}_k^{-1} \cdot \mathbf{G}_t$ gives the relative transformation from frame t to frame k. To ensure a valid rigid projection, we construct a rigid visibility mask $w_{i,k}^{\text{rigid}} \in \{0,1\}$ for each $\hat{\mathbf{u}}_{i,k}^{\text{rigid}}$, using the projection constraints in Eq. 4. The residual between the tracked and rigid trajectories isolates motion unexplained by ego-motion:

$$r_{i,k} = \| \hat{\mathbf{u}}_{i,k} - \hat{\mathbf{u}}_{i,k}^{\text{rigid}} \|_2 . \tag{7}$$

For static points, the residual $r_{i,k}$ remains close to zero, whereas true dynamic points exhibit non-zero residuals. Therefore, we use the residual as the criterion for identifying independently moving points.

Joint Visibility and Temporal Voting. The residual is reliable only when both the tracked and rigid trajectories are valid. If the tracked point is occluded or the rigid projection falls outside the image, invalid coordinates can produce a large residual unrelated to object motion. Such a residual may be interpreted as movement. We therefore define a joint visibility mask $w_{i,k}^{\text{joint}}$ that admits a frame into the comparison only when both are valid:

$$w_{i,k}^{\text{joint}} = w_{i,k}^{\text{tracker}} \cdot w_{i,k}^{\text{rigid}}. \tag{8}$$

Even under valid visibility, a large residual at a single time step can arise from transient correspondence errors. To suppress such errors, we apply temporal voting by counting the jointly valid frames in which the residual exceeds τ_{dyn}. Based on V_i, we collect the candidate moving trajectory indices:

$$V_i = \sum_{k=t}^{t+L-1} \mathbf{1}\left[r_{i,k} > \tau_{\text{dyn}}\right] w_{i,k}^{\text{joint}}, \quad \mathcal{I}^{\text{mov}} = \{i \mid V_i \geq N_{\text{move}}\}, \tag{9}$$

where $\mathbf{1}[\cdot]$, τ_{dyn}, and N_{move} denote the indicator function, residual threshold and minimum number of moving frames, respectively. For each camera view, we retain the selected trajectories only when at least N_{point} moving queries are found, requiring the motion cue to be supported by multiple queries rather than a few isolated detections:

$$\mathcal{T}^{\mathrm{mov}} = \begin{cases} \left\{ \hat{\mathcal{T}}_i \mid i \in \mathcal{I}^{\mathrm{mov}} \right\}, & |\mathcal{I}^{\mathrm{mov}}| \geq N_{\mathrm{point}}, \\ \emptyset, & \text{otherwise.} \end{cases} \tag{10}$$

Although $\mathcal{T}^{\mathrm{mov}}$ provides reliable evidence of independent motion, it remains defined on image trajectories and is not yet assigned to LiDAR points.

3.5 Visual Motion Cue Lifting

We lift the selected moving trajectories $\mathcal{T}^{\mathrm{mov}}$ to the LiDAR domain to obtain point-level dynamic evidence. Since these trajectories originate from only a subset of LiDAR points, they do not provide one-to-one correspondence with all dynamic points. Thus, we propagate the visual motion evidence to nearby LiDAR points. For each frame k, we use the visible positions of the selected trajectories and project the LiDAR points in the same frame onto the image plane. A KD-tree $\mathcal{D}_k$ is then constructed over the valid projected LiDAR coordinates. Given a selected trajectory $\hat{\mathcal{T}}_i \in \mathcal{T}^{\mathrm{mov}}$, its visible image-space position $\hat{\mathbf{u}}_{i,k}$ is associated with the K_{nn} nearest projected LiDAR points:

$$\mathcal{N}_{i,k} = \mathcal{D}_k(\hat{\mathbf{u}}_{i,k},\, K_{\mathrm{nn}})\,, \qquad d_{j,k} = \|\hat{\mathbf{u}}_{i,k} - \mathbf{u}_{j,k}\|_2, \tag{11}$$

where $\mathcal{N}_{i,k}$ denotes the index set of the K_{nn} nearest valid projected LiDAR points in frame k. Each matched LiDAR point is assigned a dynamic cue only when the nearest-neighbor distance is smaller than a lifting radius τ_{lift}:

$$e_{j,k} = \mathbf{1}[\, j \in \mathcal{N}_{i,k} \;\wedge\; d_{j,k} < \tau_{\mathrm{lift}}\,]\,. \tag{12}$$

Here, $e_{j,k}$ indicates whether LiDAR point $\mathbf{p}_{j,k}$ receives a lifted visual motion cue. These lifted cues complement the DUFOMap-based labels by recovering dynamic points that are difficult to identify from sparse LiDAR geometry alone. The resulting labels refine $\mathcal{P}^{\mathrm{dufo}}$ into $\mathcal{P}^{\textsc{TrackCue}}$, providing more reliable static-dynamic supervision.

4 Experiments

4.1 Experimental Setup

Datasets. We evaluate on Argoverse 2 [46], which contains 700 training and 150 validation sequences. The official public challenge [3] further evaluates methods on a diverse test set comprising five widely used LiDAR scene flow datasets: Argoverse 2, Waymo [47], nuScenes [48], TruckScenes [49], and AEVAScenes [50]. The test set includes 458 scenes and 9,613 frames randomly sampled from five source datasets. It provides broad coverage across driving environments and sensor configurations, such as urban and highway scenes captured with 32- and 64-beam LiDARs. For leaderboard evaluation, all scene flow models are trained only on the Argoverse 2 training split, with no additional scene flow training data.

Metrics. We report two representative scene flow metrics: three-way End Point Error (EPE) and Dynamic Bucket-Normalized EPE [13]. EPE measures the ℓ_2 distance between predicted and ground-truth flow vectors in centimeters. Three-way EPE averages EPE over Foreground Dynamic (FD), Foreground Static (FS), and Background Static (BS) points, while Bucket-Normalized EPE normalizes errors across speed buckets and semantic classes.

Baselines. We validate TRACKCUE by applying it to three self-supervised scene flow frameworks: SeFlow [8], SeFlow++ [7], and TeFlow [16]. SeFlow introduces self-supervised learning signals derived from dynamic point classification. SeFlow++ further leverages nearest-neighbor correspondences to filter static points according to their matching distances, improving geometric temporal consistency. TeFlow improves supervision reliability by integrating past trajectories within a multi-frame framework, while still relying on DUFOMap [11]-based static-dynamic classification. For a fair comparison, we build on the official implementations of all baselines and retain their provided training configurations. We provide further experimental details in Appendix A.

Table 1: Experimental results on the Argoverse 2 leaderboard [3]. We report Dynamic Bucket Normalized EPE [13], where lower is better. Small arrow indicates the scaled absolute score difference from the corresponding baseline, i.e., $|\text{Ours} - \text{Baseline}| \times 100$. Green downward arrows denote improvement, while red upward arrows denote degradation.

Method	Overall		AV2 [46]		Waymo [47]		TruckScenes [49]		NuScenes [48]		Aeva [50]	
	Mean	0–35m	0–35m	35–70m	0–35m	35–70m	0–35m	35–70m	0–35m	35–70m	0–35m	35–70m
SeFlow [8]	0.7367	0.6289	0.3113	0.6638	0.3494	0.7639	0.8885	0.8509	0.7543	1.0514	0.8412	0.8926
+ TRACKCUE	**0.7344** (↓0.23)	**0.6286** (↓0.03)	**0.3070** (↓0.43)	0.6958 (↑3.20)	**0.3234** (↓2.60)	**0.7479** (↓1.60)	0.8968 (↑0.83)	0.8675 (↑1.66)	0.7933 (↑3.90)	**0.9910** (↓6.04)	**0.8227** (↓1.85)	0.8990 (↑0.64)
SeFlow++ [7]	0.7025	0.5919	0.2827	0.6552	0.3192	0.7253	**0.8555**	0.8473	0.6965	0.9748	0.8057	0.8631
+ TRACKCUE	**0.6619** (↓4.06)	**0.5537** (↓3.82)	**0.2781** (↓0.46)	**0.6313** (↓2.39)	**0.3041** (↓1.51)	**0.6339** (↓9.14)	0.8565 (↑0.10)	**0.7978** (↓4.95)	**0.6550** (↓4.15)	**0.9632** (↓1.16)	**0.6748** (↓13.09)	**0.8238** (↓3.93)
TeFlow [16]	0.6125	0.3116	0.2019	0.8826	0.1979	0.8920	0.3884	**0.9108**	0.3822	**0.9848**	0.3875	**0.8969**
+ TRACKCUE	**0.6057** (↓0.68)	**0.2987** (↓1.29)	**0.1966** (↓0.53)	**0.8734** (↓0.92)	**0.1870** (↓1.09)	**0.8872** (↓0.48)	**0.3762** (↓1.22)	0.9123 (↑0.15)	**0.3721** (↓1.01)	0.9867 (↑0.19)	**0.3619** (↓2.56)	0.9026 (↑0.57)

Table 2: Experimental results on the Argoverse 2 validation set [46]. Small arrow indicates the absolute score difference from corresponding baseline. Green downward arrows denote improvement, while red upward arrows denote degradation.

