IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

HiMo: High-Speed Objects Motion Compensation in Point Clouds

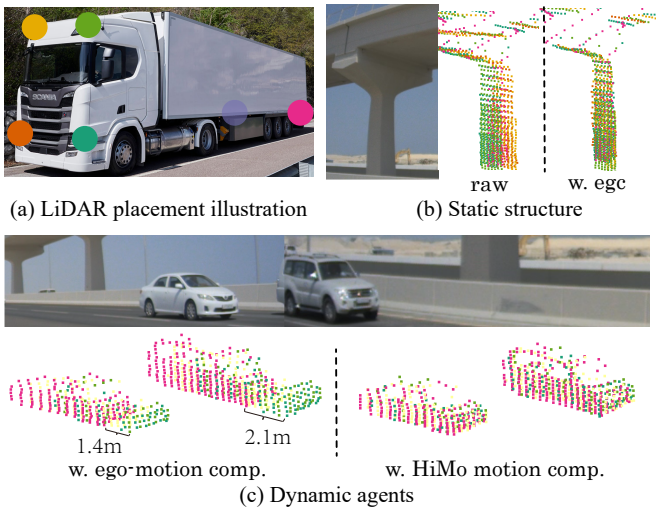
Qingwen Zhang, *Graduate Student Member, IEEE*, Ajinkya Khoche, *Member, IEEE*, Yi Yang, *Graduate Student Member, IEEE*, Li Ling, *Student Member, IEEE*, Sina Sharif Mansouri, Olov Andersson, *Member, IEEE*, Patric Jensfelt, *Member, IEEE*

***Abstract*—LiDAR point cloud is essential for autonomous vehicles, but motion distortions from dynamic objects degrade the data quality. While previous work has considered distortions caused by ego motion, distortions caused by other moving objects remain largely overlooked, leading to errors in object shape and position. This distortion is particularly pronounced in high-speed environments such as highways and in multi-LiDAR configurations, a common setup for heavy vehicles. To address this challenge, we introduce HiMo, a pipeline that repurposes scene flow estimation for non-ego motion compensation, correcting the representation of dynamic objects in point clouds. During the development of HiMo, we observed that existing self-supervised scene flow estimators often produce degenerate or inconsistent estimates under high-speed distortion. We further propose SeFlow++, a real-time scene flow estimator that achieves state-of-the-art performance on both scene flow and motion compensation. Since well-established motion distortion metrics are absent in the literature, we introduce two evaluation metrics: compensation accuracy at a point level and shape similarity of objects. We validate HiMo through extensive experiments on Argoverse 2, ZOD and a newly collected real-world dataset featuring highway driving and multi-LiDAR-equipped heavy vehicles. Our findings show that HiMo improves the geometric consistency and visual fidelity of dynamic objects in LiDAR point clouds, benefiting downstream tasks such as semantic segmentation and 3D detection. See https://kin-zhang.github.io/HiMo for more details.**

***Index Terms*—Range Sensing; Autonomous Driving Navigation; Computer Vision for Transportation; Motion Compensation**

I. Introduction

LIGHT Detection and Ranging (LiDAR) sensors are an integral part of perception systems for autonomous driving. These sensors provide detailed depth information and point cloud data, complementing other sensing modalities (such as cameras) to offer high-precision 3D scene understanding [1], [2], [3], [4], [5], [6], [7]. However, due to the rotating mechanism of mechanical LiDAR sensors, different parts of the environment are measured at different times. This introduces motion-induced point cloud distortion in dynamic scenes, making it difficult to obtain accurate environment representations. We refer to this as rolling shutter distortion [8] to highlight the close relation with the effect seen in the image domain [9], [10].



(a) LiDAR placement illustration (b) Static structure (c) Dynamic agents

Fig. 1: Multi-LiDARs are equipped in our heavy vehicles to avoid self-occlusion. (a) shows an example placement with 6 LiDARs. The point colors in (b-c) correspond to the LiDAR from which the points are captured. (b) illustrates the distortion of static structure due to fast-moving ego vehicle. *Raw* shows the raw data, *w. egc* shows the ego-motion compensation results. (c) demonstrates distortion caused by motion of other objects, which depends on the velocity of the said objects. In such case, ego-motion compensation alone (*w. ego-motion comp.*) is insufficient. In comparison, our HiMu pipeline (*w. HiMo motion comp.*) successfully undistorts the point clouds completely, resulting in an accurate representation of the objects.

There are two primary causes for LiDAR rolling shutter distortion: the motion of the ego vehicle and the motion of other agents in the scene. In the first case, the movement of the ego-vehicle, combined with the latency caused by the mechanical rotation of the LiDAR, leads to distorted representations of static objects or scenes. Such ego-motion induced distortion is well-studied in robotics [11]. In practice, localization algorithms [12], [13], [14] and additional global positioning devices can be employed to accurately correct this distortion (see Fig. 1 (b)).

The second source of error – motion of other agents, on the other hand, is underexplored. In this case, the motion distortion

Received 28 August 2024; revised 27 April 2025; accepted 19 August 2025. Date of publication 8 October 2025; date of current version 22 October 2025. This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation and Prosense (2020-02963) funded by Vinnova. This paper was recommended for publication by Associate Editor M. Walter and Editor S. Behnke upon evaluation of the reviewers' comments. *(Corresponding author: Qingwen Zhang.)*

Qingwen Zhang, Li Ling, Olov Andersson, and Patric Jensfelt are with the Division of Robotics, Perception, and Learning, KTH Royal Institute of Technology, Stockholm 114 28, Sweden. (email: qingwen@kth.se)

Ajinkya Khoche, Yi Yang are with the Division of Robotics, Perception, and Learning, KTH Royal Institute of Technology, Stockholm 114 28, Sweden, and also with Scania Group, Södertälje 151 87, Sweden.

Sina Sharif Mansouri is with Autonomous Transport Solutions Lab, Scania Group, Södertälje 151 87, Sweden.

Digital Object Identifier 10.1109/TRO.2025.3619042.

IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

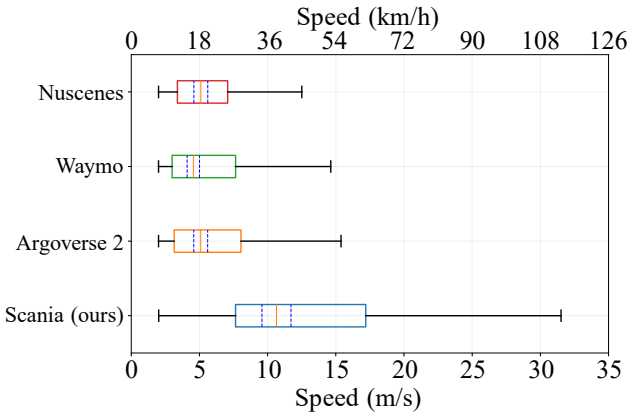


Fig. 2: Distribution of object speeds in different datasets. Only objects with speeds exceeding $2\mathrm{m/s}$ are included in the plot. The orange line shows the median speed and the blue dashed lines indicate the $\pm 10\%$ spread.

is object-specific and depends on the relative velocity between the dynamic objects and the ego-vehicle. An example of this is shown in Fig. 1 (c), where the ego-motion compensated point cloud representation of the gray car is more elongated compared to the white car due to its higher velocity. This example demonstrates that ego-motion compensation alone cannot effectively address the distortion caused by dynamic objects. This residual distortion has significant consequences: When sweeps of multiple LiDARs are combined, multiple copies of the same objects may appear in the merged point cloud, leading to incorrect object position or misleading representations that complicate downstream tasks.

However, to the best of our knowledge, the non-ego motion distortion caused by dynamic objects has not yet been reported in common public datasets. This is likely due to the low object speeds in these datasets. Existing open datasets in autonomous driving, such as KITTI [15], Argoverse [16], [17], Waymo [18] and Nuscenes [19], focus on urban environments, where the speeds of most dynamic objects are lower than 40 $\mathrm{km/h}$ (around 11.1 $\mathrm{m/s}$), as shown in Fig. 2. At such low speed, the distortion exists but is less pronounced (see Section III-A for details). Nonetheless, such distortions cannot be ignored when bringing autonomous driving solutions to real-world scenarios with large and fast-moving vehicles.

There are multiple research areas that tackle challenges in dynamic point clouds. Among them, tasks closely related to ours include moving object segmentation [20], [21], [22], which classifies points into dynamic and static categories, and scene flow estimation [23], [24], which estimates point-wise motion in successive LiDAR scans. However, these methods are primarily designed to analyze motion *after* the point cloud has been captured, and do not address **non-ego motion distortion** that occurs *during* the LiDAR scanning process. In this paper, we focus on this particular problem and propose **Hi**gh-speed object **Mo**tion compensation (HiMo), a pipeline for non-ego motion compensation for point clouds. An example of HiMo compensated results is shown in Fig. 1 (c). Our primary contributions are as follows:

- We provide an in-depth analysis of motion-induced distortions in autonomous driving datasets, and highlight that high-speed objects have larger distortions. To support this claim and facilitate further evaluation, we collect *Scania* - a heavy-vehicle highway multi-LiDAR driving dataset, featuring significantly higher average object speed than existing datasets (see Fig. 2).
- We propose HiMo, the first pipeline for non-ego motion compensation, which leverages self-supervised scene flow estimation to undistort point clouds, significantly improving object representation accuracy. We demonstrate that HiMo compensated point clouds achieve better performance in downstream tasks, including segmentation, 3D detection, and discuss its implications for planning.
- We additionally develop SeFlow++, a self-supervised scene flow method that enables efficient training via refined auto-labeling and a symmetric Chamfer loss, supporting effective motion compensation for HiMo under limited data and in high-speed conditions. It also achieves state-of-the-art scene flow performance among real-time self-supervised methods.
- We present two evaluation metrics for non-ego motion compensation and conduct extensive ablation studies with different scene flow estimators to demonstrate the modularity of HiMo and its robustness across datasets.

We provide our evaluation data and all codes at https://github.com/KTH-RPL/HiMo to promote the reproducibility and further development of our work.

II. Related Work

A. Motion Compensation

As mentioned previously, the motion-induced distortion can be decomposed into two components. The first occurs due to the motion of the ego vehicle. In robotics, particularly within the field of Simultaneous Localication and Mapping (SLAM), existing methods [12], [13] account for this through ego-motion compensation to a specific timestamp. This timestamp is typically chosen to be in the middle of the LiDAR scan, or in the middle of a scan window in case of multiple LiDARs. The ego vehicle is typically assumed to be moving at a piece-wise constant velocity, and the coordinate of each point is transformed according to the displacement between the point's timestamp and the motion compensation timestamp. This is the baseline motion compensation strategy used in all public autonomous driving datasets [15], [17], [19], [25], [26].

The second component of this distortion, non-ego motion distortion, occurs due to the motion of other agents in the scene. To the best of our knowledge, no existing work addresses the correction of such distortion in raw point cloud data. While dynamic object segmentation methods [20], [21], [22] are related to motion understanding, they do not focus on distortion correction. The recent work SMORE [27] mentions this dynamic rolling shutter distortion in point clouds. However, their work focuses on combining multiple frames of distorted data to improve the quality of reconstructed object meshes, without correcting the raw data itself. Instead, our work focuses on the distortion correction of *raw data* inside a single LiDAR frame, with many possible downstream

IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

applications. Additionally, the proposed HiMo pipeline is self-supervised, while SMORE requires ground truth tracking labels or a tracking network trained with annotated data.

In summary, our work aims to employ established self-supervised scene flow methods alongside ego-motion compensation to account for all distortions in the raw LiDAR data properly. As such, our method is general and agnostic to the downstream application.

