**ContrastMotion: Self-supervised Scene Motion Learning for Large-Scale LiDAR Point Clouds**

Xiangze Jia\textsuperscript{1}, Hui Zhou\textsuperscript{2}, Xinge Zhu\textsuperscript{2}, Yandong Guo\textsuperscript{3}, Ji Zhang\textsuperscript{1,4*}, Yuexin Ma\textsuperscript{5*}

\textsuperscript{1}Nanjing University of Aeronautics and Astronautics
\textsuperscript{2}The Chinese University of Hong Kong
\textsuperscript{3}OPPO Research Institute
\textsuperscript{4}Zhejiang Lab
\textsuperscript{5}ShanghaiTech University

jxzxiao5@nuaa.edu.cn, ji_zhang_nuaa@gmail.com, mayuexin@shanghaitech.edu.cn

**Abstract**

In this paper, we propose a novel self-supervised motion estimator for LiDAR-based autonomous driving via BEV representation. Different from usually adopted self-supervised strategies for data-level structure consistency, we predict scene motion via feature-level consistency between pillars in consecutive frames, which can eliminate the effect caused by noise points and view-changing point clouds in dynamic scenes. Specifically, we propose *Soft Discriminative Loss* that provides the network with more pseudo-supervised signals to learn discriminative and robust features in a contrastive learning manner. We also propose *Gated Multi-frame Fusion* block that learns valid compensation between point cloud frames automatically to enhance feature extraction. Finally, *pillar association* is proposed to predict pillar correspondence probabilities based on feature distance, and whereby further predicts scene motion. Extensive experiments show the effectiveness and superiority of our *ContrastMotion* on both scene flow and motion prediction tasks.

**1 Introduction**

For autonomous vehicles, it is critical to understand the dynamic cues in large-scale scenarios, e.g. distinguishing movable and static objects in the environment, to avoid obstacles and guarantee the safety [Mahjourian et al., 2018; Luo et al., 2018]. Due to the advantage of long-range depth capture, LiDAR has become one popular sensor for scene perception [Cong et al., 2022; Yin et al., 2021; Zhu et al., 2020; Zhu et al., 2021] and boosted the development of autonomous driving. Extracting motion features from LiDAR point cloud has attracted more and more attention in recent years, which is a fundamental technique for many downstream tasks, such as detection, segmentation, trajectory prediction, navigation, etc.

However, 3D labeling on large-scale point clouds requires complex processes and labor-intensive human efforts. Unlike the regular pixel representation of images, LiDAR point cloud is sparsity-varying and unordered in one scene and the discrepancy is more obvious for cross-scenes and cross-LiDAR captured data, leading to the situation that supervised methods are difficult to adapt to other domains. Considering above limitations, we focus on the self-supervised learning strategy to extract the motion pattern existing in consecutive frames of LiDAR point cloud.

Scene motion is a fine-grained representation of the motion pattern, which is defined as a 3D displacement vector between each point in two sequential frames [Wu et al., 2019]. There are already some self-supervised approaches [Gu et al., 2022; Baur et al., 2021; Li et al., 2022b] for scene flow estimation. However, most of them rely on point-wise feature extraction and correlation corresponding, which bring...
heavy costs of memory and computation for large-scale point clouds. Meanwhile, the data-level structure consistency is not applicable for dynamic scenes, because LiDAR is viewpoint-dependent and will capture different views of objects in different frames, causing many points having no correspondences in adjacent frames and leading to low accuracy for the motion estimation. These properties limit the deployment of such methods in real scenes, which is highly dynamic and requires high-efficiency point cloud processing.