Method	#Frames	Three-way EPE (cm) ↓				Dynamic Bucket-Normalized ↓				
		Mean	FD	FS	BS	Mean	CAR	OTHER	PED.	VRU
Optimization-based										
NSFP [1]	2	7.19	12.44	4.52	4.62	0.406	0.218	0.284	0.698	0.422
FastNSF [2]	2	8.32	13.09	5.73	6.15	0.365	0.236	0.405	0.501	0.318
ICP-Flow [17]	2	6.44	12.40	3.75	3.19	0.346	0.229	0.354	0.450	0.351
Feed-forward										
ZeroFlow [10]	2	6.60	12.90	3.72	3.16	0.565	0.275	0.435	0.894	0.658
VoteFlow [9]	2	5.56	13.11	2.12	1.44	0.335	0.223	0.342	0.429	0.347
SeFlow [8]	2	4.88	11.37	1.97	1.31	0.372	0.215	0.378	0.551	0.343
+ TRACKCUE	2	**4.56** (↓0.32)	**10.58** (↓0.79)	**1.96** (↓0.01)	**1.14** (↓0.17)	**0.352** (↓0.020)	**0.194** (↓0.021)	**0.375** (↓0.003)	**0.518** (↓0.033)	**0.320** (↓0.023)
SeFlow++ [7]	3	4.78	9.48	3.05	1.80	0.328	0.182	0.334	0.508	0.290
+ TRACKCUE	3	**4.29** (↓0.49)	**9.15** (↓0.33)	**2.39** (↓0.66)	**1.31** (↓0.49)	**0.298** (↓0.030)	**0.176** (↓0.006)	**0.320** (↓0.014)	**0.422** (↓0.086)	**0.276** (↓0.014)
TeFlow [16]	5	3.59	7.18	**1.36**	2.23	0.222	**0.140**	0.285	**0.284**	**0.179**
+ TRACKCUE	5	**3.50** (↓0.09)	**7.01** (↓0.17)	1.51 (↑0.15)	**1.99** (↓0.24)	**0.218** (↓0.004)	0.145 (↑0.005)	**0.234** (↓0.051)	0.302 (↑0.018)	0.193 (↑0.014)

4.2 LiDAR Scene Flow Estimation Results

Quantitative Analysis. Table 1 reports scene flow estimation results on the Argoverse 2 challenge across five datasets. As seen from the table, TRACKCUE improves flow predictions across most datasets and distance ranges. The larger improvement for SeFlow++ highlights the importance of reliable static and dynamic separation for self-supervision. While SeFlow and TeFlow rely on DUFOMap-based pseudo-labels, SeFlow++ reassigns the dynamic labels by applying geometric distance-based filtering. Note that TRACKCUE further alleviates the sparsity and ambiguity inherent to LiDAR observations through temporally continuous motion cues from RGB images, ultimately leading to larger performance gains. TRACKCUE also yields substantial improvements across diverse datasets under the zero-shot evaluation. For example, TRACKCUE achieves a 13.09 point reduction in Dynamic Bucket-Normalized EPE on AEVAScenes, despite its different sensor configuration. These observations suggest that high-fidelity pseudo-labels improve the cross-dataset generalization of scene flow models. Detailed leaderboard results are provided in Table 9 in Appendix C. Table 2 presents results on the Argoverse 2 validation split. TRACKCUE shows the effectiveness of deriving visual motion supervision for scene flow prediction through detailed class-wise performance. The key insight is that TRACKCUE consistently reduces Foreground Dynamic (FD) error across all three baselines, supporting the effectiveness of the refined labels for dynamic-object supervision.

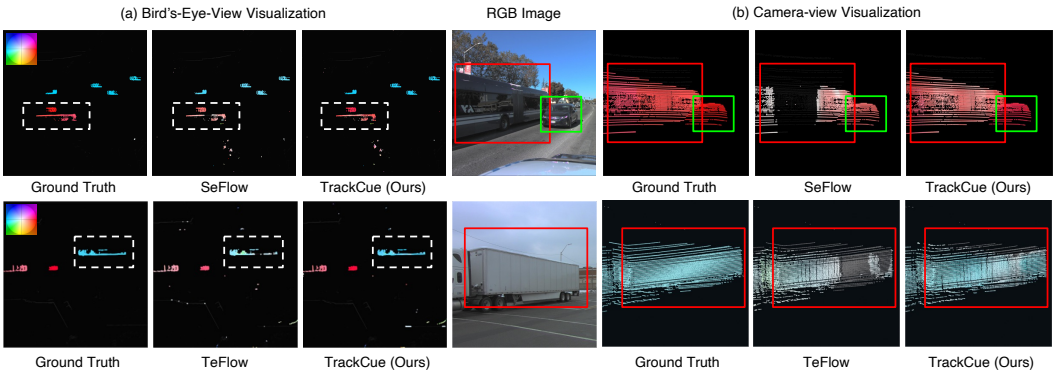


Figure 4: We visualize the flow estimation results in terms of two complementary perspectives. Motion is color-coded, where hue represents direction and intensity indicates speed.

Table 3: Comparison of dynamic pseudo-label generation quality and downstream scene flow performance. We compare the DUFOMap [11] and our TRACKCUE for auto-label generation of both SeFlow [8] and SeFlow++ [7] pipelines. The ground-truth dynamic ratio is 7.12%.

Auto-Labeling Method	Method	Pred. Dyn. Ratio (%)	Accuracy (%) ↑	Precision (%) ↑	Recall (%) ↑	F1 Score ↑	Dyn. Bucket ↓
SeFlow [8]	DUFOMap	20.93	82.26	24.62	**72.35**	36.74	0.372
	TRACKCUE (Ours)	14.80	**87.37**	**31.38**	65.22	**42.37**	**0.352**
SeFlow++ [7]	DUFOMap	10.05	91.89	45.07	**63.59**	52.75	0.328
	TRACKCUE (Ours)	7.18	**93.83**	**56.62**	57.09	**56.85**	**0.298**

Qualitative Analysis. Figure 4 visualizes the flow estimation results on some challenging scenarios. Here, the top example shows that TRACKCUE accurately estimates flow velocity, as indicated by the vivid color-coded motion. TRACKCUE also filters out points that are incorrectly classified as dynamic, preventing erroneous flow estimation in the background. In contrast, the baselines fail to predict the correct magnitude of the flow vectors, while also assigning non-zero motion to static background regions. Furthermore, the bottom example highlights the superiority of dense RGB motion cues. Specifically, the baselines miss motion predictions for parts of large instances. Since the central regions of these objects remain non-empty over short temporal windows, corresponding points are consequently misclassified as static by DUFOMap. However, TRACKCUE consistently assigns dynamic labels by focusing on image-space trajectories, rather than relying solely on sparse LiDAR observations. We present further qualitative results of TRACKCUE in Figures 9–11 of Appendix C.

4.3 Comparison of Auto-labeling for Scene Flow Estimation

In this section, we quantify the improvement in dynamic label fidelity using precision, recall, and F1 score, together with the Dynamic Bucket-Normalized EPE as the downstream metric. Following the official evaluation protocol, ground-truth dynamic labels are obtained by thresholding the residual motion between the ground-truth and the rigid flow at $0.05\,\mathrm{m}$ against which the predicted dynamic points of each auto-labeling method are evaluated. As shown in Table 3, TRACKCUE significantly reduces false-positive dynamic predictions and improves precision, while its conservative filtering leads to lower recall. Specifically, TRACKCUE improves the F1 score by 5.63 and 4.10 percentage points for SeFlow and SeFlow++, respectively. This indicates a better balance between filtering noisy labels and retaining true dynamic points. This trade-off is important, since false positives and false negatives affect the training objective differently. A false positive applies the Dynamic Chamfer and clustering terms to a stationary point and removes its static supervision. Such errors can be particularly harmful in driving scenes, where static points dominate and even a low false-positive rate can produce a substantial number of incorrect dynamic labels. In contrast, a false negative removes correct dynamic supervision from a moving point, but its effect can be partially compensated by other correctly labeled points from the same object. The cleaner supervision also improves downstream scene flow estimation, reducing the Dynamic Bucket-Normalized EPE from 0.372 to 0.352 for SeFlow and from 0.328 to 0.298 for SeFlow++.

Table 4: Effect of point tracker choice.

Metric	DUFOMap [11]	CoTracker3 [39]	TAPNext [37]	AllTracker [40]
GT Dyn. Ratio		7.12%		
Pred. Dyn. Ratio (%)	20.93	14.80	10.27	15.93
Precision (%) ↑	24.62	31.38	**35.30**	30.23
Recall (%) ↑	**72.35**	65.22	50.89	67.62
F1 Score ↑	36.74	**42.37**	41.68	41.78
Dyn. Bucket ↓	0.372	**0.352**	0.423	0.356

Table 5: Effect of image resolution.

Metric	786^2	512^2	256^2
GT Dyn. Ratio		7.12%	
Pred. Dyn. Ratio (%)	14.80	13.60	12.57
Precision (%) ↑	31.38	32.12	**32.78**
Recall (%) ↑	**65.22**	61.35	57.85
F1 ↑	**42.37**	42.16	41.85
Dyn. Bucket ↓	**0.352**	0.359	0.409

Table 6: Effect of input video clip length on dynamic clustering quality.

CoTracker3 [39]				
Clip Length (#Frames)	Pred. Dynamic Ratio (%)	Precision / Recall (%) ↑	F1 Score ↑	Dyn. Bucket ↓
3	12.84	**31.61** / 56.99	40.67	0.371
6	14.80	31.38 / 65.22	**42.37**	**0.352**
9	16.48	29.11 / **67.37**	40.65	0.357
12	16.54	28.35 / 65.90	39.64	0.373

4.4 Ablation Studies

Analysis on Point Tracker. Table 4 shows the effect of diverse point trackers on dynamic labeling quality. From this experiment, we draw the following insights. First, the points misclassified by the LiDAR-only method are consistently refined by visual motion cues, regardless of the choice of the point tracking model. Specifically, compared with DUFOMap, TRACKCUE improves precision by 5.61–10.68 percentage points and F1 score by 4.94–5.63 points. Second, we find that TAPNext yields the highest precision but extremely lower recall under the same dynamic labeling criterion, indicating that its trajectories provide reliable yet less comprehensive motion evidence. As a result, TRACKCUE with TAPNext drops more true dynamic points, which can incorrectly encourage zero-flow estimation and lead to lower scene flow performance. These observations also imply that a high-fidelity label is crucial for each self-supervised loss term to provide appropriate supervision.