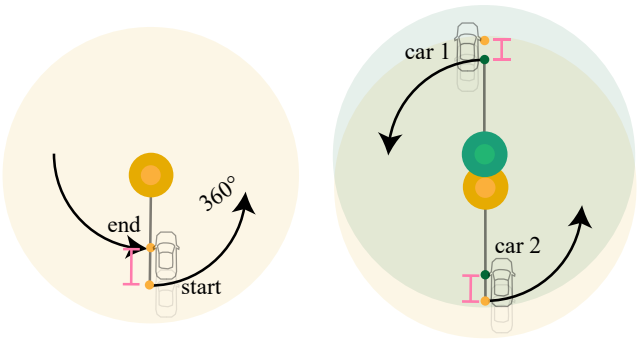
B. Scene Flow Estimation

Scene flow estimation is the task of describing a 3D motion field between temporally successive point clouds [23], [28], [29], [30], [31], [32], [8]. Existing works applied to autonomous driving datasets can be categorized into supervised [24], [33], [34], [35] and self-supervised flow estimation [36], [37], [38], [39], [40], [41], [42], [43].

Most supervised networks employ an object detection backbone and connect it to a decoder that generates output flow. For instance, FastFlow3D [24] uses a feedforward architecture based on PointPillars [44], an efficient LiDAR detector architecture, enabling efficient training and inference of flow in the real world. DeFlow [33] integrates GRU [45], [46] with iterative refinement in the decoder design for voxel-to-point feature extraction and boosts the performance of flow estimation. Supervised methods require training with detailed ground truth flow labels. Such labels are expensive and time-consuming to gather, limiting the scalability of these methods.

To train models without labeled data or directly optimize at runtime, researchers propose self-supervised pipelines for scene flow estimation [36], [38], [39], [40], [47]. Neural Scene Flow Prior (NSFP) [38] provides high-quality scene flow estimates by optimizing MLP layers at test time to minimize the Chamfer distance and maintain cycle consistency. FastNSF [39] leverages the same optimization but achieves significant speedup by computing the Chamfer loss using distance transform [48]. ICP-Flow [40] performs Iterative Closest Point (ICP) between each cluster in two point clouds and trains a feedforward neural network for real-time inference. SeFlow [36] integrates dynamic awareness and proposes novel self-supervised loss terms to allow for efficient training using large datasets.

Existing scene flow estimation methods are not intended to address non-ego motion distortions in raw point cloud data. However, they capture point-level velocity information that is valuable for the task. Guided by this insight, in this work, we repurpose scene flow estimation as a key component to correct non-ego motion distortions. In developing HiMo, we observed that existing self-supervised scene flow estimators often produce degenerate or inconsistent estimates under high-speed distortion. To mitigate these inconsistencies, we introduce SeFlow++, a scene flow estimator built on SeFlow but with higher training efficiency and better performance, particularly in high-speed scenarios. Specifically, we design a dynamic auto-labelling module (See Section IV-A) to improve the self-supervised signal at high-speed motion and a symmetric loss computation (See Section IV-B) with a larger three-frame model backbone to improve training efficiency with limited data.



(a) (b)

Fig. 3: Top-view example of LiDAR sweeps showing how distortions are created for vehicles with (a) a single LiDAR and (b) two LiDARs. The small concentric circles of yellow and green are LiDARs. Both cases cause a displacement distance for the high-speed object. The light and dark cars show the vehicle's positions at two timestamps (the beginning and the end of the LidDAR sweep). (a) One complete single LiDAR scan sweep (small yellow dots are the first and last scan points). This case is only observed when moving objects are at the scan boundaries. (b) Two LiDAR scans separated in orientation by 180 degrees. This case is always observed for fast-moving objects. An animation illustrating both cases is included in the supplementary video.

III. Motion Compensation

In this section, we begin by discussing the point cloud distortion caused by dynamic objects in autonomous driving. We then propose our general HiMo pipeline to address this challenge.

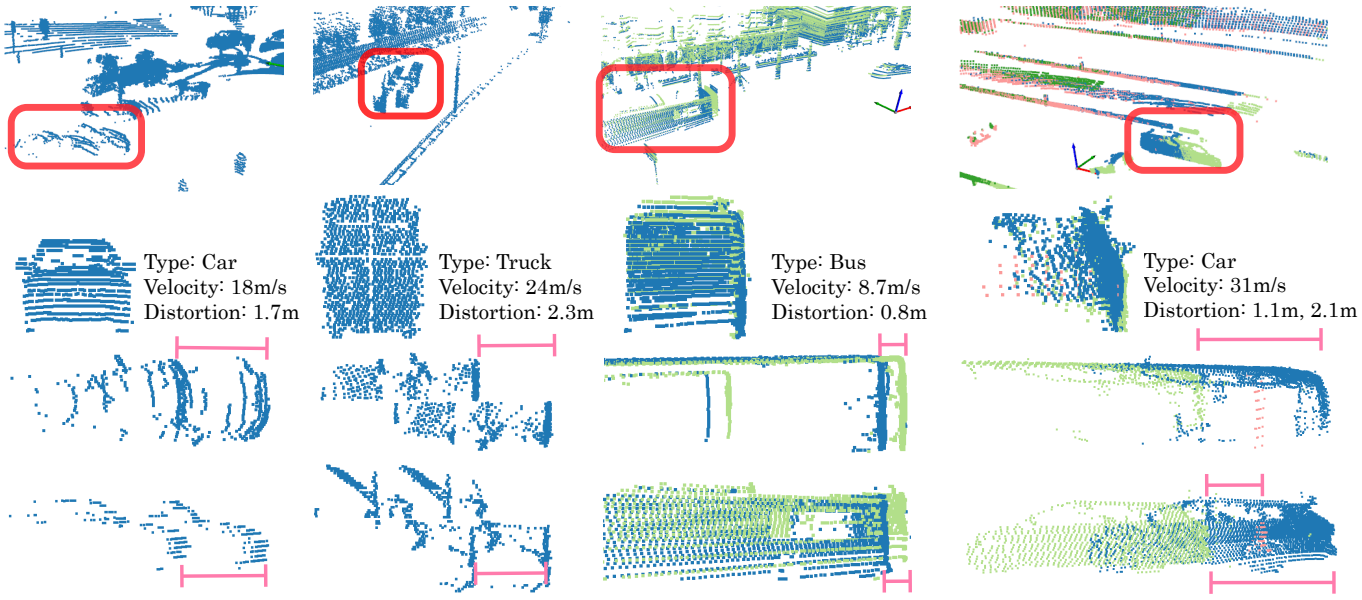
A. Non-ego Motion Distortion

Commonly used mechanical LiDAR sensors operate by sweeping laser beams in a horizontal ring pattern. This scanning process takes a certain amount of time to complete a full 360-degree sweep. If the sensed objects move during the scan, the captured point cloud will be distorted. The degree of the distortion is highly correlated with the velocity of the observed object (v_{object}) and sensor frequency (f_{sensor}), with the maximum distorted distance being $v_{\text{object}}/f_{\text{sensor}}$.

Fig. 3 illustrates this distortion in the single-LiDAR and multi-LiDAR scenarios. In the single-LiDAR case, this distortion is most visible when the dynamic object is positioned at the edge of the scan (see Fig. 3(a)). This can lead to severe shape distortion in the resulting point cloud. Fig. 4(a) and (b) provide examples from Waymo [25] and ZOD [26], respectively.

In multi-LiDAR systems, the distortion problem is even more pronounced. As shown in Fig. 3(b), multiple LiDARs mounted on the same vehicle capture the same moving object at slightly different times and from different perspectives. This leads to multiple copies of the same object in the data. An example of this can be seen in Fig. 4(c) and (d) in Argoverse 2 [17] and our Scania dataset, respectively. In this figure, each color represents a single-frame data from a separate

IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.



(a) Waymo (b) ZOD Drives (c) Argoverse 2 (d) Scania (ours)

Fig. 4: Examples of LiDAR distortion in various datasets after ego-motion compensation [17], [18], [26]. Ground points are removed in visualizations for a clearer view. Each column shows an example from a different dataset. Within each column, the top image shows the full scene, while the three images below show the zoomed-in front, top-down, and side views, respectively. (a) and (b) showcase the scenario described in Fig. 3(a), where an object is captured right at the beginning and at the end of the scan in a single LiDAR setup. (c) and (d) showcase the multi-LiDAR distortion scenario described in Fig. 3(b). The different colors in these two subfigures represent data from different LiDARs.

LiDAR. The distance between these differently colored copies of the object demonstrates the distortion effect in multi-LiDAR setups.

B. HiMo Pipeline

In most public datasets, point cloud data is routinely ego-motion compensated [17], [25], [26]. However, as shown in Fig. 1 (c) and Fig. 4, this compensation cannot correct the distortions caused by the motion of other agents in the scene. To fully compensate for **all** dynamics in the scene, we propose the following HiMo pipeline. The schematics of the pipeline can be seen in Fig. 5.

Given the raw input point cloud $\mathcal{P}_{raw}$ from a scene, the goal of the HiMo pipeline is to recover the corresponding point cloud $\mathcal{P}'$ that accurately describes the environment, where all motion-related distortions are corrected. This $\mathcal{P}'$ can be recovered if we estimate the 3D distortion correction vector for each individual point, denoted as $\mathcal{D}(\mathbf{p})$. Using this vector, the estimated undistorted point cloud then can be expressed as $\mathcal{P}' = \mathcal{P}_{raw} + \mathcal{D}$.

Note that the 3D distortion correction vector $\mathcal{D}$ can be expressed as follows:

$$\mathcal{D}(\mathbf{p}) = \mathcal{V}(\mathbf{p})\Delta T(\mathbf{p}), \tag{1}$$

where $\mathcal{V}(\mathbf{p})$ is the velocity of the point, and $\Delta T(\mathbf{p}) \in [0, T_{\text{sensor}}]$ is the time difference to the timestamp of the last point in this LiDAR scan. T_{sensor} denotes the duration of a full LiDAR scan (i.e., a complete 360° sweep), and depends on the scan frequency f_{sensor}, such that $T_{\text{sensor}} = 1/f_{\text{sensor}}$.

Further note that $\mathcal{V}(\mathbf{p})$ can be approximated from the flow of point p:

$$\mathcal{V}(\mathbf{p}) = \mathcal{F}(\mathbf{p})/T_{\text{sensor}}, \tag{2}$$

where $\mathcal{F}(\mathbf{p}) = (x, y, z)^T$ is the 3D flow vector of point $\mathbf{p}$ from the current scene to the next. In our work, this flow is extracted from the outputs of scene flow estimators. Although the goal is to correct distortions within a single LiDAR frame, consecutive frames (≥ 2) are needed for scene flow estimation.

In summary, HiMo achieves effective motion compensation by repurposing scene flow methods as velocity estimators to compute distortion correction vectors. As highlighted in Fig. 5, the pipeline is agnostic to the choice of scene flow estimator, allowing for the adoption of improved scene flow methods as they emerge.