Instead of dense motion prediction, researchers tend to explore the motion representation on Bird’s Eye View (BEV). Due to the rich semantic and geometry information existing in the BEV representation, especially for the traffic scenarios, BEV provides a trade-off choice for accurate and efficient perception in large-scale scenes [Ma et al., 2022]. Moreover, BEV representation can, to some extent, eliminate the effect caused by unordered and noisy points by discretizing the point cloud into BEV grid cells. Such compact representation is friendly for defining the search window for point matching. Thus, based on one popular BEV [Li et al., 2022c] and pillar [Lang et al., 2019] representation, [Luo et al., 2021; Baur et al., 2021] propose effective motion estimators, which are more applicable for autonomous driving. However, the self-supervised architecture is similar to previous scene flow methods by mainly using the point-level structure consistency and regularization between consecutive frames of point cloud for self-supervision. Such methods, depending on data-level correspondences, still suffers from dynamic changes of scenes.

In this paper, based on the pillar representation, we propose a novel self-supervised scene motion estimator for LiDAR captured large-scale point clouds, which has solved above problems. To reduce the effect of point-level correspondences missing during the view-dependent scanning in dynamic scenes, we extract the high-level feature representation of each pillar and then predict pillar corresponding probabilities between consecutive frames (Fig 1). After obtaining the pillar correspondences, we easily calculate the motion information of the whole scene. Specifically, we utilize contrastive learning to bootstrap the network to extract discriminative features of each pillar, where the positive pillar pairs are two pillars that have a corresponding relationship in consecutive frames. To acquire positive pairs without any labels, we leverage commonly used data augmentation methods to generate two frames of point clouds from the same sample and record the pillar correspondences according to the transformation matrix. Although the hard one-to-one positive pillar assignment increases the discriminability of features, it provides less supervision and makes it difficult for the network to learn strong and robust features. Therefore, we propose Soft Discriminative Loss (SD–Loss), which generates positive pillar pairs with a soft constraint to further provide a margin of fault tolerance in the dynamic scenes. We also propose Gated Multi-frame Fusion (GMF) block learning valid compensation to enhance feature consistency of corresponding pillars, where the gate status also provides dynamic and static information about the scene. In pillar association, we predict the pillar correspondence probabilities mainly through the feature distance of pillars between consecutive frames. Moreover, in order to reduce the complexity of pillar association from quadratic of the number of pillars to linear, we adopt Patching and Asymmetric Patch strategy.

We validate our ContrastMotion on the KITTI Scene Flow dataset and nuScenes dataset and achieve state-of-the-art performance for both scene flow and motion prediction tasks. We also conduct ablation studies to illustrate the effectiveness of the novel designs in our method.

The contributions of our work can be summarized as follows:

- We propose a novel pillar-based self-supervised motion estimator for LiDAR point cloud, which predicts pillar corresponding probabilities between point cloud pair by feature distance, and further acquires scene motion.
- We propose Soft Discriminative Loss (SD–Loss) to provide the network with more pseudo-supervised signals to learn discriminative and robust features, and it can also tackle the case of correspondence points missing in real scenes.
- We propose Gated Multi-frame Fusion (GMF) block learning valid compensation between point cloud pairs automatically to enhance feature consistency of corresponding pillars, which also has the capability of perceiving dynamic-static information of the scene.
- We analyze the degradation solution, and our ContrastMotion has achieved state-of-the-art performance on both scene flow estimation and motion prediction tasks.

2 Related Work

2.1 Supervised Motion Learning

This task aims to estimate motions and predict the future locations of various agents via past information [Luo et al., 2021]. The previous works [Ma et al., 2019; Sadeghian et al., 2019] formulate this task as a trajectory problem based on 3D object detection and tracking. Another active research line [Wu et al., 2020; Lee et al., 2020; Wei et al., 2021; Li et al., 2022a; Wang et al., 2022] estimate the motions in an end-to-end framework. However, 3D labeling on large-scale point clouds requires complex processes and labor-intensive human efforts, and there are significant discrepancies between cross-scenes and cross LiDAR captured data.