Image Resolution Sensitivity. Table 5 reports dynamic clustering quality under different image resolutions. Experiments are done using CoTracker3 [39] as a point tracker. High-resolution images preserve finer visual details and provide more reliable point trajectories, allowing more true dynamic points to be retained as dynamic. In contrast, low-resolution images introduce larger projection errors and tracking ambiguities when 3D points are projected onto the 2D image plane. As a result, some true dynamic points are missed, leading to lower recall. Consistent with the previous results, imbalanced dynamic classification introduces misleading supervision for object motion status.

Effect of Video Length. Table 6 reports the classification quality and downstream scene flow estimation performance. We observe that short clips provide insufficient trajectory evidence, causing more true dynamic points to be missed and resulting in lower recall. As the clip length increases, TRACKCUE predicts more points as dynamic, leading to a higher predicted dynamic ratio. Increasing the clip length initially improves recall by providing more temporal evidence, while overly long clips introduce tracking errors that offset this benefit. However, longer clips can accumulate tracking drifts, occlusion noise, and viewpoint changes, introducing additional false-positive dynamic labels and reducing precision. Therefore, since a clip length of 6 provides a favorable precision–recall trade-off, we use it as the default setting in our main experiments.

Analysis of Trajectory Filtering. Point trackers already predict a per-frame visibility mask, but Table 7 shows that relying on it alone is insufficient. This setting yields 24.98% precision and 36.96 F1 score, leaving the DUFOMap labels it starts from essentially unchanged. Point tracking by itself therefore provides little benefit without the proposed filtering. To improve the reliability of the motion evidence, we introduce temporal voting and joint visibility filtering. Temporal voting yields an F1 score of 42.28, raising precision to 31.28% while retaining a recall of 65.21%, since transient correspondence errors rarely persist across multiple frames. Further adding the joint visibility mask restricts residual computation to frames where both the tracked and rigid trajectories are valid, yielding 31.38% precision and an F1 score of 42.37. It also reduces the Dynamic Bucket error from 0.359 to 0.352. These results show that the reliability of the visual motion cues comes from the proposed filtering rather than from the point tracker itself.

Table 7: Effect of trajectory filtering. All settings use the ego-compensated residual, to which the joint visibility mask and temporal voting are added progressively. Results are obtained on the Argoverse 2 validation set [46], where the ground-truth dynamic ratio is 7.12%.

CoTracker3 [39] Mask	Temporal Voting	Joint Visibility Mask	Precision (%) ↑	Recall (%) ↑	F1 ↑	Dyn. Bucket ↓
✓			24.98	**71.02**	36.96	0.368
✓	✓		31.28	65.21	42.28	0.359
✓	✓	✓	**31.38**	65.22	**42.37**	**0.352**

Trajectory Residual Analysis. TRACKCUE identifies dynamic queries by comparing tracked trajectories with those expected under ego-motion alone. The residual in Eq. 7 captures motion that cannot be explained by ego-motion, and a query is classified as moving only when sufficiently large residuals persist over multiple jointly valid frames. To evaluate the discriminative ability of this residual independently of a particular threshold choice, we construct an ROC curve. For each trajectory, we use the second-largest residual over jointly valid frames as the trajectory-level motion score, which is equivalent to the $N_{\text{move}} = 2$ criterion in Eq. 9. We vary τ_{dyn} over the range of trajectory-level scores to trace the ROC curve shown in Figure 5. The resulting AUROC of 0.7474, well above the chance level of 0.5, indicates that the trajectory residual provides meaningful discrimination between dynamic and static candidates independently of a particular choice of τ_{dyn}. This supports the interpretation that the precision gain is guided by discriminative motion evidence rather than an arbitrary reduction in the predicted dynamic ratio. The curve rises steeply in the low false-positive regime, while the gain in the true-positive rate diminishes as the threshold is relaxed. This suggests that the trajectory residual is particularly effective for conservative dynamic detection, whereas further relaxing the threshold yields diminishing returns in recall.

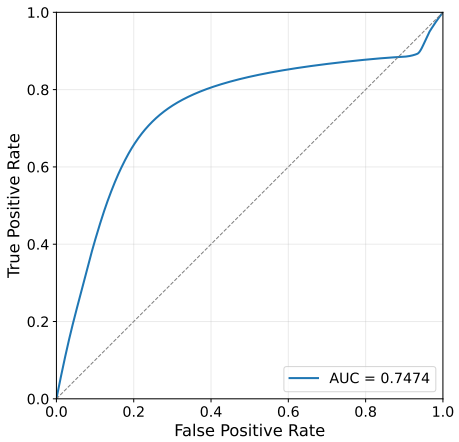


Figure 5: ROC curve of the trajectory motion score. Sweeping τ_{dyn} over the second-largest valid residual yields an AUROC of 0.7474.

5 Conclusion

We introduce TRACKCUE, a fully automated pipeline for refining static-dynamic pseudo-labels in self-supervised LiDAR scene flow estimation. We find that LiDAR-only dynamic-awareness mapping produces unreliable labels under sparsity and occlusion. To address this limitation, TRACKCUE complements sparse LiDAR geometry with rich visual motion cues while compensating for ego-induced image motion and filtering unreliable trajectories. Our experiments show that the resulting labels substantially improve dynamic-label precision and F1 score, leading to consistent gains in scene flow estimation across multiple self-supervised baselines. These results demonstrate the potential of image-based motion cues as complementary supervision for self-supervised LiDAR scene flow.

Limitations & Future Work. While TRACKCUE shows promising results, several limitations raise important research questions. First, tracking over 3D-to-2D flattened points is inherently limited, since projection inevitably loses 3D geometric information. As a result, small or distant objects may be missed or overly refined. Thus, object masks from foundation segmentation models such as SAM [51] could provide additional spatial constraints, preventing small or far-range dynamic objects from being overly suppressed. Second, the scope of our work is limited to generating high-fidelity pseudo-labels for clear self-supervised signals. Future work could further explore richer cross-modal information exchange, where RGB and LiDAR cues mutually refine both supervision signals and feature representations for 3D motion learning.

Acknowledgment

This research was supported by Autonomous Driving Development Center, Hyundai Motor Company R&D Division (85%), Culture, Sports and Tourism R&D Program through the Korea Creative Content Agency grant funded by the Ministry of Culture, Sports and Tourism (International Collaborative Research and Global Talent Development for the Development of Copyright Management and Protection Technologies for Generative AI, RS-2024-00345025, 10%), the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (RS-2025-00521602, 1%), Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No. RS-2019-II190079, Artificial Intelligence Graduate School Program(Korea University), 1%; IITP-2025-RS-2024-00436857, 1%; IITP-2026-RS-2026-25615490, AI Star Fellowship Support Program, 1%), and Advanced GPU Utilization Support Program funded by the Government of the Republic of Korea (Ministry of Science and ICT, 1%).

References

[1] Xueqian Li, Jhony Kaesemodel Pontes, and Simon Lucey. Neural scene flow prior. *Advances in Neural Information Processing Systems*, 34:7838–7851, 2021.

[2] Xueqian Li, Jianqiao Zheng, Francesco Ferroni, Jhony Kaesemodel Pontes, and Simon Lucey. Fast neural scene flow. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9878–9890, 2023.

[3] Siyi Li, Qingwen Zhang, Ishan Khatri, Kyle Vedder, Deva Ramanan, and Neehar Peri. Uniflow: Towards zero-shot lidar scene flow for autonomous vehicles via cross-domain generalization, 2025.

[4] Qingwen Zhang, Xiaomeng Zhu, Yushan Zhang, Yixi Cai, Olov Andersson, and Patric Jensfelt. Deltaflow: An efficient multi-frame scene flow estimation method. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026.

[5] Kyle Vedder, Neehar Peri, Ishan Khatri, Siyi Li, Eric Eaton, Mehmet Kemal Kocamaz, Yue Wang, Zhiding Yu, Deva Ramanan, and Joachim Pehserl. Neural eulerian scene flow fields. 2025.

[6] David T Hoffmann, Syed Haseeb Raza, Hanqiu Jiang, Denis Tananaev, Steffen Klingenhoefer, and Martin Meinke. Floxels: Fast unsupervised voxel based scene flow estimation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 22328–22337, 2025.

[7] Qingwen Zhang, Ajinkya Khoche, Yi Yang, Li Ling, Sina Sharif Mansouri, Olov Andersson, and Patric Jensfelt. HiMo: High-speed objects motion compensation in point cloud. *IEEE Transactions on Robotics*, 41:5896–5911, 2025.

[8] Qingwen Zhang, Yi Yang, Peizheng Li, Olov Andersson, and Patric Jensfelt. SeFlow: A self-supervised scene flow method in autonomous driving. In *European Conference on Computer Vision (ECCV)*, page 353–369. Springer, 2024.

[9] Yancong Lin, Shiming Wang, Liangliang Nan, Julian Kooij, and Holger Caesar. Voteflow: Enforcing local rigidity in self-supervised scene flow. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 17155–17164, 2025.