IV. Scene Flow

Scene flow is the core module in our HiMo motion compensation pipeline. To train or optimize a scene flow estimator, we need a supervision signal and a proper loss function. In supervised training, the signal is provided by human-labeled ground truth flow [33], [35]. However, such human annotation is costly and potentially error-prone due to the distortions of high-speed vehicles in raw LiDAR data. We therefore focus on self-supervised methods that do not rely on annotations. In our experiments, we found that existing self-supervised scene flow methods either require huge computational resources or do not perform well under the conditions of scarce training data and high-speed object distortions. To adapt to the high-speed

IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

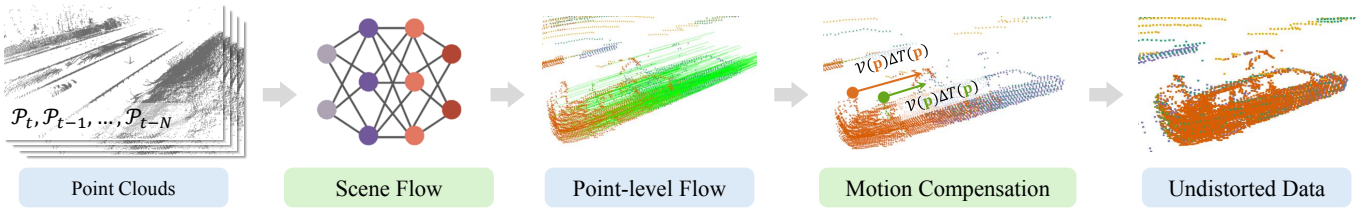


Fig. 5: Schematic of the HiMo pipeline. Given a sequence of consecutive point cloud frames, a scene flow estimator is employed to calculate the flow of each point. This flow, together with the known LiDAR scan interval and point time difference $\Delta T(\mathbf{p})$, allows us to compute the 3D distortion correction vector. Finally, the undistorted point cloud is computed by combining the correction vector with the raw point cloud.

regime and reduce the requirement on the amount of training data, we present SeFlow++, a new scene flow method based on the previous state-of-the-art method SeFlow [36]. Compared to SeFlow, SeFlow++ enhances training efficiency and temporal consistency by combining symmetric loss functions with a larger three-frame model backbone, and additionally provides more robust supervision signals through a refined dynamic auto-labeler.

A. Auto-labeler

To obtain a self-supervision signal, SeFlow [36] classifies points into static and dynamic based on DUFOMap [49]. The dynamic points are then grouped into clusters, as shown in Fig. 6 (top). However, we observe that the scene flow estimator trained using the SeFlow strategy is not always accurate. This is because errors in the dynamic point classification of DUFOMap get propagated into the subsequent object clustering step, resulting in misclassification of dynamic and static points. To refine the auto-labeling in SeFlow++, we compute the dynamic clusters by aggregating more information. As shown in Fig. 6 (bottom), this process starts with the parallel computation of clusters and point-level dynamic classification:

1) Clustering: We cluster all points in $\mathcal{P}_t$ using HDBSCAN [50] into object instances $\mathcal{C} = \{c_1, c_2, \cdots, c_n\}$.

2) Dynamic classification: : To improve the robustness of this module, we employ two independent dynamic classification methods: DUFOMap [49] and a threshold-based nearest neighbor method [51]. The key insight of DUFOMap is that points observed inside a region that at any time has been observed as empty must be dynamic. The empty regions can be inferred using ray-casting. The result of DUFOMap for $\mathcal{P}_t$ is a dynamic point set $\mathcal{P}_{\text{dufo}}$. The second method, threshold-based nearest neighbor [51], requires two consecutive point clouds $\mathcal{P}_t$ and $\mathcal{P}_{t+1}$ as input. The dynamic point set is defined by:

$$\mathcal{P}_{\text{nnd}} = \{\mathbf{p} | \mathrm{D}_{\text{min}}(\mathbf{p}, \mathcal{P}_{t+1}) > \tau_d, \mathbf{p} \in \mathcal{P}_t\}, \tag{3}$$

where $\mathrm{D}_{\text{min}}(\mathbf{p}, \mathcal{P}_{t+1})$ represents the distance between point $\mathbf{p}$ and its nearest neighbor in $\mathcal{P}_{t+1}$, and τ_d is a user defined threshold.

Reassign label: Given the clusters $\mathcal{C}$ and the two dynamic point classification results $\mathcal{P}_{\text{dufo}}$ and $\mathcal{P}_{\text{nnd}}$ on an input point set $\mathcal{P}_t$, the final subset of dynamic points is defined as follows (the subscript t is dropped for readability):

$$\mathcal{P}_d = \bigcup \{\mathbf{p} | f(c), \mathbf{p} \in \mathcal{P}_c, c \in \mathcal{C}\}, \tag{4}$$

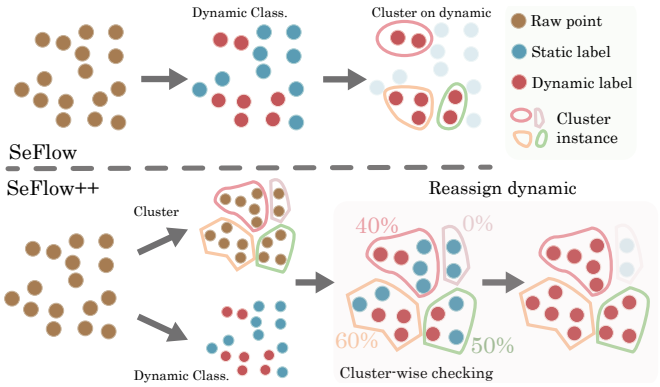


Fig. 6: Dynamic auto-labeling process in SeFlow (top) and the improved SeFlow++ (bottom). In SeFlow, dynamic classification, denoted by *Dynamic Class.*, is first performed by DUFOMap. Then, this classification is used to guide dynamic object clustering. In SeFlow++, on the other hand, we simultaneously perform object clustering and dynamic classification using two independent methods. The *Cluster-wise checking* step then reassigns labels based on whether the proportion of dynamic points labeled in cluster instances exceeds the thresholds.

where $\mathcal{C}$ is the set of all clusters based on HDBSCAN on $\mathcal{P}$, c and $\mathbf{p}$ represent an individual cluster and a point in the cluster, respectively. The function f integrates the two dynamic labels $\mathcal{P}_{\text{dufo}}$ and $\mathcal{P}_{\text{nnd}}$ is defined as follows:

$$f(c) = \begin{cases} \texttt{true}, & \text{if } \min(r_1, r_2) \geq \tau_1 \ \& \ \max(r_1, r_2) \geq \tau_2 \\ \texttt{false}, & \text{otherwise} \end{cases}, \tag{5}$$

where $r_1 = \frac{|\mathcal{P}_{\text{dufo}}(c)|}{|\mathcal{P}_c|}$ and $r_2 = \frac{|\mathcal{P}_{\text{nnd}}(c)|}{|\mathcal{P}_c|}$ represent the proportion of dynamic points labeled in the cluster instance c by the two methods respectively. $|\mathcal{P}|$ denotes the cardinality of (i.e., the number of points in) the point cloud $\mathcal{P}$. Conceptually, this means that we examine each individual object clusters c separately, and assign all points inside this cluster as dynamic only if both methods identify a proportion of dynamic points of at least τ_1, and at least one of them provides a stronger signal with a proportion of at least τ_2. The decision thresholds τ_1, τ_2 are hyperparameters.

In summary, this improved auto-labeler incorporates clustering and point-level dynamic classification results to provide more robust and accurate object-level dynamic labels. For each point, these labels provide information on its dynamicness, as

well as its object cluster, both of which are utilized in the subsequent self-supervised training.

B. Self-supervised Loss

The most popular loss in self-supervised scene flow estimation is the Chamfer distance, a standard metric for the shape dissimilarity between two point clouds. The definition is:

$$\mathrm{CD}(\mathcal{P}_i, \mathcal{P}_j) = \frac{\sum_{p\in\mathcal{P}_i} \mathrm{D}_{\min}(p, \mathcal{P}_j)}{|\mathcal{P}_i|} + \frac{\sum_{p\in\mathcal{P}_j} \mathrm{D}_{\min}(p, \mathcal{P}_i)}{|\mathcal{P}_j|} \tag{6}$$

SeFlow [36] highlighted the association errors in the Chamfer distance and proposed a more robust loss design with the following four-term loss function:

$$\mathcal{L}_{total} = \mathcal{L}_{\text{cham}} + \mathcal{L}_{\text{dcham}} + \mathcal{L}_{\text{static}} + \mathcal{L}_{\text{dcls}}, \tag{7}$$

where $\mathcal{L}_{\text{cham}}$ and $\mathcal{L}_{\text{dcham}}$ are Chamfer distance loss on all points and only dynamic points, respectively. $\mathcal{L}_{\text{static}}$ and $\mathcal{L}_{\text{dcls}}$ optimize flow estimation in static points and points inside dynamic object clusters, respectively. The three loss terms $\mathcal{L}_{\text{dcham}}$, $\mathcal{L}_{\text{static}}$ and $\mathcal{L}_{\text{dcls}}$ all require auto-labels from dynamic point classification. In SeFlow++, these labels are obtained as per description in Section IV-A.

Although SeFlow achieves state-of-the-art performance in self-supervised scene flow estimation, we noticed that it occasionally produces degenerate or inconsistent flow estimation between consecutive frames. This inconsistency introduces errors when we apply the estimated flow to correct motion distortions. Prior works have introduced cycle consistency to improve scene flow estimation and mitigate degenerate solutions, particularly in sparse point clouds [52], [38]. However, these methods rely on explicit backward flow computation or duplicated networks, which are inefficient. To encourage consistent flow estimate with minimal overhead, we propose a symmetric flow loss that can be calculated within a single forward pass. Note that due to the symmetry around $\mathcal{P}_t$, applying the forward flow with a **negative sign** $(-)$ effectively simulates a backward flow from $\mathcal{P}_t$ to $\mathcal{P}_{t-1}$. Guided by this insight, we reformulate the first chamfer distance loss in Eq. (7) to a symmetric version as follows:

$$\mathcal{L}_{\text{cham}} = \mathrm{CD}(\hat{\mathcal{P}}_{t+1}, \mathcal{P}_{t+1}) + \mathrm{CD}(\hat{\mathcal{P}}_{t-1}, \mathcal{P}_{t-1}), \tag{8}$$

where $\hat{\mathcal{P}}_{t\pm1} = \mathcal{P}_t \pm \hat{\mathcal{F}}_t$. We also augment the dynamic chamfer distance loss $\mathcal{L}_{\text{dcham}}$ to a symmetric version, except it only considers points that are classified as dynamic in Eq. (4).

The last two loss terms in Eq. (7), the static loss $\mathcal{L}_{\text{static}}$ and the dynamic cluster loss $\mathcal{L}_{\text{dcls}}$, are kept unchanged in this work. The static loss encourages the model to estimate zero flow for static points:

$$\mathcal{L}_{\text{static}} = \frac{1}{|\mathcal{P}_s|} \sum_{\mathbf{p}\in\mathcal{P}_s} ||\Delta\hat{\mathcal{F}}(\mathbf{p})||_2^2. \tag{9}$$

For the dynamic cluster loss, as in [36], the objective is to encourage points within the same dynamic cluster to have similar scene flow. Specifically, we identify the point with the largest distance to its dynamic nearest neighbor in the next frame for each dynamic cluster, which is used to estimate a representative flow for the entire cluster. More specifically, we find the index of the point p_k in cluster $c_i \in \mathcal{C}_d$ with the largest distance to its nearest neighbor point in $\mathcal{P}_{t+1,d}$, i.e.,

$$\kappa_i = \arg\max_k \{\mathrm{D}_{\min}(p_k, \mathcal{P}_{t+1,d}) | p_k \in \mathcal{P}_{c_i}\}. \tag{10}$$

We then calculate the upper bound, $\tilde{f}_{c_i}$ on the flow for cluster c_i as

$$\tilde{f}_{c_i} = p'_{\kappa_i} - p_{\kappa_i}, \tag{11}$$

where p'_{κ_i} is the nearest neighbor to point p_{κ_i} in $\mathcal{P}_{t+1,d}$. We use this to drive the estimated flows of cluster c_i towards $\tilde{f}_{c_i}$ as follows:

$$\mathcal{L}_{\text{dcls}} = \frac{1}{|\mathcal{P}_{t,d}|} \sum_{c_i\in\mathcal{C}_{t,d}} \left(\sum_{p_j\in\mathcal{P}_{c_i}} ||\hat{f}_{p_j} - \tilde{f}_{c_i}||_2^2 \right). \tag{12}$$

In summary, SeFlow++ improves the SeFlow loss from two aspects. Firstly, it improves the temporal consistency and data efficiency by augmenting $\mathcal{L}_{\text{cham}}$ and $\mathcal{L}_{\text{dcham}}$ with symmetric flow computation. Secondly, it refines the dynamic auto-labeling using the procedure described in Section IV-A.