2.2 Self Supervised Motion Learning

In the field of the point cloud, There are already self-supervised [Wu et al., 2019; Mittal et al., 2020; Kittenplon et al., 2021; Li et al., 2021a; Hur and Roth, 2021; Kittenplon et al., 2021; Wu et al., 2019] and non-learning [Li et al., 2021b] methods for motion estimation. PointPWC [Wu et al., 2019] estimates the motions with a coarse-to-fine paradigm and introduces the losses for scene consistency, smoothness and shape approximation. However, these methods perform significant sampling in the training and inference pipelines due to the heavy costs of memory and computation. Instead of dense motion, researchers tend to explore the motion in pillar representation. PillarMotion [Luo et al.,
2021] introduces structural consistency augmented with motion masking and a cross-sensor (LiDAR and camera) regularization to predict the motion of pillars. SLIM [Baur et al., 2021] proposes a self-supervised model for motion estimation and segmentation in a BEV map. However, these methods are similar to previous scene flow methods by mainly using the point-level structure consistency and regularization between consecutive frames, which suffer from the dynamic changes of scenes.

### 2.3 Contrastive Learning

Contrastive learning [Radford et al., 2021; Bai et al., 2022] has been widely used due to the great potential in pre-training and feature correspondence learning. In the point cloud field, PointContrast [Xie et al., 2020] proposes a pre-trained method for learning representations via matching corresponding points in different views. Specifically, it establishes the connections of corresponding points in different perspectives with the proposed PointInfoNCE to enhance the representation of the model. However, the pretext task only involves point-level correspondence matching which completely ignores the spatial configuration and context in the scene [Hou et al., 2021], leading to a limited performance in motion tasks.

### 3 Method

#### 3.1 Overview

We represent the point cloud at time \( t \) as \( \mathcal{PC}_t = \{p^t_i\}_{i=1}^{N_t} \), where \( p^t_i \) represents one point and \( N_t \) represents the number of points. Considering the efficiency requirement in dealing with large-scale point cloud for autonomous driving, the encoder of our ContrastMotion is based on the popular pillar representation. The point cloud \( \mathcal{PC}_t \) are discretized into pillars \( \{P^t_i\}_{i=1}^{N_p} \), where \( N_p \) is the number of pillars.

As shown in Figure 2, our ContrastMotion is in a contrastive learning mechanism. In training pipeline, We leverage commonly used data augmentation methods to generate \( \mathcal{PC}_t \) and \( \mathcal{PC}_{t+1} \) from the same point cloud sample, learn the representations of \( \mathcal{PC}_t, \mathcal{PC}_{t+1} \) organized in pillars with PFE [Qi et al., 2017], and then feed them to our Encoder which is similar to backbone in FastFlow3D [Jund et al., 2021], respectively. After that, The Gated Multi-frame Fusion(GMF) block learn valid compensation to enhance feature consistency of corresponding pillars, and its outputs are two flattened feature maps of \( N_p \times D \), where \( N_p \) is the number of pillars and \( D \) denotes feature dimension. We then predict pillar corresponding probabilities based on feature distance between pillars in point cloud pair. The errors between predicted pillar corresponding probabilities and ground truth pillar correspondences are taken as the loss to optimize parameters. Given consecutive frames during inference, ContrastMotion generates pillar correspondences based on pillar corresponding probabilities, and further scene motion.

#### 3.2 Contrastive Learning

**Self-Contrast and Cross-Contrast**

Different from the methods with point-level correspondences and structural consistency suffering from dynamic changes of scenes, we learn the motion pattern via feature-level consistency. To learn the discriminative features distinguishing each pillar from others and feature consistency between corresponding pillars, we formulate the problem as a one-to-\( N_p \) classification task in contrastive learning, and propose Self-contrast and Cross-contrast.