[10] Kyle Vedder, Neehar Peri, Nathaniel Chodosh, Ishan Khatri, Eric Eaton, Dinesh Jayaraman, Yang Liu, Deva Ramanan, and James Hays. Zeroflow: Scalable scene flow via distillation. In *International Conference on Learning Representations*, volume 2024, pages 3131–3143, 2024.

[11] Daniel Duberg, Qingwen Zhang, Mingkai Jia, and Patric Jensfelt. Dufomap: Efficient dynamic awareness mapping. *IEEE Robotics and Automation Letters*, 9(6):5038–5045, 2024.

[12] Philipp Jund, Chris Sweeney, Nichola Abdo, Zhifeng Chen, and Jonathon Shlens. Scalable scene flow from point clouds in the real world. *IEEE Robotics and Automation Letters*, 7(2):1589–1596, 2021.

[13] Ishan Khatri, Kyle Vedder, Neehar Peri, Deva Ramanan, and James Hays. I can't believe it's not scene flow! In *European Conference on Computer Vision*, pages 242–257. Springer, 2024.

[14] Ajinkya Khoche, Qingwen Zhang, Laura Pereira Sánchez, Aron Asefaw, Sina Sharif Mansouri, and Patric Jensfelt. Ssf: Sparse long-range scene flow for autonomous driving. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6394–6400. IEEE, 2025.

[15] Jaeyeul Kim, Jungwan Woo, Ukcheol Shin, Jean Oh, and Sunghoon Im. Flow4d: Leveraging 4d voxel network for lidar scene flow estimation. *IEEE Robotics and Automation Letters*, 2025.

[16] Qingwen Zhang, Chenhan Jiang, Xiaomeng Zhu, Yunqi Miao, Yushan Zhang, Olov Andersson, and Patric Jensfelt. TeFlow: Enabling multi-frame supervision for self-supervised feed-forward scene flow estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2026.

[17] Yancong Lin and Holger Caesar. ICP-Flow: Lidar scene flow estimation with icp. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15501–15511, June 2024.

[18] Zhijian Liu, Haotian Tang, Alexander Amini, Xinyu Yang, Huizi Mao, Daniela L Rus, and Song Han. Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation. In *2023 IEEE international conference on robotics and automation (ICRA)*, pages 2774–2781. IEEE, 2023.

[19] Guowen Zhang, Chenhang He, Liyi Chen, and Lei Zhang. Bevdilation: Lidar-centric multi-modal fusion for 3d object detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 12448–12456, 2026.

[20] Haiyang Wang, Hao Tang, Shaoshuai Shi, Aoxue Li, Zhenguo Li, Bernt Schiele, and Liwei Wang. Unitr: A unified and efficient multi-modal transformer for bird's-eye-view representation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6792–6802, 2023.

[21] Junjie Huang, Yun Ye, Zhujin Liang, Yi Shan, and Dalong Du. Detecting as labeling: Rethinking lidar-camera fusion in 3d object detection. In *European Conference on Computer Vision*, pages 439–455. Springer, 2024.

[22] Gyusam Chang, Wonseok Roh, Sujin Jang, Dongwook Lee, Daehyun Ji, Gyeongrok Oh, Jinsun Park, Jinkyu Kim, and Sangpil Kim. Cmda: Cross-modal and domain adversarial adaptation for lidar-based 3d object detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 972–980, 2024.

[23] Xu Zheng, Yuanhuiyi Lyu, and Lin Wang. Learning modality-agnostic representation for semantic segmentation from any modalities. In *European Conference on Computer Vision*, pages 146–165. Springer, 2024.

[24] Jiaming Zhang, Ruiping Liu, Hao Shi, Kailun Yang, Simon Reiß, Kunyu Peng, Haodong Fu, Kaiwei Wang, and Rainer Stiefelhagen. Delivering arbitrary-modal semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1136–1147, 2023.

[25] Jingyi Pan, Zipeng Wang, and Lin Wang. Co-occ: Coupling explicit feature fusion with volume rendering regularization for multi-modal 3d semantic occupancy prediction. *IEEE Robotics and Automation Letters*, 9(6):5687–5694, 2024.

[26] Xiaofeng Wang, Zheng Zhu, Wenbo Xu, Yunpeng Zhang, Yi Wei, Xu Chi, Yun Ye, Dalong Du, Jiwen Lu, and Xingang Wang. Openoccupancy: A large scale benchmark for surrounding semantic occupancy perception. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17850–17859, 2023.

[27] Corentin Sautier, Gilles Puy, Spyros Gidaris, Alexandre Boulch, Andrei Bursuc, and Renaud Marlet. Image-to-lidar self-supervised distillation for autonomous driving data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9891–9901, 2022.

[28] Xiang Xu, Lingdong Kong, Song Wang, Chuanwei Zhou, and Qingshan Liu. Beyond one shot, beyond one perspective: Cross-view and long-horizon distillation for better lidar representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 25506–25518, October 2025.

[29] Youquan Liu, Lingdong Kong, Jun Cen, Runnan Chen, Wenwei Zhang, Liang Pan, Kai Chen, and Ziwei Liu. Segment any point cloud sequences by distilling vision foundation models. *Advances in Neural Information Processing Systems*, 36:37193–37229, 2023.

[30] Haisong Liu, Tao Lu, Yihui Xu, Jia Liu, Wenjie Li, and Lijun Chen. Camliflow: bidirectional camera-lidar fusion for joint optical flow and scene flow estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5791–5801, 2022.

[31] Adam W Harley, Zhaoyuan Fang, and Katerina Fragkiadaki. Particle video revisited: Tracking through occlusions using point trajectories. In *European Conference on Computer Vision*, pages 59–75. Springer, 2022.

[32] Carl Doersch, Ankush Gupta, Larisa Markeeva, Adria Recasens, Lucas Smaira, Yusuf Aytar, Joao Carreira, Andrew Zisserman, and Yi Yang. Tap-vid: A benchmark for tracking any point in a video. *Advances in Neural Information Processing Systems*, 35:13610–13626, 2022.

[33] 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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10061–10072, 2023.

[34] Mel Vecerik, Carl Doersch, Yi Yang, Todor Davchev, Yusuf Aytar, Guangyao Zhou, Raia Hadsell, Lourdes Agapito, and Jon Scholz. RoboTAP: Tracking arbitrary points for few-shot visual imitation. *International Conference on Robotics and Automation*, pages 5397–5403, 2024.

[35] Carl Doersch, Pauline Luc, Yi Yang, Dilara Gokay, Skanda Koppula, Ankush Gupta, Joseph Heyward, Ignacio Rocco, Ross Goroshin, João Carreira, and Andrew Zisserman. BootsTAP: Bootstrapped training for tracking-any-point. *Asian Conference on Computer Vision*, 2024.

[36] 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. *Advances in Neural Information Processing Systems*, 2024.

[37] Artem Zholus, Carl Doersch, Yi Yang, Skanda Koppula, Viorica Patraucean, Xu Owen He, Ignacio Rocco, Mehdi S. M. Sajjadi, Sarath Chandar, and Ross Goroshin. Tapnext: Tracking any point (tap) as next token prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9693–9703, October 2025.

[38] 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*, pages 18–35. Springer, 2024.

[39] Nikita Karaev, Yuri Makarov, Jianyuan Wang, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker3: Simpler and better point tracking by pseudo-labelling real videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6013–6022, 2025.

[40] Adam W Harley, Yang You, Xinglong Sun, Yang Zheng, Nikhil Raghuraman, Yunqi Gu, Sheldon Liang, Wen-Hsuan Chu, Achal Dave, Suya You, et al. Alltracker: Efficient dense point tracking at high resolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5253–5262, 2025.

[41] Seokju Cho, Jiahui Huang, Jisu Nam, Honggyu An, Seungryong Kim, and Joon-Young Lee. Local all-pair correspondence for point tracking. In *European conference on computer vision*, pages 306–325. Springer, 2024.

[42] Jianyuan Wang, Nikita Karaev, Christian Rupprecht, and David Novotny. Vggsfm: Visual geometry grounded deep structure from motion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 21686–21697, 2024.

[43] Guillaume Le Moing, Jean Ponce, and Cordelia Schmid. Dense optical tracking: Connecting the dots. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19187–19197, 2024.

[44] Nan Huang, Wenzhao Zheng, Chenfeng Xu, Kurt Keutzer, Shanghang Zhang, Angjoo Kanazawa, and Qianqian Wang. Segment any motion in videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3406–3416, 2025.

[45] Frano Rajič, Lei Ke, Yu-Wing Tai, Chi-Keung Tang, Martin Danelljan, and Fisher Yu. Segment anything meets point tracking. In *Proceedings of the Winter Conference on Applications of Computer Vision*, pages 9284–9293, 2025.

[46] Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal, Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemodel Pontes, Deva Ramanan, Peter Carr, and James Hays. Argoverse 2: Next generation datasets for self-driving perception and forecasting. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks (NeurIPS Datasets and Benchmarks 2021)*, 2021.

[47] 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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2446–2454, 2020.

[48] 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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020.

[49] Felix Fent, Fabian Kuttenreich, Florian Ruch, Farija Rizwin, Stefan Juergens, Lorenz Lechermann, Christian Nissler, Andrea Perl, Ulrich Voll, Min Yan, et al. Man truckscenes: A multimodal dataset for autonomous trucking in diverse conditions. *Advances in Neural Information Processing Systems*, 37:62062–62082, 2024.

[50] Gautham Narayan Narasimhan, Heethesh Vhavle, Kumar Bhargav Vishvanatha, and James Reuther. Aevascenes: A dataset and benchmark for fmcw lidar perception, 2025.

[51] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4015–4026, 2023.