V. Experiments Setup

A. Dataset

Experiments are conducted mainly on two large-scale autonomous driving datasets: our highway dataset (Scania) and Argoverse 2 [17]. Additionally, qualitative results from the Zenseact Open Dataset [26] (ZOD) are presented. For all datasets, ground removal is performed using line-fit ground segmentation [53].

Scania Our dataset consists of around 500 sequences, 10 to 15 seconds per sequence, captured in and around downtown Södertalje, Sweden. The total training dataset consists of 48,936 frames from 500 sequences. The validation set consists of 100 frames from 10 sequences containing high-speed scenarios. Our platforms, as shown in Fig. 7, consist of buses and trucks equipped with multiple 32-channel LiDARs. The frequency of each LiDAR is around 10Hz. Points from multiple LiDARs within a fixed time interval (T_{sensor}) are combined into a single point cloud.

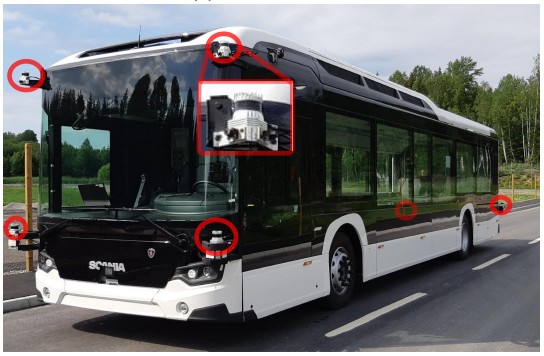
Argoverse 2 Its *Sensor* dataset encompasses 700 training (110,071 frames) and 150 validation sequences. A passenger car equipped with two roof-mounted VLP-32C lidar sensors running at 10 Hz is used for data collection. Each scene is approximately 15-20 seconds long, with complete annotations for evaluation. As shown in Fig. 2, the average object speed in Argoverse 2 is relatively slow. To focus the evaluation on high-speed object motion compensation, the evaluation sequence is selected based on whether it includes at least three fast-moving annotated objects.

More interactive visualization results on Argoverse 2 [17] and ZOD [26] are available on the project page[1].

[1] https://kin-zhang.github.io/HiMo

IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

(a) Scania Truck



(b) Scania Bus

Fig. 7: Our Scania dataset contains data collected from two different multi-LiDAR setups. (a) A truck with six LiDARs on the top of the truck and two on the front bottom. There are two at the back of the truck that are not visible in the image. (b) A bus with four in the front of the vehicle and two in the back of the bus.

B. Annotated Ground truth

To evaluate the non-ego motion compensation performance, we manually annotated validation data and created undistorted ground truth data through a multi-step process. A flowchart of the annotation and refinement pipeline is provided in Appendix A.

Recall from Eq. (1) that the 3D distortion correction vector $\mathcal{D}$ can be computed using the velocity $\mathcal{V}$ and the time difference ΔT. We can therefore compute the ground truth using the annotated point velocities, which we obtain through object tracking annotations. For Argoverse 2, the official sensor annotations already contain manually labelled object bounding boxes and their tracking IDs. For Scania, we obtained these by manual labeling of bounding boxes using the protocol in [54]. However, since we chose to evaluate high-speed scenarios, the distortions in the raw data are significant enough to cause errors in the manual object labels. To mitigate these errors, we enlarged the bounding boxes in the direction of motion to include all object copies. These refined object tracking labels, combined with the known scanning timestamps and sensor frequency, were used to generate the motion compensation ground truth. To ensure geometric consistency, we manually reviewed the ground truth in each frame before including it in the validation set.

C. Evaluation Metrics

Due to the lack of well-established motion distortion metrics in the literature, we present two metrics: one captures shape similarity inspired by 3D reconstruction, and the other measures point-level accuracy similar to end point error in scene flow.

Shape similarity measures the correctness of shape descriptions. We quantify this similarity using the Chamfer distance error (CDE) defined as:

$$\text{CDE} = \frac{1}{|\mathcal{C}_{gt}|} \sum_{c_i \in \mathcal{C}_{gt}} \left(\frac{|\mathcal{P}_{c_i}|}{|\mathcal{P}_{\mathcal{C}_{gt}}|} \text{CD}(\mathcal{P'}_{est,c_i}, \mathcal{P'}_{gt,c_i}) \right), \quad (13)$$

where $|\cdot|$ denotes cardinality of a set, $\mathcal{C}_{gt}$ denotes the ground truth clusters in the frame, $\mathcal{P}_{c_i}$ and $\mathcal{P}_{\mathcal{C}_{gt}}$ denote the set of points in cluster c_i and in all ground truth clusters, $\mathcal{P'}_{est,c_i}$ and $\mathcal{P'}_{gt,c_i}$ denote the point set c_i compensated using the estimated motion and ground truth, respectively. The smaller the CDE, the greater the shape similarity between the estimated instance shape and the ground truth.

While we believe that shape similarity best measures a method's ability to undistort moving objects, its reliance on Chamfer distance implies that it is based on nearest neighbor matching. Such matching results do not guarantee correct association at point level. Therefore, we also compute the point-level accuracy that can be represented as the mean point error (MPE),

$$\text{MPE} = \frac{\sum_{c_i \in \mathcal{C}_{gt}} \left(\sum_{\mathbf{p} \in \mathcal{P}_{c_i}} \|\mathbf{p}_{est} - \mathbf{p}_{gt}\|_2 \right)}{|\mathcal{C}_{gt}||\mathcal{P}_{\mathcal{C}_{gt}}|} \quad (14)$$

where $\mathbf{p}_{est}$ and $\mathbf{p}_{gt}$ denote the estimated and corresponding ground truth point, respectively. The rest of the notations are the same as per CDE.

Note that these two metrics have different focuses: CDE captures the error at an object level, and quantifies the correctness of the undistorted object shape, whilst MPE metric captures the errors at a point level.

Since the motion distortion in high-speed objects is more pronounced (see Section III-A and Fig. 4), we separately report metrics for two types of vehicle categories. In all result tables, CAR means regular and passenger vehicles, and OTHER VEHICLES (OTHERS) include trucks, long buses, heavy vehicles, vehicles with trailers, etc.

D. Evaluated Methods

In this work, we address the problem of non-ego motion distortion compensation in LiDAR point clouds, which has not been tackled by prior methods. The only existing baseline for motion compensation is ego-motion compensation, where the point cloud data is adjusted solely for the motion of the ego vehicle. In this baseline, the distortion correction vector $\mathcal{D}$ is the velocity of the ego vehicle at the current timestamp, scaled by $\Delta T(p)$ for each point.

To address non-ego motion distortion compensation, we propose the HiMo pipeline (see Fig. 5), which leverages scene flow estimation to perform motion compensation. Given consecutive frames of point cloud data, HiMo employs any

IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

TABLE I: Object motion compensation result comparisons using different scene flow methods in our HiMo pipeline on the Scania validation set. The first row reports the errors for raw data with ego-motion compensation only, and the rest is our HiMo pipeline motion compensation with different scene flow estimators as an ablation study. Upper groups are supervised methods with scene flow networks trained on Argoverse 2 [17] and inference directly on the Scania dataset. Lower groups are self-supervised methods. $\pm$ means the standard deviation in the evaluation data (100 frames). All methods in the HiMo pipeline achieve better accuracy performance and object shape similarity compared to raw data. Our proposed SeFlow++ achieves state-of-the-art compensated performance in shape description for both car and other vehicle object types. The $_{\downarrow blue}$ value represents the error percentage decrease relative to the distorted raw data.

Methods		Reference	Chamfer Distance Error (CDE) ↓			Mean Point Error (MPE) ↓		
			Total	CAR	OTHERS	Total	CAR	OTHERS
Ego-motion Compensation		-	0.284	0.257 ± 0.13	0.310 ± 0.11	0.935	0.913 ± 0.16	0.957 ± 0.07
HiMo (Ours)	FastFlow3D [24]	RAL'22	0.144 $_{\downarrow 49\%}$	0.121 ± 0.04	0.168 ± 0.04	0.546 $_{\downarrow 42\%}$	0.378 ± 0.16	0.714 ± 0.17
	DeFlow [33]	ICRA'24	0.088 $_{\downarrow 69\%}$	0.057 ± 0.02	0.118 ± 0.05	0.315 $_{\downarrow 66\%}$	**0.139** ± 0.08	0.491 ± 0.19
	NSFP [38]	NeurIPS'21	0.073 $_{\downarrow 74\%}$	0.064 ± 0.07	0.083 ± 0.04	**0.255** $_{\downarrow 73\%}$	0.188 ± 0.16	0.323 ± 0.15
	FastNSF [39]	ICCV'23	0.078 $_{\downarrow 72\%}$	0.074 ± 0.05	0.081 ± 0.05	0.279 $_{\downarrow 70\%}$	0.251 ± 0.16	**0.308** ± 0.14
	ICP-Flow [40]	CVPR'24	0.183 $_{\downarrow 36\%}$	0.203 ± 0.13	0.163 ± 0.05	0.695 $_{\downarrow 26\%}$	0.698 ± 0.21	0.692 ± 0.18
	SeFlow [36]	ECCV'24	0.096 $_{\downarrow 66\%}$	0.094 ± 0.04	0.098 ± 0.01	0.452 $_{\downarrow 52\%}$	0.444 ± 0.17	0.461 ± 0.18
	SeFlow++ (Ours)	-	**0.054** $_{\downarrow 81\%}$	**0.050** ± 0.03	**0.059** ± 0.02	0.267 $_{\downarrow 72\%}$	0.179 ± 0.10	0.356 ± 0.18

scene flow estimators to estimate point velocities and then transforms these velocities into distortion distances. These distortion distances then can be used to undistort all motions in the raw point cloud.

As mentioned in Section III-B, the HiMo pipeline is agnostic to the choice of scene flow estimators. To evaluate the performance and highlight the flexibility of HiMo, we incorporate different state-of-the-art scene flow estimators into the pipeline. The evaluated scene flow estimators[2] are outlined below:

1) FastFlow3D [24]: A supervised model trained on the Argoverse 2 sensor dataset.
2) DeFlow [33]: A supervised model featuring a voxel-to-point flow decoder with refinement, also trained on the Argoverse 2 sensor dataset.
3) NSFP [38]: A runtime optimization method that uses Chamfer distance between $\mathcal{P}_t + \hat{\mathcal{F}}_t$ and $\mathcal{P}_{t+1}$. It needs thousands of iterations to optimize a simple neural network to output the flow of each new frame.
4) FastNSF [39]: A runtime speedup version of NSFP that uses a distance transform to calculate the nearest neighbor error.
5) ICP-Flow [40]: This approach first processes point clouds with a clustering algorithm. Then it employs the conventional Iterative Closest Point (ICP) algorithm that aligns the clusters over time and outputs the corresponding rigid transformations between frames.
6) SeFlow [36]: This method integrates efficient dynamic classification into a learning-based scene flow pipeline and designs three novel losses to achieve self-supervised flow training as described in Eq. (7).
7) SeFlow++ (Ours): Proposed in this paper, it improves SeFlow with refined dynamic auto-labeling (see Section IV-A). It leverages a three-frame DeFlow backbone to implement a symmetric self-supervised loss (see Section IV-B), enhancing flow consistency and training efficiency.