Specifically, Self-contrast and Cross-contrast are used for learning the discriminative features of different pillars in the same frame and feature consistency of corresponding pillars in consecutive frames, respectively. For the former, each pillar forms a positive pair with itself only and forms the negative pairs with all others in the same frame. For the latter, since there are no pillar correspondence labels for consecutive frames in self-supervised training, we leverage commonly used data augmentation methods to generate \( \mathcal{PC}_t \) and \( \mathcal{PC}_{t+1} \).
from the same point cloud training sample, including random rotation, shift, scaling, and jitter. All the data augmentation transformations are applied to the entire point clouds to ensure the completeness of the object surfaces. Then we acquire pillar correspondences based on the known transformation matrices in data augmentation, where each pillar $P_i^t$ forms a positive pair with its corresponding pillar $P_{i'}^{t+1}$ and negative pairs with all other pillars $P_{j'}^{t+1}$ where $j \neq i'$. After that, the feature consistency of corresponding pillars can be learned by Cross-contrast in a contrastive learning manner. In addition, we apply the random removal to the points in generated $\mathcal{P}C$ to simulate the change in the appearance of the objects caused by ego-motion to avoid over-fitting. Self-contrast and Cross-contrast learn the discriminative and consistent pillar features on which pillar corresponding probabilities depend, which reduces the effect of noise points and point correspondences missing in dynamic scenes.

### Soft Discriminative Loss

As mentioned above, the training pipeline of Self- and Cross-contrast is considered a contrastive learning manner. For unsupervised contrastive learning, InfoNCE [Oord et al., 2018] and PointInfoNCE proposed by PointContrast [Xie et al., 2020] are widely used recently. Specifically, they encourage one query to be similar to one positive key and dissimilar to, typically many, negative keys. However, the one-to-one correspondence is not applicable for dynamic scenes, because LiDAR is view-dependent and will capture different views of objects in different frames, causing many points having no correspondences in adjacent frames. Moreover, most points of one rigid object have similar motion and contextual geometric information. Therefore, we generate positive keys with a soft constraint and propose soft discriminative loss (SD-Loss).

$$\mathcal{L} = -\sum_t \sum_{i,j \in V(P_i^t)} w_{ij} \log \frac{\exp(\sigma(P_i^t) \cdot \sigma(P_j^{t+1}))}{\sum_k \exp(\sigma(P_i^t) \cdot \sigma(P_k^{t+1}))}$$

(1)

Here $\cdot$ denotes vector inner product. $V(P_i^t) = \{P_j^{t+1} \mid d(P_j^{t+1}, P_{i'}^{t+1}) < \varepsilon\}$ represents the positive keys of $P_i^t$, where $d(\cdot, \cdot)$ is the Euclidean distance function, $P_i^t$ is determined by the transformation matrix and the corresponding pillar of $P_i^t$, and $\varepsilon$ represents the distance threshold. $\sigma(x)$ is the pillar features extracted by the networks, where $ReLU$ guarantees a positive feature vector, and $w_{ij}$ represents the weight of each positive key, which has a negative correlation with the distance between pillar $P_j^{t+1}$ and $P_{i'}^{t+1}$. For each pillar $P_i^t$, we generate the weighted positive keys with not only the corresponding pillar $P_{i'}^{t+1}$ defined by transformation matrix but also the neighbouring pillars of $P_{i'}^{t+1}$. Compared to the PointInfoNCE, our SD-Loss retains the feature similarities of local pillars and provides a margin of fault tolerance to avoid the destruction of changing views of point clouds in dynamic scene. Moreover, SD-Loss tackles a widely observed challenge that one-to-one positive key assignment provides less supervision, making it difficult for the network to learn strong and robust features.

### 3.3 Gated Multi-Frame Fusion Block

Adjacent frames can provide additional information to enhance the features of current sparse point cloud. However, naive fusion of multiple frames makes the moving objects trailing and makes the pillar association which relies heavily on the feature consistency of pillars ambiguous. Therefore, we propose Gated Multi-frame Fusion (GMF) block to learn the valid complement features from adjacent frames, which consists of a gated network. After the Encoder extract the features $F_t$ and $F_{t+1}$, we align two feature maps according to ego-motion to make the features at the same grid correspond to the same real-world location. Considering a static scene, the fusion feature can be denoted as $F_t + F_{t+1}$ due to warping ego-motion between consecutive frames. To avoid wrong fusion caused by moving objects in real scene, we introduce pillar description maps. We feed $F_t$ and $F_{t+1}$ to gated network consisting of two convolutional layers, followed activation functions, a linear layer and Sigmoid to get pillar description maps $m_t, m_{t+1}$, respectively.