[52] Qingwen Zhang, Yi Yang, Heng Fang, Ruoyu Geng, and Patric Jensfelt. Deflow: Decoder of scene flow network in autonomous driving. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2105–2111. IEEE, 2024.

[53] Ricardo JGB Campello, Davoud Moulavi, and Jörg Sander. Density-based clustering based on hierarchical density estimates. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 160–172. Springer, 2013.

[54] 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, pages 37738–37767, 2026.

Appendix

A Experiment settings

A.1 Dataset Details

In this section, we provide additional details on the Argoverse 2 leaderboard [3] test set, which comprises five different source datasets: Argoverse 2 [46], Waymo [47], nuScenes [48], TruckScenes [49], and AEVAScenes [50]. To ensure a fair evaluation, the source identity of each scene is hidden by replacing dataset-specific information with anonymized random identifiers.

Argoverse 2 is a widely adopted benchmark for LiDAR scene flow estimation. It provides annotated sensor data from 1,000 scenes, split into 700 training, 150 validation, and 150 test scenes. Each frame is captured with seven ring cameras, two stereo cameras, and a point cloud acquired by two 32-beam LiDAR sensors. The dataset primarily covers urban driving scenarios in U.S. cities.

Waymo is another large-scale benchmark in 3D perception for automative vehicles. It consists of 1,150 scenes with multimodal sensor data, including images from five high-resolution surround-view cameras and point clouds from five 64-beam LiDAR sensors. The dataset spans diverse driving conditions, ranging from urban and suburban environments to challenging nighttime scenarios.

nuScenes also contains 1,000 fully annotated scenes for various downstream 3D perception tasks and end-to-end autonomous driving. It captures urban driving scenarios using six surround-view cameras, a 32-beam LiDAR sensor, and five radar sensors.

TruckScenes contains geometric distribution shifts by collecting driving data from trucks, which differ from vehicles in both sensor placement and platform geometry. It provides over 740 driving scenes, each lasting 20 seconds, across a broad range of environmental conditions. Each scene primarily captures high-speed highway scenarios using four pinhole cameras, six 64-beam LiDAR sensors, and six radar sensors.

AEVAScenes provides high-fidelity LiDAR data with per-point measurements and long-range sensing capabilities for 3D scene understanding. The dataset contains 10,000 frames captured by six wide- and narrow-FOV cameras, along with FMCW LiDAR sensors that cover front and rear views up to 200 m. It includes diverse driving conditions across cities and highways, covering both daytime and nighttime scenes in the Bay Area.

A.2 Implementation Details

In this work, we focus on self-supervised LiDAR scene flow estimation. We choose three strong baselines: SeFlow (ECCV'24) [8], SeFlow++ (T-RO'25) [7], and TeFlow (CVPR'26) [16]. Note that we use the official OpenSceneFlow repository[1] to reproduce all experimental results. Table 8 summarizes the experimental configurations of the three baselines. Scene flow models with TRACKCUE labels are trained on four NVIDIA B200 GPUs, and all methods are evaluated on a single NVIDIA RTX A6000 GPU. The point tracker is used off-the-shelf and remains frozen throughout the pipeline.

Table 8: Training configurations for self-supervised scene flow baselines. All configurations follow the official OpenSceneFlow implementation.

Category	Specification	SeFlow [8]	SeFlow++ [7]	TeFlow [16]
Model	Backbone	DeFlow [52]	DeFlow [52]	DeltaFlow [4]
	Input frames	2	3	5
	Voxel size	$[0.2, 0.2, 6]$	$[0.2, 0.2, 6]$	$[0.15, 0.15, 0.15]$
	Point cloud range	$[-51.2, -51.2, -3, 51.2, 51.2, 3]$	$[-51.2, -51.2, -3, 51.2, 51.2, 3]$	$[-38.4, -38.4, -3, 38.4, 38.4, 3]$
Training	Auto-labeler for SSL	SeFlow auto-labeler	SeFlow++ auto-labeler	SeFlow auto-labeler
	Optimizer	AdamW	AdamW	Adam
	Learning rate	2×10^{-4}	2×10^{-4}	2×10^{-3}
	Scheduler	–	StepLR	StepLR
	Epochs	9	9	15
	Batch size	16	4	2

[1] https://github.com/KTH-RPL/OpenSceneFlow

B Method Details

B.1 Dynamic-awareness Mapping

To obtain self-supervised signals without manual annotations, existing methods commonly rely on occupancy-based dynamic-awareness mapping. DUFOMap [11] uses voxel grids and ray-casting to identify dynamic points based on temporal occupancy consistency. It marks ray-traversed voxels as void regions and classifies points later observed in those regions as dynamic. Formally, let $\mathcal{V}_{<t}^{\text{void}}$ denote the set of voxels observed as void before time t. A point $\mathbf{p}_{i,t} \in \mathcal{P}_t$ is classified as dynamic if its voxel index $\text{v}(\mathbf{p}_{i,t})$ conflicts with the previous void region:

$$m_i = \mathbb{I}\left[\text{v}(\mathbf{p}_{i,t}) \in \mathcal{V}_{<t}^{\text{void}}\right], \tag{13}$$

where $m_i = 1$ indicates a dynamic point and $m_i = 0$ indicates a static point. Then, the DUFOMap-based dynamic point set is defined as $\mathcal{P}_t^{\text{dufo}} = \{\mathbf{p}_{i,t} \mid m_i = 1\}$. Although DUFOMap provides static-dynamic labels without manual annotations, it is sensitive to LiDAR sparsity and occlusions, which introduce noisy static-dynamic labels.

B.2 Auto-labeling in self-supervised learning

SeFlow [8] auto-labeler. SeFlow first proposes an auto-labeling strategy in self-supervised methods that use the DUFOMap-based dynamic point set $\mathcal{P}^{\text{dufo}}$ to construct self-supervision signals. While static points are constrained to have zero motion, dynamic points are clustered and optimized using geometric consistency objectives. To improve the accuracy of these dynamic labels, a cluster-level reassignment strategy is employed. Here, $\mathcal{P}^{\text{c}}$ is obtained by applying HDBSCAN [53] only to dynamic points, with the minimum cluster size set to 20 and the cluster selection epsilon set to 0.7. Ultimately, $\mathcal{P}^{\text{c}}$ is aligned with $\mathcal{P}^{\text{dufo}}$ by masking out non-DUFO dynamic points.

SeFlow++ [7] auto-labeler. This strategy incorporates a nearest-neighbor dynamic point set $\mathcal{P}^{\text{nnd}}$ into the SeFlow auto-labeler. Specifically, a point $p \in \mathcal{P}_t$ is assigned to $\mathcal{P}^{\text{nnd}}$ if its nearest-neighbor distance to $\mathcal{P}_{t+1}$ exceeds a user-defined threshold τ_d:

$$\mathcal{P}^{\text{nnd}} = \{p \mid D_{\min}(p, \mathcal{P}_{t+1}) > \tau_d,\ p \in \mathcal{P}_t\},$$

where $D_{\min}$ computes the distance between a selected point p and its nearest-neighbor counterpart. For Argoverse 2, they set the nearest-neighbor distance threshold to $\tau_d = 0.14\,\text{m}$. Given a set of clusters $\mathcal{C}$, each cluster $C \in \mathcal{C}$ is classified as dynamic based on the ratio of dynamic points:

$$f(C) = \begin{cases} \text{Dynamic}, & \text{if } \min(r_1, r_2) \geq \tau_1 \wedge \max(r_1, r_2) \geq \tau_2, \\ \text{Static}, & \text{otherwise}, \end{cases} \tag{14}$$

where $r_1 = \frac{|\mathcal{P}^{\text{dufo}} \cap C|}{|C|}$, $r_2 = \frac{|\mathcal{P}^{\text{nnd}} \cap C|}{|C|}$. Following the given configuration, we set $\tau_1 = 0.05$ and $\tau_2 = 0.30$ for Argoverse 2. Although this cluster-level reassignment improves robustness, the auto-labeling process still remains entirely dependent on LiDAR geometry. Consequently, the resulting labels are still vulnerable to errors caused by sparsity and occlusions. Thus, starting from the DUFOMap-based dynamic point set $\mathcal{P}^{\text{dufo}}$, our proposed TRACKCUE pipeline produces the refined dynamic point set $\mathcal{P}^{\text{TrackCue}}$. For all experiments, we replace $\mathcal{P}^{\text{dufo}}$ with $\mathcal{P}^{\text{TrackCue}}$ when generating pseudo-labels for each auto-labeling strategy.

B.3 TRACKCUE Details

This section summarizes the proposed TRACKCUE pipeline and presents the hyperparameter settings used throughout our experiments. After generating $\mathcal{P}^{\text{TrackCue}}$, we train the three baseline models following the official training setup (see Table 8). In our main experiments, we adopt CoTracker3 as a point tracker. Seven surround-view RGB images are resized to 768×768, and a 6-frame video clip is used as input. Furthermore, we project up to 2,048 points from $\mathcal{P}^{\text{dufo}}$ onto the image plane as initial dynamic point queries. For the TRACKCUE pipeline, we use fixed thresholds throughout the entire process. We set the moving criterion τ_{dyn} to 5.0 pixels, require dynamic evidence over at least 2 frames for temporal voting, and use a minimum of 10 points for moving retention. Then, we lift the selected visual motion cues back to the LiDAR point cloud. We use the top-K_{nn} nearest projected points with $K_{\text{nn}} = 4$ and a lifting radius $\tau_{\text{lift}} = 0.4$. To enhance clarity and facilitate understanding, we provide a pseudo-code of TRACKCUE pipeline in Algorithm 1.