[2]The scene flow performance of these scene flow methods is provided in Appendix B.

All code to reproduce results and run the HiMo pipeline can be found in https://github.com/KTH-RPL/HiMo. The main training hyperparameters are listed here for Scania highway datasets: learning rate ($2e^{-4}$) with Adam optimizer [55], batch size (8), the total training epoch (9). More configuration details can be found in the code. All runtime experiments are executed on a desktop powered by an Intel Core i9-12900KF CPU and equipped with a GeForce RTX 3090 GPU.

VI. Results and Discussion

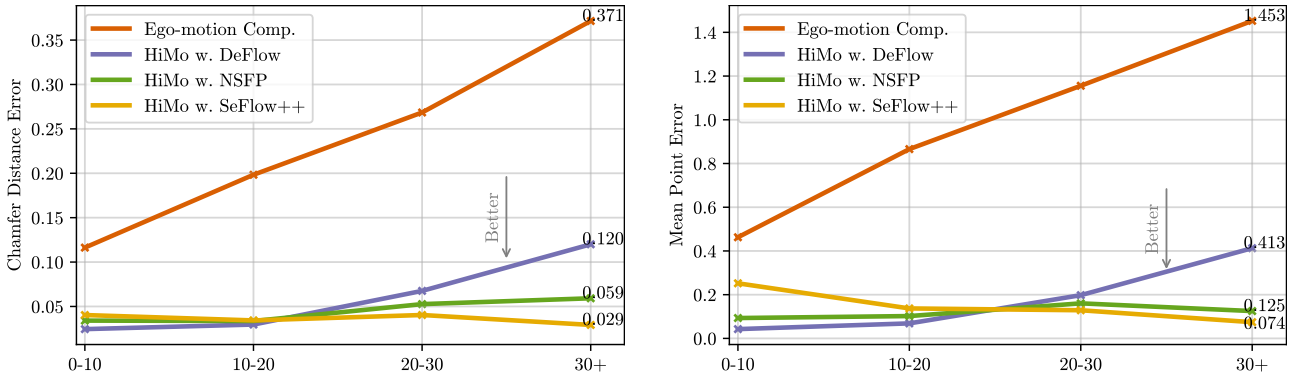
A. Quantitative Results

The comparative analysis of different scene flow methods using the HiMo pipeline on the Scania dataset is detailed in Table I. The baseline results from *Ego-motion Compensation* are shown on the first row, with large values on both CDE and MPE. Perfect undistortion would result in both metrics being zero. As shown in the rest of Table I, compared to this baseline, the HiMo pipeline reduces distortion errors in both CDE and MPE regardless of the scene flow estimator it is coupled with. This demonstrates the effectiveness of our HiMo pipeline in motion compensation, with up to 81% in shape improvement. However, when combined with different scene flow estimators, the performance of the HiMo pipeline differs. Despite not having seen Scania data before, DeFlow performs well on the CAR category according to both CDE and MPE. This is because the car-type objects in Argoverse 2 and Scania data are similar in shape. Comparatively, DeFlow does not perform as well on the OTHERS category. This is because the OTHERS category contains objects with high motion speed and long vehicle sizes, with heavier motion-related distortions in raw data. Such distortions cause difficulty in transferring the knowledge from the previous dataset. Compared to supervised methods, most self-supervised methods achieve both lower CDE and MPE on the OTHERS category. The only exception is ICP-Flow, which performs on par or worse than the supervised methods on both CAR and OTHERS categories. This is

IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

TABLE II: Object motion compensation result comparisons using different scene flow methods in our HiMo pipeline on Argoverse 2. The first row reports the errors for raw data with ego-motion compensation only, and the rest also includes compensation for non ego-motion distortion using HiMo. Self-supervised methods are listed without human-labeled data needed to undistort raw data. ± means the standard deviation in the evaluation data (100 frames). All methods in the HiMo pipeline achieve better accuracy performance and object shape similarity compared to raw data. The $_{\downarrow blue}$ value represents the error percentage decrease relative to the distorted raw data.

Methods		Reference	Chamfer Distance Error (CDE) ↓			Mean Point Error (MPE) ↓		
			Total	CAR	OTHERS	Total	CAR	OTHERS
Ego-motion Compensation		-	0.180	0.176 ± 0.02	0.185 ± 0.01	0.619	0.585 ± 0.13	0.654 ± 0.03
HiMo (Ours)	NSFP [38]	NeurIPS'21	0.052 $_{\downarrow 71\%}$	0.073 ± 0.03	0.032 ± 0.01	0.144 $_{\downarrow 77\%}$	0.209 ± 0.12	<u>0.079</u> ± 0.02
	FastNSF [39]	ICCV'23	0.079 $_{\downarrow 56\%}$	0.103 ± 0.03	0.054 ± 0.00	0.260 $_{\downarrow 58\%}$	0.331 ± 0.14	0.190 ± 0.00
	ICP-Flow [40]	CVPR'24	0.053 $_{\downarrow 71\%}$	0.060 ± 0.03	0.046 ± 0.00	0.135 $_{\downarrow 78\%}$	0.168 ± 0.13	0.101 ± 0.00
	SeFlow [36]	ECCV'24	<u>0.040</u> $_{\downarrow 78\%}$	<u>0.041</u> ± 0.01	**0.039** ± 0.00	<u>0.073</u> $_{\downarrow 88\%}$	<u>0.059</u> ± 0.02	0.088 ± 0.01
	SeFlow++ (Ours)		**0.038** $_{\downarrow 79\%}$	**0.037** ± 0.01	<u>0.040</u> ± 0.00	**0.067** $_{\downarrow 89\%}$	**0.058** ± 0.02	**0.077** ± 0.00



(a) Chamfer distance error v.s. object velocity (m/s)

(b) Mean point error v.s. the object velocity (m/s)

Fig. 8: Error distribution concerning object velocity (m/s) of CAR category for the three best performing flow estimators in our HiMo pipeline on Scania data. The approximately linear relationship between velocity and error is clear in the ego-motion compensated data.

because ICP-Flow employs many heuristics in its optimization and ICP matching procedure. Hence, extensive parameter tweaking is needed for different data and scenarios. As shown in Table II, ICP-Flow performs much better on Argoverse 2, the dataset it is optimized for. The best performance in shape similarity, quantified by the lowest CDE, is achieved by the HiMo pipeline with our proposed SeFlow++. This setup also shows competitive performance on MPE.

To highlight the effectiveness of HiMo, we also provide a quantitative evaluation of the pipeline on the public Argoverse 2 dataset in Table II. Comparing the baseline *ego-motion compensation* results on Scania in Table I and Argoverse 2 in Table II, we can notice that Argoverse 2's baseline has both lower CDE and MPE. This is caused by the lower object speeds as shown in Fig. 2. As shown in Table II, the HiMo pipeline achieves at least 50% error reduction when combined with any of the five tested self-supervised scene flow estimators.

B. Error Distribution

In Fig. 8, we explore the error distributions associated with objects in the CAR category across varying velocities. The x-axis represents the velocity range of the objects, segmented into intervals (0-10 m/s, 10-20 m/s, etc.).

From the *baseline* ego-motion compensation line plots in Fig. 8, it is evident that higher object velocity is directly correlated with larger errors in both CDE and MPE. This trend confirms our discussions on distortion impacts detailed in Section III-A.

Among the methods, DeFlow, which was trained on Argoverse 2 with ground truth supervision and directly applied to our Scania data, exhibits increased errors as the object velocity increases. This suggests that DeFlow's adaptability to high-speed scenarios is constrained, likely due to its training not fully capturing the fast-moving dynamics. The low performance of this pre-trained supervised model on fast-moving objects reveals the necessity of self-supervised learning. In comparison, both self-supervised methods (NSFP and SeFlow++) in the HiMo pipeline achieve consistently low CDE and MPE across different object velocities, indicating the advantages of self-supervised learning in handling rapid motion dynamics.

C. Qualitative Results

Our qualitative analysis, as shown in Fig. 9, provides visual results on the performance of different motion compensation methods on a truck object that exhibits significant distortion. The ego-motion compensated baseline data demonstrates se-

IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

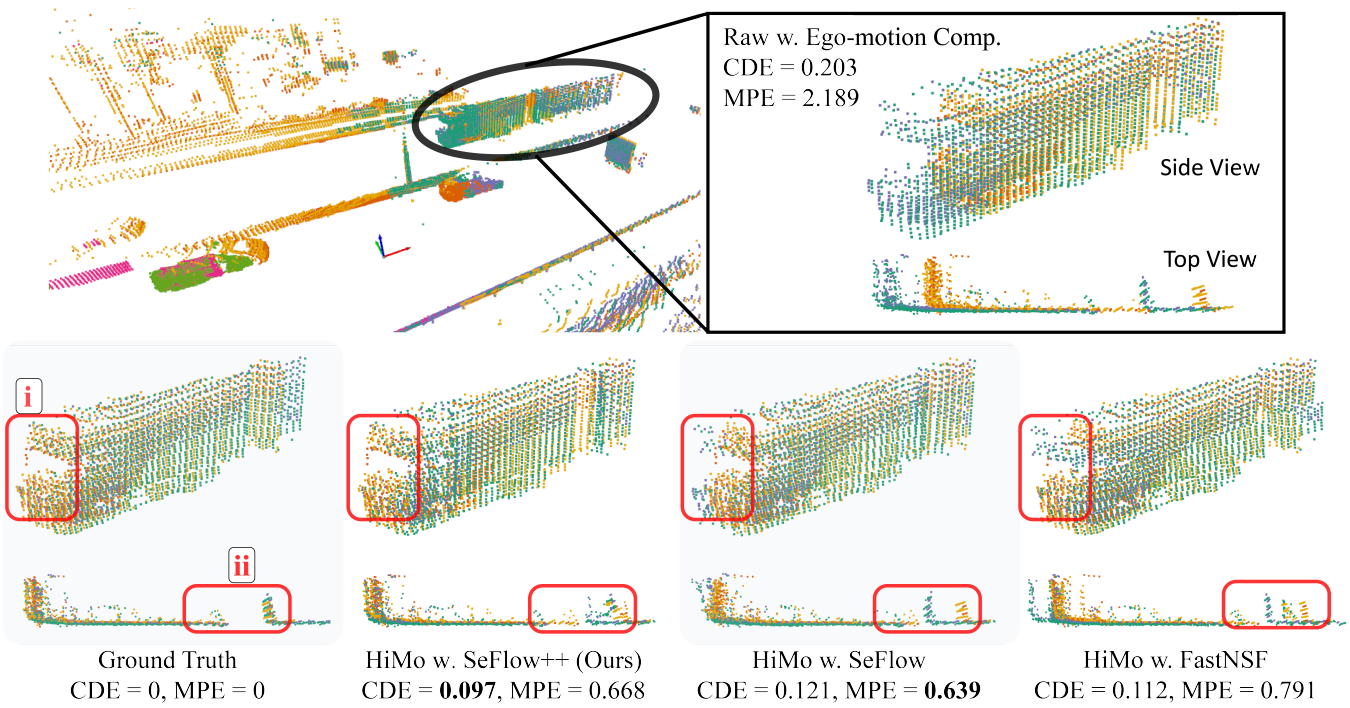


Fig. 9: Qualitative comparison of scene flow methods inside the HiMo pipeline for motion compensation on a distorted truck object. Top: Raw point cloud data after the baseline ego-motion compensation showing evident residual motion distortion. Bottom (left to right): Ground truth, SeFlow++ (Ours), SeFlow, and FastNSF results. Side and top views are provided for each method. CDE and MPE values are also reported to complement the visualizations.

vere distortion on the truck, with its shape appearing extended and fragmented due to its motion during LiDAR scanning. All scene flow estimators in our HiMo pipeline, including SeFlow++, SeFlow, and FastNSF, show improvements over the baseline in terms of both quantitative metrics (CDE and MPE) and visual appearance. However, each method exhibits different aspects in its refinement results. FastNSF provides a balanced performance, improving over the baseline in both CDE and MPE, but does not match the refinement quality of SeFlow in point accuracy or SeFlow++ in shape reconstruction. SeFlow demonstrates the best performance in terms of MPE, indicating high point-level accuracy. However, this superiority in point accuracy is not immediately apparent in the visual representation. After applying HiMo with SeFlow++, the truck's outline and structure are more coherent and closely resemble the ground truth (Fig. 9.i), as evidenced by its lowest CDE among all evaluated methods. Interestingly, SeFlow++ shows a more scattered point distribution at the center part of the truck (Fig. 9.ii), which may explain its slightly higher MPE despite better shape preservation. These observations highlight the importance of considering both shape similarity (CDE) and point-level accuracy (MPE) in evaluating motion compensation methods. While an ideal method would excel in both metrics, practical limitations often lead to tradeoffs. Among the evaluated approaches in the HiMo pipeline, SeFlow++ most effectively preserves object-level geometric integrity, and it is particularly critical for downstream perception tasks in autonomous driving.