$$m = \delta(\text{linear}(conv(F)))$$

(2)

where $\delta$ represents Sigmoid to generate pillar description between 0 and 1. Then, the results of multiplying description maps with the original feature maps are added to another original feature maps for feature enhancement.

$$z_t = F_t + m_{t+1} \cdot F_{t+1}$$

(3)

$$z_{t+1} = F_{t+1} + m_t \cdot F_t$$

(4)

where $\cdot$ denotes elemental multiplication, and $z$ represents the modulated feature map. GMF is straightforward but adaptively distinguishes between static and dynamic pillars and achieves correct feature fusion. As a result, the feature maps $z$ contain much more geometric features than $F$, even if the relevant point is missing. GMF facilitates the extraction of discriminative features of the scene due to feature compensation between multiple frames and is beneficial for pillar association due to the enhanced feature consistency between corresponding pillars. Even without additional supervised signals, the pillar description maps actually embodies the dynamic-static information of the pillars and provides more information for the feature extraction, which is introduced in the ablation experiments.

### 3.4 Pillar Association

After extracting the feature representation of each pillar, we implement pillar association to predict pillar correspondences $\{(P_i^t, P_{i'}^{t+1})\}_{i=1}^{N_p}$ using feature distance. Specifically, the pillar association is in a query-key manner where $query$ is the feature of $P_i^t$ and $keys$ are features of $\{P_{j'}^{t+1}\}_{j'=1}^{N_p}$. However, the cost of calculating the feature distance between query and all keys is expensive. Therefore, we adopt Patching strategy, where the pseudo-image composed of pillars is partitioned into several $s \times s$ patches, named query patches $\{Q_i^t\}_{i=1}^{N_p}$. For each query patch $Q_i^t$, there exists a corresponding key patch $K_{i'}^{t+1}$ whose patch size is $\alpha s \times \alpha s$ where $\alpha > 1$, and the central pillars of the corresponding patches have the same row and column numbers. The pillars in each query patch
only calculate the feature distance with the keys in the corresponding key path $K_{t+1}$ and then predict pillar correspondence probability. We define pillar correspondence probability between $P_i^t$ and $P_j^{t+1}$ as

$$p_b(i, j) = \frac{\exp(z(P_i^t) \ast z(P_j^{t+1})))}{\sum_{k \in k^{t+1}} exp(z(P_i^t) \ast z(P_k^{t+1})))}$$

where $z(P)$ represents pillar features, $\ast$ denotes vector inner product. We identify the one with the maximum correspondence probability with $P_i^t$ as its corresponding pillar $P_j^{t+1}$. The adoption of asymmetric patch sizes ensures that queries at the border of query patch still have sufficient keys to implement pillar association. In summary, We adopt Patching to achieve greater efficiency and Asymmetric Patch to ensure its accuracy. After acquiring the pillar correspondences $\{(P_i^t, P_j^{t+1})\}_{i=1}^{N}$, we calculate the pillar flow of $P_i^t$ through $c(P_i^{t+1}) - c(P_i^t)$ where $c(P)$ represents the center of pillars in the real-world coordinate system.

3.5 Task-Specific Post-Processing

Scene Flow

Scene flow is defined as a 3D displacement vector between each surface point rather than each pillar, we directly scatter the flow of pillars to points.

Motion Prediction

Generally, the related works [Wu et al., 2020; Luo et al., 2021] predict future motion with the scene motions from the past by assuming consistent velocities and accelerations. ContrastMotion estimates the motions with past frames and converts them into velocities, then predict future motion.