Algorithm 1 PyTorch-style Pseudocode for TRACKCUE

```
# input:
# P, I: LiDAR point clouds and RGB images
# G: ego poses
# D: DUFOMap dynamic labels (P^dufo)
# F: Point tracker (e.g., CoTracker3)

# Hyperparameters:
# τ_dyn: motion threshold
# N_move: minimum moving frames
# N_point: minimum number of dynamic points
# τ_lift: lifting radius
# K_nn: top-K_nn nearest projected points for lifting

# output:
# Y: TRACKCUE dynamic labels (P^TRACKCUE)

def trackcue(P, I, G, D, F):
    # initialize refined dynamic labels
    Y = init_dynamic_masks(P)

    # make short clip for tracking
    for clip in split_into_clips(P, I, G, D):
        for v in camera_views:
            # query LiDAR points from DUFOMap dynamic labels
            Q = select_lidar_queries(
                clip.P[0], clip.D[0], camera_view=v
            )

            # Image-based Point Tracking (Sec. 3.3)
            q = lidar_to_image_projection(Q.xyz, camera_view=v)
            z_img, vis = F.track(clip.I[:, v], q)

            # Visually Consistent Motion Compensation (Sec. 3.4)
            z_rigid, rigid_vis = rigid_reproject(
                Q.xyz, clip.G, camera_view=v
            )

            e = torch.norm(z_img - z_rigid, dim=-1)
            votes = ((e > tau_dyn) & vis & rigid_vis).sum(dim=1)
            dynamic = votes >= N_move
            dynamic = dynamic & (dynamic.sum() >= N_point)

            # Visual Motion Cue Lifting (Sec. 3.5)
            Y = assign_tracks_to_lidar(
                Y, z_img[dynamic], clip.P, camera_view=v,
                radius=tau_lift, top_k=K_nn
            )

    return Y
```

C Additional Experimental Results

C.1 Quantitative Results.

We further provide additional quantitative results on the leaderboard, including detailed class-wise scores and Three-way End-Point Error across the Argoverse 2 data split. As shown in Table 9, TRACKCUE consistently yields overall improvements in terms of object classes. This implies that the proposed method effectively identifies not only vehicles but also small and slow pedestrians, ultimately strengthening the network's generalization capability through highly refined supervision.

Table 9: Experimental results on the Argoverse 2 leaderboard [3] – **AV2** [46]

Method	#Frames	Three-way EPE (cm) ↓				Dynamic Bucket-Normalized ↓				
		Mean	FD	FS	BS	Mean	CAR	OTHER	PED.	VRU
SeFlow [8]	2	4.76	**10.93**	2.23	1.12	0.3113	0.2365	**0.2949**	0.4406	0.2733
+ TRACKCUE	2	**4.66** (↓0.10)	10.97 (↑0.04)	**2.00** (↓0.23)	**1.02** (↓0.10)	**0.3070** (↓0.0043)	**0.2313** (↓0.0052)	0.3198 (↑0.0249)	**0.4073** (↓0.0333)	**0.2695** (↓0.0038)
SeFlow++ [7]	3	**4.20**	9.49	**2.02**	**1.08**	0.2827	0.2023	0.3075	**0.3867**	0.2343
+ TRACKCUE	3	4.22 (↑0.02)	**9.14** (↓0.35)	2.35 (↑0.33)	1.16 (↑0.08)	**0.2781** (↓0.0046)	**0.2009** (↓0.0014)	**0.2865** (↓0.0210)	0.3959 (↑0.0092)	**0.2291** (↓0.0052)
TeFlow [16]	5	3.54	7.02	2.18	1.42	0.2019	**0.1673**	0.2208	0.2734	**0.1461**
+ TRACKCUE	5	**3.41** (↓0.13)	**6.88** (↓0.14)	**1.98** (↓0.20)	**1.36** (↓0.06)	**0.1966** (↓0.0053)	0.1712 (↑0.0039)	**0.2165** (↓0.0043)	**0.2477** (↓0.0257)	0.1511 (↑0.0050)

Table 10: Comparison with monocular 4D reconstruction on the Argoverse 2 validation set [46]. Trace Anything [54] is evaluated with its officially released weights without training or fine-tuning on Argoverse 2, following the same protocol as TRACKCUE. Computational cost is measured per scene on a single RTX A6000 GPU. The ground-truth dynamic ratio is 7.12%.

Method	Pred. Dyn. Ratio (%)	Precision (%) ↑	Recall (%) ↑	F1 Score ↑	Dyn. Bucket ↓	Time (s) ↓	Mem. (GB) ↓
Trace Anything (GT pose)	15.66	27.90	61.37	38.37	0.428	406.1	11.24
Trace Anything (Est. pose)	19.49	13.02	35.63	19.07	0.587	759.6	11.24
TRACKCUE (Ours)	**14.80**	**31.38**	**65.22**	**42.37**	**0.352**	**56.6**	**3.45**

C.2 Comparison with Monocular 4D Reconstruction.

A natural alternative to TRACKCUE is to recover full 3D trajectories through monocular 4D reconstruction and derive dynamic regions from them. Table 10 compares against Trace Anything [54], where dynamic regions are obtained from the temporal variance of reconstructed trajectories and transferred to LiDAR points. With ground-truth camera poses, which eliminate pose estimation errors, Trace Anything still over-predicts dynamic points at a ratio of 15.66 compared to the ground-truth ratio of 7.12, and achieves 38.37 F1, compared with 42.37 for TRACKCUE. When poses are instead estimated from the predicted point maps, F1 drops to 19.07 as pose errors propagate through the reconstructed trajectories. In contrast, TRACKCUE uses vehicle odometry, which is independent of image-based pose estimation and already required by the self-supervised baselines for ego-motion decomposition in Eq. 1. Even with ground-truth poses, Trace Anything requires 7.2× more computation and 3.3× more memory than TRACKCUE. These results indicate that reliable static-dynamic labels do not follow automatically from the monocular 4D reconstruction and that the process is inefficient.

C.3 Predicted Flow Magnitude on Dynamic Points.

We further investigate the predicted residual flow $\Delta\mathcal{F}$ in Eq. 1, which remains after removing the ego-motion flow $\mathcal{F}^{\text{ego}}$ from the scene flow and is the component directly predicted by the model. We measure its magnitude on ground-truth dynamic points to examine whether improved pseudo-labels reduce static-like motion predictions. Ground-truth dynamic points are those whose residual motion exceeds $0.05\,\text{m}$ per frame, so a prediction below this value assigns static-like motion to a point that genuinely moves. As shown in Table 11, SeFlow [8] predicts less than $0.05\,\text{m}$ of residual motion for 39.5% of ground-truth dynamic points, whereas TRACKCUE reduces this fraction to 34.7%. The median predicted residual-flow magnitude also increases from $0.24\,\text{m}$ to $0.35\,\text{m}$. Together with the lower Dynamic Bucket error, these results show that the cleaner dynamic supervision provided by TRACKCUE reduces static-like predictions on truly moving points while improving dynamic scene-flow estimation.

Table 11: Distribution of predicted residual-flow magnitudes on ground-truth dynamic points and Dynamic Bucket-Normalized EPE on the Argoverse 2 validation set. The four bins report the fraction of ground-truth dynamic points grouped by the predicted residual-flow magnitude. The median is computed from the predicted residual-flow magnitudes of all ground-truth dynamic points. Following the official scene flow evaluation protocol [13], ground-truth dynamic points are defined using a $0.05\,\mathrm{m}$ residual-motion threshold. Accordingly, predictions below $0.05\,\mathrm{m}$ are considered static-like.

	Fraction of ground-truth dynamic points (%)				Median (m)	Dyn. Bucket ↓
Method	$< 0.05\,\mathrm{m}\downarrow$	0.05–0.2 m	0.2–0.5 m	$\geq 0.5\,\mathrm{m}$		
SeFlow [8]	39.5	8.7	14.6	37.2	0.24	0.372
+ TRACKCUE	**34.7**	8.3	16.0	41.0	0.35	**0.352**

C.4 Qualitative Results.

Comparison of Dynamic Clustering Quality. We visualize a dynamic awareness map by accumulating point-wise labels over a single scene. Red points denote non-stationary points. As clearly shown in Figure 6, TRACKCUE produces more consistent dynamic labels than DUFOMap [11]. Importantly, we observe superiority with improvements of up to 46.94 percentage points in precision and 20.43 points in F1 score.

Visualization of TRACKCUE Results. Beyond numerical metrics, Figures 7–8 further verify the effectiveness of TRACKCUE through step-by-step visualization results. As shown in the top panel, the initial query points obtained from DUFOMap [11] are sparse and vulnerable to noise. Then, we scatter these points into image space, which is occupied by moving or non-moving objects. Using the point tracker (e.g., CoTracker3 [39]), we acquire dense visual motion trajectories for candidate query points. After that, adjusting the rigid motion induced by the ego-vehicle successfully distinguishes static and dynamic objects. Specifically, it can capture pedestrian motions while rejecting temporarily stopped vehicles due to traffic conditions. These ego-compensated 2D trajectories are projected back to 3D points by visual motion cue lifting. These results show that temporally consistent visual trajectories provide dense motion evidence, enabling TRACKCUE to complete initially partial dynamic labels.

Additional Qualitative Results for Scene Flow Estimation. For this study, we show two complementary views with corresponding images. The visualization results in Figures 9—11 derive the following insights. The baseline models suffer from the zero-flow estimation problem, producing weak motion responses for dynamic objects. For instance, the light hue in the color-coded flow indicates an underestimated motion magnitude. However, TRACKCUE provides reliable dynamic guidance while avoiding the bias toward zero-flow predictions, enabling scene flow models to learn more accurate motion through sophisticated self-supervised signals. Second, TRACKCUE resolves the discontinuous labeling issue, where large dynamic objects, such as trucks, are partially classified as static. Such cases are difficult to handle using sparse geometric cues alone, since the central regions of large objects can remain locally occupied across consecutive frames. By leveraging dense visual trajectory cues, TRACKCUE propagates motion evidence across the object and produces more coherent dynamic labels.