Another qualitative motion compensation result on two regular car objects is presented in Fig. 10. All methods in our HiMo demonstrate significant improvements in both CDE and MPE for cars against the baseline, with more substantial reductions compared to the truck scenario. For instance, SeFlow++ achieves a remarkable 71% reduction in CDE and an 89% decrease in MPE. DeFlow, despite being trained on different datasets, shows competitive performance in CDE reduction (57% decrease to 0.079) for the two CAR-type objects. However, its compensation for the faster-moving car (the left vehicle) with the speed of 27 $\mathrm{m/s}$ appears less refined compared to SeFlow++ and NSFP, particularly in preserving the vehicle's shape integrity.

The qualitative results in Fig. 9 and Fig. 10 on the Scania dataset of other flow methods tested in the HiMo pipeline can be found on the project page https://kin-zhang.github.io/HiMo. To demonstrate the generalizability of our HiMo pipeline, we provide qualitative motion compensation results on public datasets Argoverse 2 [17] and ZOD [26] in Fig. 11.

D. Computational Cost Comparison

In this section, we consider the computational cost for the three top-performing self-supervised methods in our HiMo pipeline: SeFlow++, NSFP, and FastNSF as shown in Table I. These methods differ significantly in training duration, inference time and deployment efficiency, all of which are crucial factors for practical applications.

As shown in Table III, the computational time of SeFlow++ can be decomposed into 4 hours of preparation time (auto-labeling), 10 GPU hours of training, and 1 hour of inference to undistort the whole Scania dataset (around 60,000 frames). Despite its considerable initial cost during the auto-labeling and

IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

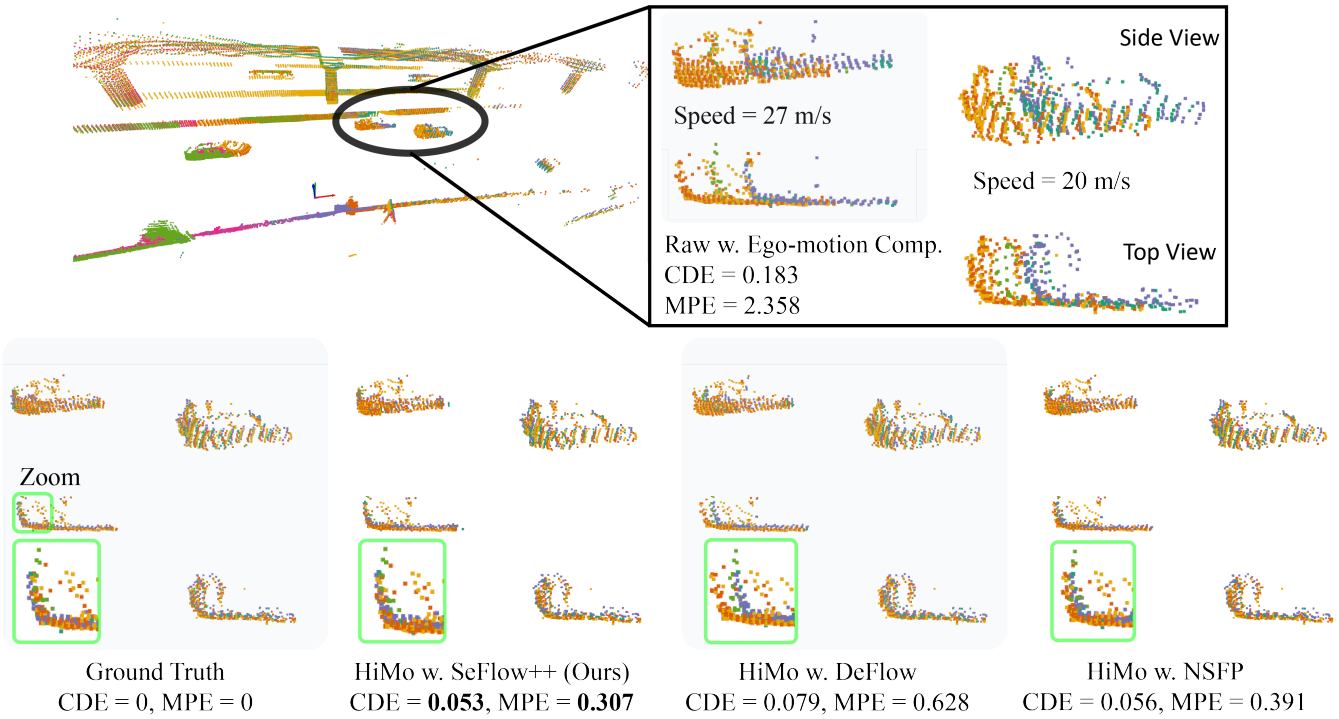
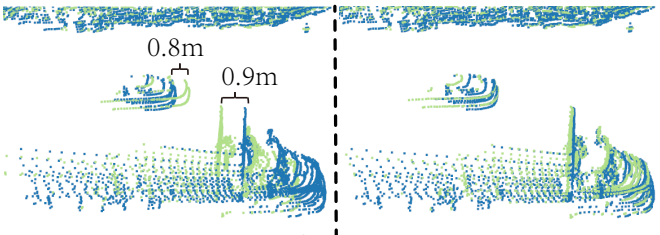
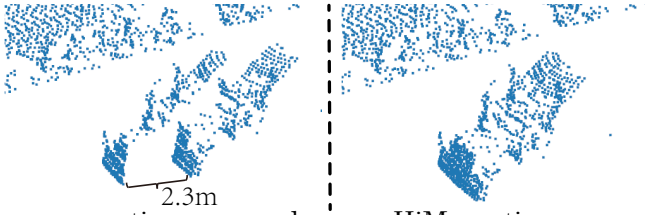


Fig. 10: Qualitative comparison of scene flow methods inside the HiMo pipeline for motion compensation on two distorted regular car objects. Top: Raw point cloud data after the baseline ego-motion compensation showing evident distortion. Bottom (left to right): Ground truth, SeFlow++ (Ours), DeFlow, and NSFP results. Side and top views are provided for each method. All scene flow methods in the HiMo pipeline demonstrate effective compensation for the distorted data, with SeFlow++ showing superior performance in both metrics.



(a) Argoverse 2



(b) ZOD Drives

Fig. 11: Our HiMo qualitative results on two public datasets. Different point colors in Argoverse 2 represent different source LiDARs. The left half of the figure shows the ego-motion compensation results provided by the dataset, and the right side is our HiMo with SeFlow++ undistortion results.

training stage, SeFlow++ excels in inference and deployment efficiency afterward on large datasets. It requires less than one GPU hour to perform motion compensation on the entire Scania dataset (0.06 seconds per frame). Conversely, NSFP, the second top-performing method on the Scania dataset, demands substantially more computational resources. It requires approximately 250 GPU hours to apply undistortion across all frames of the full dataset, averaging 15 seconds per frame. FastNSF, while quicker than NSFP, still requires around 83 GPU hours (3 seconds per frame). However, both NSFP and FastNSF are runtime optimization methods that do not require prior training. As such, they allow for quick deployment and are advantageous if one only needs to perform motion compensation on a handful of frames.

TABLE III: Comparative analysis of undistortion performance and total computational time. Performance is the average reduction in error percentage on CDE and MPE compared to baseline ego-motion compensated data, illustrating the undistortion performance efficacy of each method in HiMo. Computational Time (hours) contains training time and undistortion inference time on the full Scania dataset (around 60,000 frames).

HiMo w. Flow Estimator	Performance ↑	Computational Time (hours) ↓		
		Training	Inference	Total
NSFP [38]	73.51%	**0**	250	250
FastNSF [39]	71.35%	**0**	83	83
ICP-Flow [40]	30.62%	**0**	150	150
SeFlow [36]	58.93%	12	**0.8**	**13**
SeFlow++ (Ours)	**76.21%**	14	1	15

IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

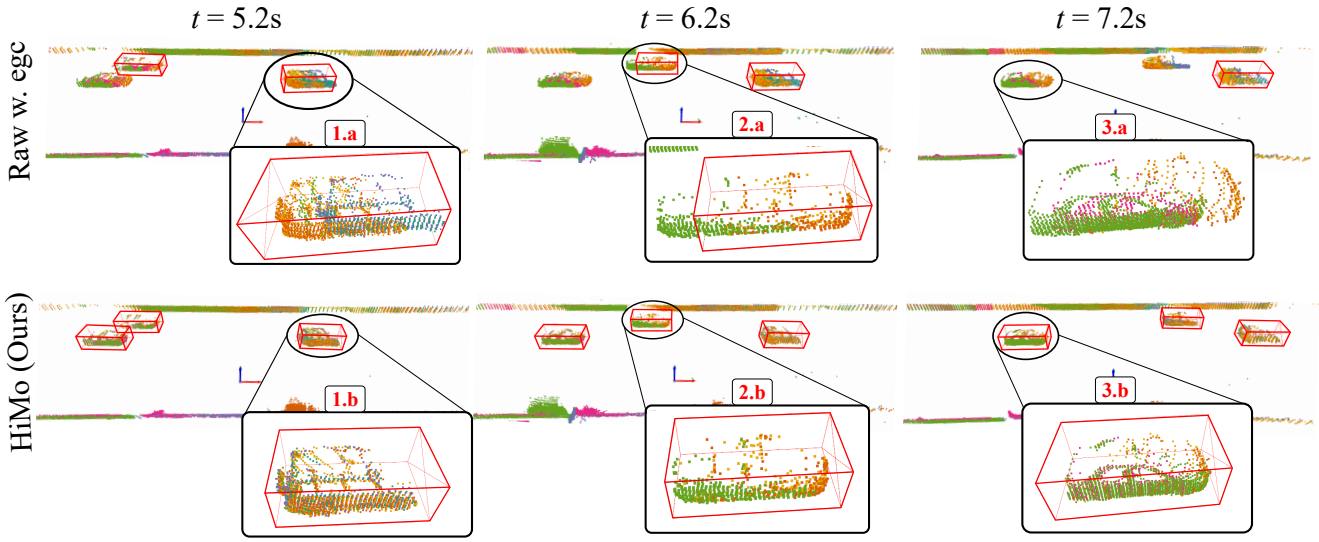


Fig. 12: Impact of motion distortion on 3D detection using TransFusion [56] applied to Scania highway scenes. The examples are from the same scene at different timestamps during a 2-second sequence ($t = 5.2, 6.2, 7.2$ s). Top: detection results on data with ego-motion compensation show three different failure modes. (1.a) shows an over-extended bounding box with incorrect orientation; (2.a) shows missing geometry due to distortion; (3.a) shows a failure in detection (no bounding box). Bottom: detection results on undistorted data processed by the HiMo pipeline. Note how the proposed data undistortion mitigates all three detection errors over the same 2s sequence. As shown in the figure, detection using HiMo-processed data produces compact, accurate, and consistent bounding boxes.