3.6 Analysis

Compared to point cloud registration which estimates the transformation matrix between point cloud pairs, scene motion estimates fine-grained motion and is more complex. From the perspective of training samples, our ContrastMotion looks like addressing registration rather than scene motion. Because we utilize data augmentation to generate a pair of point clouds as training sample, and the data augmentation transformation is applied to all points in the point cloud in order to ensure the integrity of the object surface. Does the degradation solution occur during training, where the network infers the transformation matrix and obtains the motion of each pillar directly, making the model much less capable of inferring scene motion? The reason why ContrastMotion is working relies on the fact that we use a pillar association process instead of the conventional prediction head. In pillar association, we use a query-key manner and for each key we have no additional positional encoding. The distinction between the different pillars(keys) relies on their geometric contextual information rather than on their real position. In pillar association process, the network is agnostic to pillar motion, which avoids the degenerate solution. Thus, even with the training samples generated from the transformation matrix, our model still achieves good performance in real scenes.

<table>
<thead>
<tr>
<th>Method</th>
<th>EPE3D$\downarrow$</th>
<th>Acc3DR$\uparrow$</th>
<th>Acc3DS$\uparrow$</th>
<th>Outliers3D$\downarrow$</th>
</tr>
</thead>
<tbody>
<tr>
<td>PointPWC</td>
<td>0.3648</td>
<td>0.0726</td>
<td>0.2974</td>
<td>0.8579</td>
</tr>
<tr>
<td>PoseFlow</td>
<td>0.3256</td>
<td>0.1104</td>
<td>0.2058</td>
<td>0.9778</td>
</tr>
<tr>
<td>SLIM$^+$</td>
<td>0.0688</td>
<td>0.7095</td>
<td>0.9342</td>
<td>0.2488</td>
</tr>
<tr>
<td>FlowStep3D</td>
<td>0.1741</td>
<td>0.5821</td>
<td>0.7359</td>
<td>0.3596</td>
</tr>
<tr>
<td>RCP$^+$</td>
<td>0.0974</td>
<td>0.6058</td>
<td>0.8021</td>
<td>0.3263</td>
</tr>
<tr>
<td>ContrastMotion</td>
<td>0.0656</td>
<td>0.7382</td>
<td>0.9417</td>
<td>0.1847</td>
</tr>
</tbody>
</table>

Table 1: Evaluation results on FT3D $\rightarrow$ KITTI Scene Flow. These self-supervised methods are trained only on FT3D and evaluated on KITTI Scene Flow dataset for all points. PointPWC, FlowStep3D and RCP are the point-based methods, while SLIM and our ContrastMotion are pillar-based. $^+$: models trained on KITTI-RL. $^\dagger$: reproducing experiments. $^\dagger$: 8192 points downsampling.

<table>
<thead>
<tr>
<th>Method</th>
<th>Moving EPE3D(m)</th>
<th>Stat. EPE3D(m)</th>
<th>Moving Acc3DR</th>
<th>Stat. Acc3DR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zero</td>
<td>0.6381</td>
<td>0.1632</td>
<td>0.5248</td>
<td></td>
</tr>
<tr>
<td>PointPWC</td>
<td>0.3539</td>
<td>0.2543</td>
<td>0.1974</td>
<td></td>
</tr>
<tr>
<td>PoseFlow</td>
<td>0.7399</td>
<td>0.0000</td>
<td>0.0570</td>
<td></td>
</tr>
<tr>
<td>SLIM</td>
<td>0.1050</td>
<td>0.7365</td>
<td>0.0925</td>
<td></td>
</tr>
<tr>
<td>ContrastMotion(ours)</td>
<td>0.0925</td>
<td>0.7082</td>
<td>0.0545</td>
<td></td>
</tr>
</tbody>
</table>

Table 2: Self-supervised training & evaluation on nuScenes scene flow. Zero predicts that the environment is static.