D Broader Impact

LiDAR scene flow estimation is an important component for autonomous driving, as it provides point-level motion cues that can support perception, prediction, and planning. By improving self-supervised scene flow learning without requiring dense ground-truth flow annotations, TRACKCUE can reduce the dependence on costly manual labeling and make motion estimation methods easier to scale to large driving datasets. The proposed use of image point tracking also highlights the potential of combining complementary sensor modalities to improve dynamic object understanding in sparse LiDAR scenes. At the same time, the reliability of such systems remains critical. Since TRACKCUE uses image-based trajectories to refine dynamic labels, its performance may be affected by camera quality, illumination changes, adverse weather, calibration errors, and failures of the point tracker. Incorrect dynamic labels could propagate to scene flow learning and affect downstream autonomous driving modules. Therefore, additional validation under diverse driving conditions

and careful integration with safety-critical perception pipelines are necessary before deployment in real-world autonomous systems.

E Code base and License

Official URLs for the datasets and leaderboard utilized throughout our experiments (see Table 12).

Table 12: License and URLs for Datasets

Name	License	URL
AV2 2026 Scene Flow Challenge	Argoverse 2 Terms of Use	`https://www.argoverse.org/sceneflow.html`
OpenSceneFlow	BSD-3-Clause license	`https://github.com/KTH-RPL/OpenSceneFlow`

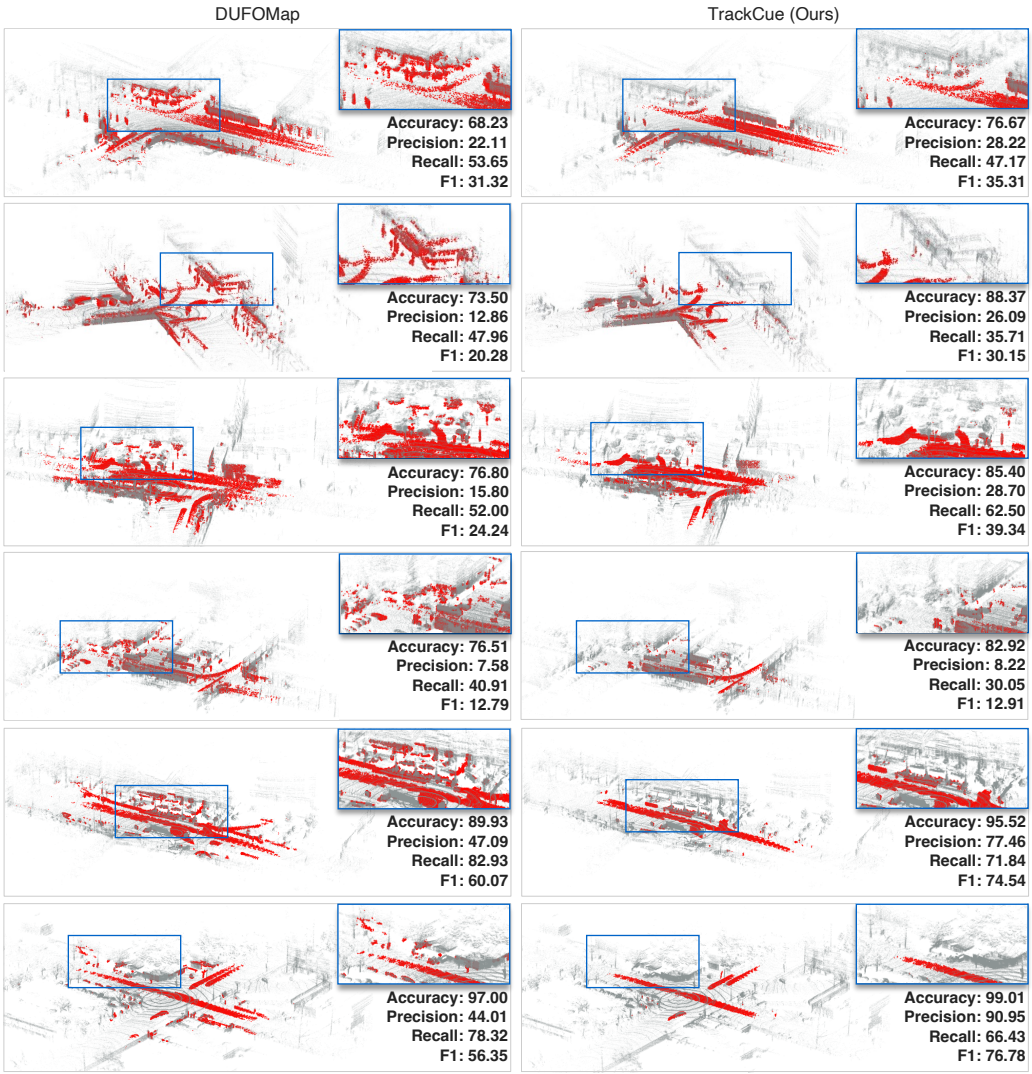


Figure 6: Qualitative comparison of dynamic awareness map between DUFOMap [11] and TRACKCUE.

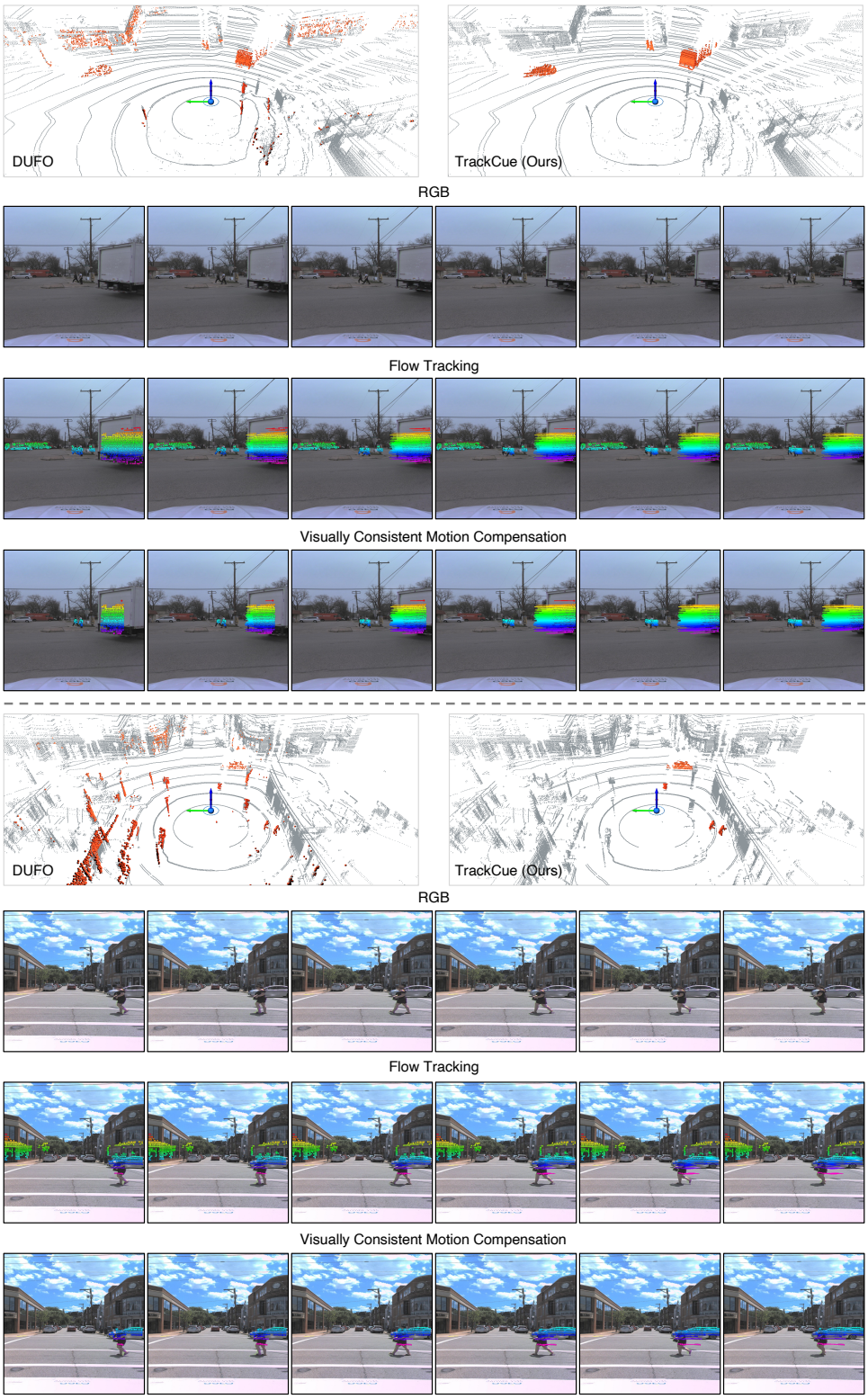


Figure 7: Step-by-step visualization of TRACKCUE.