Nevertheless, given the increasing amount of data and the need for online undistortion in future scenarios, HiMo with SeFlow++ emerges as more advantageous in terms of runtime. Its deployment speed is nearly 100 times faster than that of NSFP and 20 times faster than FastNSF.

E. Downstream Effects

Our proposed HiMo pipeline is designed to correct motion-related geometric distortions in LiDAR point clouds. These distortions impact subsequent downstream tasks such as scene understanding, semantic segmentation, 3D detection and decision planning. As shown in Fig. 11, distortion can lead to visibly incorrect object shapes, duplicated contours and fragmented structures. These artifacts are not only misleading to human annotators but also degrade the performance of perception models, especially in high-speed driving scenarios.

Scene Understanding To quantify the impact of motion distortion on scene understanding, we evaluate the HiMo motion compensation method on the recent segmentation model WaffleIron [57], chosen due to its competitive performance and ease of deployment. The model is trained on KITTI - an urban dataset with mainly low-speed scenes and therefore low distortion. We then apply WaffleIron to the high-speed Argoverse 2 validation frames, comparing two input variants: (i) raw point clouds with baseline ego-motion compensation and (ii) point clouds corrected by HiMo. To ensure a fair evaluation across different sensor setups and annotation coverage, we report two sets of IoU scores. "All" reports the IoU over all points, whilst "Mask only" reports the IoU over points that fall into the labelled ground-truth bounding boxes. As shown in Table IV, HiMo consistently improves segmentation performance across both the CAR and OTHERS categories. In the "Mask Only" setting, the IoU for OTHERS increases by 3.95 points — a relative gain of over 12.5% — despite no fine-tuning of the segmentation model. These results demonstrate that HiMo improves object representation accuracy by preventing the segmentation model from misinterpreting fast-moving objects (e.g. cars), which might otherwise appear elongated due to motion distortions. It shows that even without fine-tuning, existing models benefit from inference on distortion-corrected data, highlighting the semantic consistency achieved through our correction.

TABLE IV: Segmentation IoU (%) on high-speed Argoverse 2 validation frames using WaffleIron [57]. We compare segmentations of raw point clouds with ego-motion compensation and point clouds corrected by HiMo. Results are reported for the CAR and OTHERS categories over all points and ground-truth-labeled regions only.

Input Point Cloud	Mask Only		All	
	CAR	OTHERS	CAR	OTHERS
w. Ego-motion Comp.	80.90	31.44	66.08	9.83
w. HiMo Motion Comp.	81.51	35.39	66.43	11.15

3D Detection We further analyze the impact of distortion on 3D object detection by applying TransFusion [56], a well-performing detector with publicly available NuScenes-pretrained weights. We use the implementation from OpenPCDet [58], making it easy to swap in other detectors in the codebase if desired. Again, we compare the object detection results using two inputs: the raw point clouds with baseline ego-motion compensation, and the undistorted point clouds produced by the HiMo pipeline. The comparison is shown

IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

in Figure 12 on the Scania dataset. Using the ego-motion compensated baseline point cloud as input (a), the pretrained detector exhibits three different failure modes even within a short 2-second sequence. This highlights how non-ego motion distortions in point clouds can cause significant degradation in detection quality. In contrast, with HiMo-corrected data as input (b), the same detector produces consistently aligned, compact, and correctly oriented bounding boxes, mitigating multiple downstream errors within this brief 2s sequence. These examples highlight how motion-induced distortions degrade the reliability of perception, especially in high-speed and multi-LiDAR settings, while HiMo offers a direct and effective correction strategy.

Decision-making and Planning Above, we have shown qualitative and quantitative results of HiMo compensation on downstream perception tasks. The errors introduced by these motion distortions, such as incorrect object shapes and duplicated positions, will also propagate into downstream decision-making and planning tasks that rely on correct 3D representations. These errors may lead to obstacle avoidance failures, unstable trajectory planning, or delayed decision-making. By restoring object geometry and motion consistency, HiMo can also enhance the reliability of planning systems built on top of it.

VII. Conclusion

In this paper, we addressed the critical issue of motion-induced distortions in LiDAR point clouds, which significantly impact the accuracy of environmental perception in autonomous driving systems. We analyzed the source of distortions and found that in addition to ego-motion, the motion of surrounding objects was a large source of distortion. Our investigation revealed the existence of moving object distortions across various datasets, particularly affecting high-speed objects and multi-LiDAR setups.

To tackle this challenge, we introduced HiMo, a novel pipeline that repurposes scene flow estimation for non-ego motion compensation. To further enhance motion estimation within HiMo, we also propose SeFlow++, a real-time self-supervised scene flow estimator that refines dynamic classification and symmetric loss, improving training efficiency and performance with smaller datasets. We demonstrated the effectiveness of HiMo through extensive experiments on our Scania highway and public datasets. These experiments compared the undistorted performance of HiMo with the ego-motion compensation baseline and evaluated the impact of different scene flow methods within HiMo. The results show that HiMo significantly improves point cloud representations, benefiting downstream tasks such as semantic segmentation and 3D detection, even without any fine-tuning. Our work contributes to the field by comprehensively analyzing point cloud distortions, proposing an effective compensation method, and offering open-source evaluation data and code.

Future work could explore the integration of HiMo with developing assistance 3D detection annotation systems as well as real-time perception systems.

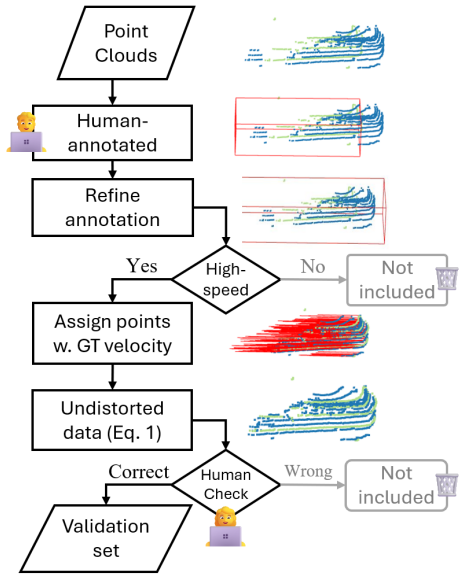


Fig. 13: Flowchart of the annotated ground-truth generation process. The process starts with manual annotation and velocity-based refinement of object bounding boxes. The velocities of these bounding boxes between frames are then used to undistort the raw data. Finally, we manually verify the ground-truth before including it in the validation set.

Acknowledgement

Thanks to Bogdan Timus and Magnus Granström from Scania and Ci Li from KTH RPL, who helped with this work. We also thank Yixi Cai, Yuxuan Liu, Peizheng Li and Shenghai Yuan for helpful discussions during revision. We thank the anonymous reviewers and the associate editor for their useful comments. The computations were enabled by the supercomputing resource Berzelius provided by National Supercomputer Centre at Linköping University and the Knut and Alice Wallenberg Foundation, Sweden.

Appendix

A. Annotated Ground Truth

Section V-B provides a detailed description of the ground-truth generating process for both the Scania and Argoverse 2 datasets. The process is also illustrated in the flowchart in Fig. 13. We also analyze how the annotation strategy, specifically the use of velocity-aware bounding box expansion, affects current scene flow evaluation in Appendix C.

B. Scene Flow Performance

To support our claim that SeFlow++ achieves state-of-the-art performance in real-time self-supervised scene flow estimation, we report its results on the Argoverse 2 public test set leaderboard [59]. As shown in Table V, SeFlow++ outperforms all other real-time methods, including SeFlow and ZeroFlow, in Three-way EPE mean metrics.

While several optimization-based methods (e.g., NSFP, EulerFlow, Floxels) achieve high accuracy, their runtimes span from several minutes to hours per sequence, making them unsuitable for real-time applications. In contrast, SeFlow++

offers a favorable trade-off between accuracy and efficiency, completing each sequence in just 8.2 seconds.

TABLE V: Scene flow performance comparisons on Argoverse 2 test set from the public leaderboard [59]. Our proposed SeFlow++ achieves state-of-the-art performance in real-time self-supervised scene flow estimation. Runtime is reported per sequence (around 157 frames), with '-' indicating unreported runtime. 's', 'm', and 'h' represent seconds, minutes, and hours, respectively. Red highlights the runtime of offline methods. For each method, we report its end point error (EPE) for foreground dynamic (FD), foreground static (FS), background static (BS) points, as well as the average EPE of all three point categories (three-way).

Methods	Runtime per seq	EPE (cm) ↓ Three-way	FD	FS	BS
Ego Motion Flow	-	18.13	53.35	1.03	0.00
FastNSF [39]	12m	11.18	16.34	8.14	9.07
NSFP [38]	1.0h	6.06	11.58	3.16	3.44
Floxels [60]	24m	**3.57**	7.73	**1.44**	**1.54**
EulerFlow [37]	24h	4.23	**4.98**	2.45	5.25
ZeroFlow [47]	5.4s	4.94	11.77	1.74	1.31
ICP-Flow [40]	-	6.50	13.69	3.32	2.50
SemanticFlow [61]	-	4.69	12.26	**1.41**	**0.40**
SeFlow [36]	7.2s	4.86	12.14	1.84	0.60
VoteFlow [62]	8.0s	4.60	12.14	1.84	0.60
SeFlow++ (Ours)	8.2s	**4.40**	**10.99**	1.44	0.79

C. Motion Compensation in Scene Flow Ground Truth

Originally, the Argoverse 2 dataset expands scene flow human-annotated bounding boxes by a fixed 20cm offset to account for distorted points [16], [63]. However, this method is ineffective in high-speed scenarios where the motion distortion can be significantly larger than 20cm. As a result, the labelled bounding boxes might not completely cover the entire object, leading to errors in the 3D scene flow labels. To mitigate such errors, following our HiMo pipeline insight, we expand the bounding boxes based on the relative velocity of dynamic objects, ensuring all relevant points are included.

Table VI shows the effect of this correction of groundtruth labels[3]. Compared to the GT corrected with fixed offset (20cm), all methods perform worse when evaluated using the GT corrected with our velocity-based procedure. The drop in performance is most visible in the FD category that captures errors in foreground dynamic points. This indicates that the previous ground truth fails to properly capture the dynamics of all foreground objects, especially those with high speed, leading to underestimated flow errors.

TABLE VI: Impact of ground truth correction process on the performance of three scene flow estimators. The original fixed bounding box expansion procedure ('Fixed (20cm)') led to missed points and inflated scores, particularly for dynamic objects. Our updated strategy with velocity-based bounding box expansion ('Velocity-based') ensures that all points belonging to the object are included, with flow values assigned, correcting the previously inflated scores. Pre-trained weights, as released by the original papers, are used for this evaluation.

Method	GT Correction	Three-way EPE (cm) ↓ Mean	FD	FS	BS
Flow4D [35]	Fixed (20cm)	3.09	6.98	1.17	1.11
	Velocity-based	3.80	8.88	1.66	0.86
DeFlow [33]	Fixed (20cm)	3.40	7.21	2.00	1.00
	Velocity-based	3.93	9.12	1.94	0.74
SeFlow [36]	Fixed (20cm)	5.84	13.83	2.24	1.45
	Velocity-based	6.29	15.56	2.16	1.16

[3]More detailed information can be found in https://github.com/KTH-RPL/OpenSceneFlow/pull/5 with visualization included.