4 Experiments

4.1 Dataset

KITTI Scene Flow [Menze et al., 2015; Menze et al., 2018] consists of 200 training scenes and 200 test scenes. Point clouds and ground truth scene flows are obtained by lifting the annotated disparity maps and optical flows to 3D. Following previous works [Gu et al., 2019; Wu et al., 2019], FlyingThings 3D (FT3D) dataset, a synthetic dataset with 19640 samples, is used for training and the training scenes in KITTI are used for inference.

nuScenes [Caesar et al., 2019] is a large-scale public dataset for autonomous driving collected from 1000 real scenes, whose initial purpose is object detection and segmentation. The point clouds are captured with a frequency of 20Hz, and [Caesar et al., 2019] annotates the entire dataset with accurate 3D bounding boxes for 23 object classes at 2Hz. Following previous works [Wu et al., 2020; Jund et al., 2021; Li et al., 2022b], we assume that each object is in rigid motion and derive the ground truth motion from the origin detection and tracking annotations.

4.2 Experimental Setup

Implementation Details

We implement all the experiments on PyTorch [Paszke et al., 2019]. For FT3D and nuScenes datasets, we train the ContrastMotion for 300 and 100 epochs respectively, and set the initial learning rate to 0.001, weight decay to 0.001. The batch size is set to the maximum available on a 32G GPU, and Adam is used as the optimizer. We crop the point clouds following baseline models [Li et al., 2022b; Luo et al., 2021], and the pillar size is set to $[0.25, 0.25]$. For training $\mathcal{P}_C$ generation, the upper limits of shift, rotation angle, scaling and jitter are $3m, 0.17, 1.05$ and $0.1m$. We scale the samples of FT3D to fit the cropping range during training. In KITTI, we predict the motion of points beyond crop-
Table 3: Comparison with the state-of-the-art results on the nuScenes motion prediction. We report the mean and median errors on the three speed groups. Self indicates self-supervised methods trained with nuScenes, and Pre. indicates the methods that are not trained with the annotations of nuScenes but are supervised pre-trained on two scene flow datasets. Zero predicts that the environment is static.

Table 4: Ablation study for different modules of ContrastMotion. We evaluate each model on nuScenes motion prediction and report mean errors. Ground Mask: Removing estimated ground points before feeding to network. Gated Block: Gated Multi-frame Fusion Block.

Evaluation Metrics
We report the evaluation criteria used by Hplflownet [Gu et al., 2019] in scene flow: EPE3D(m) average end-point-error over each point; Acc3DS point ratio where EPE3D < 0.05 or relative error < 5%; Acc3DR point ratio where EPE3D < 0.1 or relative error < 10%; Outliers3D point ratio where EPE3D > 0.3 or relative error > 10%.

We report the average L2 distances and median errors on non-empty pillars which are classified as static, slow and fast groups in motion prediction, which used in PillarMotion.

4.3 Comparison with SOTA Methods

Scene Flow
We implement the experiments on FT3D → KITTI-SF, and mainly compare our ContrastMotion with the point-based methods, including PointPWC [Wu et al., 2019], FlowStep3D [Kittenplon et al., 2021], PoseFlow [Tishchenko et al., 2020] and RCP [Gu et al., 2022], and pillar-based method SLIM [Li et al., 2022b]. Our ContrastMotion and SLIM can process a full scene from KITTI. While the baselines can run inference on a full scene, during training it is infeasible to run them with roughly 30K points due to the cost of memory and the training time [Li et al., 2022b].

As shown in Table 1, the point-based methods, e.g., PointPWC and RCP, achieve limited results compared with the pillar-based approaches. These methods perform significant sampling in the training pipelines due to the heavy cost of memory and computation for large-scale point clouds, while pillar-based methods can be trained with the raw data. The increased points providing more fine-grained information enhance the stable inference ability on the dense point clouds. In Table 1, we also report the performance of a pillar-based method, SLIM. Although trained with KITTI-RL [Geiger et al., 2013], the performance of SLIM is not satisfying, which is due to the effect of point-level correspondence problems in the dynamic scenes. Our ContrastMotion leverages feature-level consistency to predict pillar flows and achieves SOTA performances in EPE3D and Outliers3D metrics. The results show the effectiveness of pillar-based methods and our ContrastMotion.