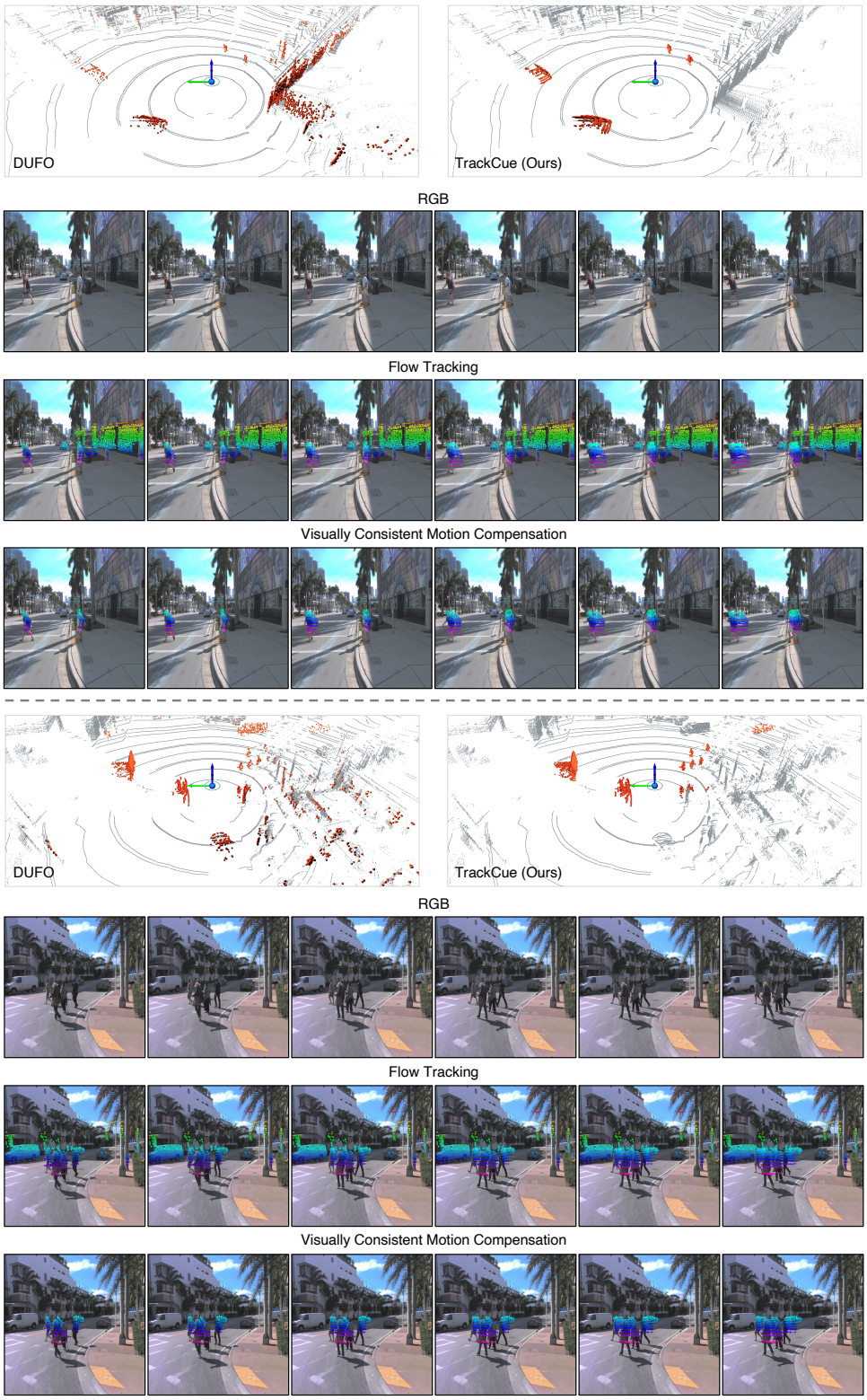


Figure 8: Step-by-step visualization of TRACKCUE.

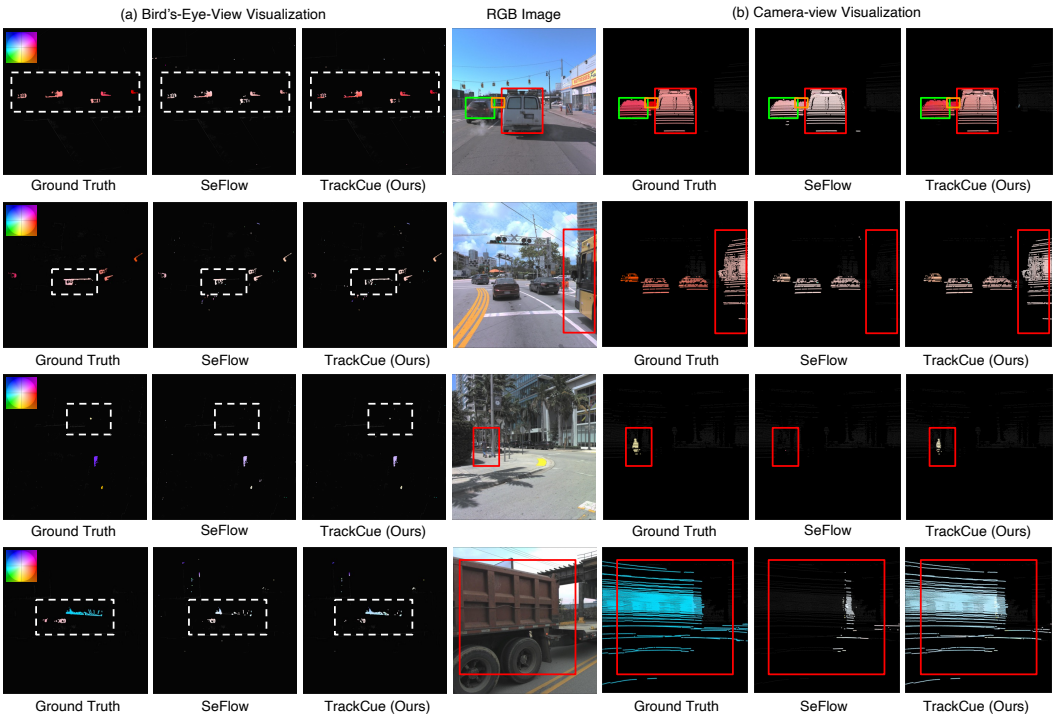


Figure 9: Qualitative comparison with SeFlow [8] for scene flow estimation.

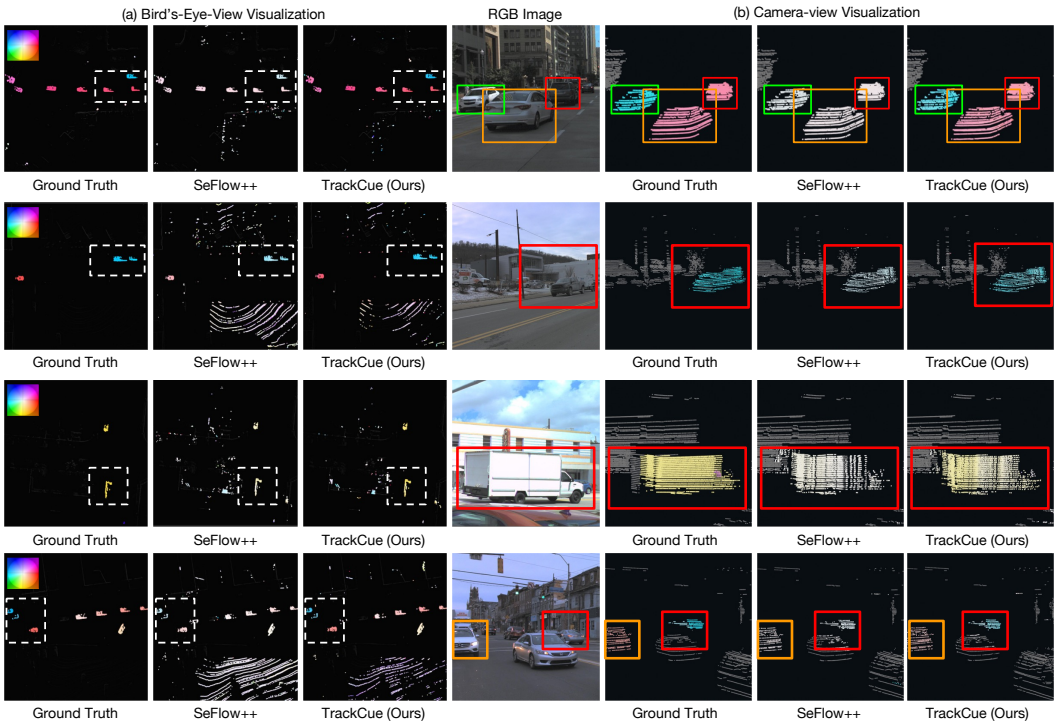


Figure 10: Qualitative comparison with SeFlow++ [7] for scene flow estimation.



Figure 11: Qualitative comparison with TeFlow [16] for scene flow estimation.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We have stated in both abstract and introduction that the main focus of our work is on generating high-fidelity dynamic labels via point tracking for scene flow estimation.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We have included the discussions of the limitation of the work, especially about the geometric information loss.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: The paper does not include theoretical results.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We have provided the details of the experimental settings including the hardware device, hyperparameter settings, and training duration.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: We plan to release source code upon acceptance.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Detailed descriptions of the experimental conditions can be found in the main manuscript as well as in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: The paper does not report error bars.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We have provided the computational resources used for our experiments in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: No potential ethical harms are expected in our research.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We have discussed in the appendix the potential of our research to improve the reliability of real-world deployment.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: No such risks are expected in our paper.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We have provided the licenses and credits for the benchmark datasets and code base used in our work.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve research with human subjects or crowdsourcing. Therefore, IRB approval was not required.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: Our work does not use LLMs as core components, nor as original or non-standard modules.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.