References

[1] Y. Pan, X. Zhong, L. Wiesmann *et al.*, "PIN-SLAM: Lidar slam using a point-based implicit neural representation for achieving global map consistency," *IEEE Transactions on Robotics (TRO)*, 2024.

[2] X. Zhong, Y. Pan, J. Behley *et al.*, "Shine-mapping: Large-scale 3d mapping using sparse hierarchical implicit neural representations," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

[3] X. Zhong, Y. Pan, C. Stachniss *et al.*, "3D LiDAR Mapping in Dynamic Environments using a 4D Implicit Neural Representation," in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[4] Y. Cai, F. Kong, Y. Ren *et al.*, "Occupancy grid mapping without ray-casting for high-resolution lidar sensors," *IEEE Transactions on Robotics*, vol. 40, pp. 172–192, 2024.

[5] Y. Zhang, A. Ošep, L. Leal-Taixé *et al.*, "Zero-shot 4d lidar panoptic segmentation," in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 24 506–24 517.

[6] Y. Wang, Y. Chen, and Z.-X. Zhang, "4d unsupervised object discovery," *Advances in Neural Information Processing Systems*, vol. 35, pp. 35 563–35 575, 2022.

[7] Y. Li, S. Li, X. Liu *et al.*, "Sscbench: A large-scale 3d semantic scene completion benchmark for autonomous driving," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2024, pp. 13 333–13 340.

[8] J. Yang, B. Ivanovic, O. Litany *et al.*, "EmerneRF: Emergent spatial-temporal scene decomposition via self-supervision," in *The Twelfth International Conference on Learning Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=ycv2z8TYur

[9] M. Cao, S. Yang, Y. Yang *et al.*, "Rolling shutter correction with intermediate distortion flow estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[10] D. Qu, Y. Lao, Z. Wang *et al.*, "Towards nonlinear-motion-aware and occlusion-robust rolling shutter correction," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2023, pp. 10 680–10 688.

[11] I. MathWorks. (2024) Motion compensation in 3-d lidar point clouds using sensor fusion. [Online]. Available: https://se.mathworks.com/help/lidar/ug/motion-compensation-in-lidar-point-cloud.html

[12] W. Xu, Y. Cai, D. He *et al.*, "Fast-lio2: Fast direct lidar-inertial odometry," *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2053–2073, 2022.

[13] T.-M. Nguyen, D. Duberg, P. Jensfelt *et al.*, "Slict: Multi-input multi-scale surfel-based lidar-inertial continuous-time odometry and mapping," *IEEE Robotics and Automation Letters*, vol. 8, no. 4, pp. 2102–2109, 2023.

[14] J. Jiao, H. Ye, Y. Zhu *et al.*, "Robust odometry and mapping for multi-lidar systems with online extrinsic calibration," *IEEE Transactions on Robotics*, vol. 38, no. 1, pp. 351–371, 2021.

[15] A. Geiger, P. Lenz, C. Stiller *et al.*, "Vision meets robotics: The kitti dataset," *International Journal of Robotics Research*, 2013.

[16] M.-F. Chang, J. W. Lambert, P. Sangkloy *et al.*, "Argoverse: 3d tracking and forecasting with rich maps," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[17] B. Wilson, W. Qi, T. Agarwal *et al.*, "Argoverse 2: Next generation datasets for self-driving perception and forecasting," in *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, 2021.

[18] P. Sun, H. Kretzschmar, X. Dotiwalla *et al.*, "Scalability in perception for autonomous driving: Waymo open dataset," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[19] H. Caesar, V. Bankiti, A. H. Lang *et al.*, "nuscenes: A multimodal dataset for autonomous driving," in *CVPR*, 2020.

[20] J. Sun, Y. Dai, X. Zhang *et al.*, "Efficient spatial-temporal information fusion for lidar-based 3d moving object segmentation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2022, pp. 11 456–11 463.

[21] J. Cheng, K. Zeng, Z. Huang *et al.*, "Mf-mos: A motion-focused model for moving object segmentation," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 12 499–12 505.

[22] Z. Li, Y. Cui, J. Zhong *et al.*, "Streammos: Streaming moving object segmentation with multi-view perception and dual-span memory," *IEEE Robotics and Automation Letters*, vol. 10, no. 2, pp. 1146–1153, 2025.

[23] S. Vedula, P. Rander, R. Collins *et al.*, "Three-dimensional scene flow," *IEEE transactions on pattern analysis and machine intelligence*, vol. 27, no. 3, pp. 475–480, 2005.

[24] P. Jund, C. Sweeney, N. Abdo *et al.*, "Scalable scene flow from point clouds in the real world," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1589–1596, 2021.

[25] P. Sun, H. Kretzschmar, X. Dotiwalla *et al.*, "Scalability in perception for autonomous driving: Waymo open dataset," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[26] M. Alibeigi, W. Ljungbergh, A. Tonderski *et al.*, "Zenseact open dataset: A large-scale and diverse multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

[27] N. E. Chodosh, A. Madan, S. Lucey *et al.*, "SMORE: Simultaneous map and object REconstruction," in *International Conference on 3D Vision 2025*, 2025. [Online]. Available: https://openreview.net/forum?id=1NhnG9BvQB

[28] C. Jiang, G. Wang, J. Liu *et al.*, "3dsflabelling: Boosting 3d scene flow estimation by pseudo auto-labelling," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 15 173–15 183.

[29] I. Khatri, K. Vedder, N. Peri *et al.*, "I can't believe it's not scene flow!" in *European Conference on Computer Vision*. Springer, 2024, pp. 242–257.

[30] Y. Zhang, J. Edstedt, B. Wandt *et al.*, "Gmsf: Global matching scene flow," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[31] J. Liu, G. Wang, W. Ye *et al.*, "Difflow3d: Toward robust uncertainty-aware scene flow estimation with iterative diffusion-based refinement," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 15 109–15 119.

[32] C. Wang, L. E. MacDonald, L. A. Jeni *et al.*, "Flow supervision for deformable nerf," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2023, pp. 21 128–21 137.

[33] Q. Zhang, Y. Yang, H. Fang *et al.*, "DeFlow: Decoder of scene flow network in autonomous driving," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 2105–2111.

[34] A. Khoche, Q. Zhang, L. P. Sánchez *et al.*, "SSF: Sparse long-range scene flow for autonomous driving," in *2025 IEEE International Conference on Robotics and Automation*, 2025, pp. 6394–6400.

[35] J. Kim, J. Woo, U. Shin *et al.*, "Flow4D: Leveraging 4d voxel network for lidar scene flow estimation," *IEEE Robotics and Automation Letters*, pp. 1–8, 2025.

[36] Q. Zhang, Y. Yang, P. Li *et al.*, "SeFlow: A self-supervised scene flow method in autonomous driving," in *European Conference on Computer Vision (ECCV)*. Springer, 2024, p. 353–369.

[37] K. Vedder, N. Peri, I. Khatri *et al.*, "Neural eulerian scene flow fields," in *The Thirteenth International Conference on Learning Representations*, 2025. [Online]. Available: https://openreview.net/forum?id=0CieWy9ONY

[38] X. Li, J. Kaesemodel Pontes, and S. Lucey, "Neural scene flow prior," *Advances in Neural Information Processing Systems*, vol. 34, pp. 7838–7851, 2021.

[39] X. Li, J. Zheng, F. Ferroni *et al.*, "Fast neural scene flow," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 9878–9890.

[40] Y. Lin and H. Caesar, "Icp-flow: Lidar scene flow estimation with icp," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 15 501–15 511.

[41] P. Vacek, D. Hurych, K. Zimmermann *et al.*, "Let-it-flow: Simultaneous optimization of 3d flow and object clustering," *IEEE Transactions on Intelligent Vehicles*, pp. 1–10, 2024.

[42] R. Ahuja, C. Baker, and W. Schwarting, "Optflow: Fast optimization-based scene flow estimation without supervision," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2024, pp. 3161–3170.

[43] L. Zhu, Y. Jia, S. Huang *et al.*, "Deflow: Self-supervised 3d motion estimation of debris flow," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 6508–6517.

[44] A. H. Lang, S. Vora, H. Caesar *et al.*, "Pointpillars: Fast encoders for object detection from point clouds," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 12 697–12 705.

[45] K. Cho, B. Van Merriënboer, C. Gulcehre *et al.*, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 2014, pp. 1724–1734. [Online]. Available: https://aclanthology.org/D14-1179/

[46] Y. Wei, Z. Wang, Y. Rao *et al.*, "Pv-raft: Point-voxel correlation fields for scene flow estimation of point clouds," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 6954–6963.

[47] K. Vedder, N. Peri, N. Chodosh *et al.*, "ZeroFlow: Fast Zero Label Scene Flow via Distillation," *International Conference on Learning Representations (ICLR)*, 2024.

[48] M. Asad, R. Dorent, and T. Vercauteren, "Fastgeodis: Fast generalised geodesic distance transform," *Journal of Open Source Software*, vol. 7, no. 79, p. 4532, 2022. [Online]. Available: https://doi.org/10.21105/joss.04532

[49] D. Duberg, Q. Zhang, M. Jia *et al.*, "DUFOMap: Efficient dynamic awareness mapping," *IEEE Robotics and Automation Letters*, vol. 9, no. 6, pp. 5038–5045, 2024.

[50] R. J. Campello, D. Moulavi, and J. Sander, "Density-based clustering based on hierarchical density estimates," in *Pacific-Asia conference on knowledge discovery and data mining*. Springer, 2013, pp. 160–172.

[51] M. Najibi, J. Ji, Y. Zhou *et al.*, "Motion inspired unsupervised perception and prediction in autonomous driving," in *European Conference on Computer Vision*. Springer, 2022, pp. 424–443.

[52] H. Mittal, B. Okorn, and D. Held, "Just go with the flow: Self-supervised scene flow estimation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 177–11 185.

[53] M. Himmelsbach, F. V. Hundelshausen, and H.-J. Wuensche, "Fast segmentation of 3d point clouds for ground vehicles," in *Intelligent Vehicles Symposium (IV), 2010 IEEE*. IEEE, 2010, pp. 560–565.

[54] A. Khoche, A. Asefaw, A. González *et al.*, "Addressing data annotation challenges in multiple sensors: A solution for scania collected datasets," in *2024 European Control Conference*. IEEE, 2024, pp. 1032–1038.

[55] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, San Diega, CA, USA, 2015.

[56] X. Bai, Z. Hu, X. Zhu *et al.*, "Transfusion: Robust lidar-camera fusion for 3d object detection with transformers," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 1090–1099.

[57] G. Puy, A. Boulch, and R. Marlet, "Using a waffle iron for automotive point cloud semantic segmentation," in *ICCV*, 2023.

[58] O. D. Team, "Openpcdet: An open-source toolbox for 3d object detection from point clouds," https://github.com/open-mmlab/OpenPCDet, 2020.

[59] A. 2, "Argoverse 2 scene flow online leaderboard," https://eval.ai/web/challenges/challenge-page/2210/leaderboard/5463, 2025 Apr 25th.

[60] D. T. Hoffmann, S. H. Raza, H. Jiang *et al.*, "Floxels: Fast unsupervised voxel based scene flow estimation," in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 22 328–22 337.

[61] Y. Chen, M. Zhang, Q. Hao *et al.*, "SemanticFlow: A self-supervised framework for joint scene flow prediction and instance segmentation in dynamic environments," *arXiv preprint arXiv:2503.14837*, 2025.

[62] Y. Lin, S. Wang, L. Nan *et al.*, "Voteflow: Enforcing local rigidity in self-supervised scene flow," in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 17 155–17 164.

[63] N. Chodosh, D. Ramanan, and S. Lucey, "Re-evaluating lidar scene flow," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2024, pp. 6005–6015.