In order to get closer to real-world evaluation, we also implement self-supervised experiments on the dataset without point correspondences, nuScenes, and compare our ContrastMotion with SLIM [Li et al., 2022b]. As shown in Table 2, Our ContrastMotion achieves the SOTA performance in moving EPE3D. Where around 95% points in nuScenes are static. Our proposed GMF enhances the feature consistency between the corresponding static pillars and helps our ContrastMotion achieve the best performance compared to the baseline methods in static EPE3D.

Motion Prediction
As same as PillarMotion [Luo et al., 2021], we use the same 500 training scenes, 100 validation scenes and 250 testing scenes with PillarMotion [Luo et al., 2021]. But unlike PillarMotion which uses LiDAR, camera and pose data,
only LiDAR and pose data are used in our work. The related methods we compare include self-supervised PillarMotion [Luo et al., 2021], and pre-trained supervised estimators on other datasets, e.g., FlowNet3D [Liu et al., 2019] and HPLFlowNet [Gu et al., 2019].

Table 3 shows the results of different approaches. The performances of FlowNet3D and HPLFlowNet which are trained in FT3D and tested on nuScenes without fine-tuning indicate the existence of a large domain gap. In addition, 3D labeling on real larges-scale scenes requires complex processes and labor-intensive human efforts, and the discrepancy is obvious for cross-scenes and cross-LiDAR captured data. PillarMotion achieves a limited performance with consistency and cross-device regularization, which is also due to the effect of point-level correspondence missing in the dynamic scenes. In addition, the optical flow used as a signal to guide motion learning introduces errors, including optical flow prediction and coordinate transformation errors. Our ContrastMotion achieves SOTA performances among the compared methods in the low and high speed groups, and the prediction errors decrease by 35% and 0.02 compared to PillarMotion. In terms of median errors, our ContrastMotion achieves SOTA performances in all the groups. The results show the advantages of self-supervision compared with supervision and the potential of ContrastMotion in motion prediction tasks.

4.4 Ablation Studies

Gated Multi-Frame Fusion Block

We perform experiments to explore the effects of proposed GMF, and the results are shown in Table 4. The baseline model (a) trained only with our SD–Loss does not perform well for static and low speed group. However, model (b) incorporating of proposed GMF performs significantly better in the static and low speed groups, and achieves comparable performances with (a) in high speed group. In addition, we visualize the pillar description maps in GMF, which represents the probability of inserting the features of each pillar to another point cloud feature map. As shown in Figure 3, there are significantly high mask values in positions without motions, such as the ground, but low values where movable objects and empty pillars appear. What is mentioned above shows GMF learns valid movable-static information and compensation, and further enhance the feature consistency between corresponding pillars which facilitates the pillar association to acquire accurate pillar correspondence.

Soft Discriminative Loss

We also conduct experiments to compare proposed SD–Loss with PointInfoNCE proposed by PointContrast [Xie et al., 2020]. Comparing models (c) and (d) in Table 4, we find that model (c) achieves better performances in all the groups. As shown in Figure 4, we believe that SD–Loss considering the feature similarities of the local points can make an approximate estimation in the case that the corresponding points are not scanned and the corresponding probabilities inferred by model (c) show a Gaussian-like distribution due to the soft pillar correspondence adopted in SD–Loss, which provide a margin of fault tolerance and avoid the destruction of noises and changing views of dynamic scenes.

For ablation experiments of ground points, qualitative and runtime analysis, please refer to the supplementary material.

5 Conclusion

In this paper, we propose ContrastMotion, a self-supervised framework in contrastive learning manner for scene motion. We propose Soft Discriminative Loss to bootstrap discriminative feature extraction and alleviate the correspondence missing situations. We also propose Gated Multi-frame Fusion block learning valid compensation. Pillar association predicts pillar correspondence probabilities and further predicts scene motion. We will focus on improving the inference speed of pillar association in the future.
Acknowledgements
This work was supported by NSFC (No.62206173), Shanghai Frontiers Science Center of Human-centered Artificial Intelligence (ShangHAI) and Natural Science Foundation of China (No. 62172372).

References


