Self-Supervised 3D Scene Flow Estimation Guided by Superpoints

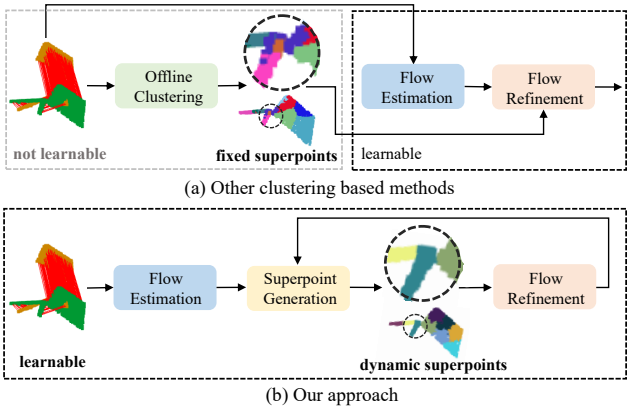
Yaqi Shen, Le Hui, Jin Xie*, Jian Yang
PCA Lab, Nanjing University of Science and Technology, Nanjing, China
{syq, le.hui, csjxie, csjyang}@njust.edu.cn

Abstract

3D scene flow estimation aims to estimate point-wise motions between two consecutive frames of point clouds. Superpoints, i.e., points with similar geometric features, are usually employed to capture similar motions of local regions in 3D scenes for scene flow estimation. However, in existing methods, superpoints are generated with the offline clustering methods, which cannot characterize local regions with similar motions for complex 3D scenes well, leading to inaccurate scene flow estimation. To this end, we propose an iterative end-to-end superpoint based scene flow estimation framework, where the superpoints can be dynamically updated to guide the point-level flow prediction. Specifically, our framework consists of a flow guided superpoint generation module and a superpoint guided flow refinement module. In our superpoint generation module, we utilize the bidirectional flow information at the previous iteration to obtain the matching points of points and superpoint centers for soft point-to-superpoint association construction, in which the superpoints are generated for pairwise point clouds. With the generated superpoints, we first reconstruct the flow for each point by adaptively aggregating the superpoint-level flow, and then encode the consistency between the reconstructed flow of pairwise point clouds. Finally, we feed the consistency encoding along with the reconstructed flow into GRU to refine point-level flow. Extensive experiments on several different datasets show that our method can achieve promising performance. Code is available at https://github.com/supersyq/SPFlowNet.

1. Introduction

Scene flow estimation is one of the vital components of numerous applications such as 3D reconstruction [10], autonomous driving [37], and motion segmentation [2]. Estimating scene flow from stereo videos and RGB-D images has been studied for many years [17, 19]. Recently, with the rapid development of 3D sensors, estimating scene flow from two consecutive point clouds has receiving more and more attention. However, due to the irregularity and sparsity of point clouds, scene flow estimation from point clouds is still a challenging problem in real scenes.



(a) Other clustering based methods

(b) Our approach

Figure 1. Comparison with other clustering-based methods. (a) Other clustering based methods utilize offline clustering algorithms to split the point clouds into some fixed superpoints for subsequent flow refinement, which is not learnable. (b) Our method embeds the differentiable clustering (superpoint generation) into our pipeline and generates dynamic superpoints at each iteration. We visualize part of the scene in FlyingThings3D [38] for better visualization. Different colors indicate different superpoints and red lines indicate the ground truth flow.

In recent years, many 3D scene flow estimation methods have been proposed [11, 34, 37, 56, 57, 59]. Most of these methods [34, 56] rely on dense ground truth scene flow as supervision for model training. However, collecting point-wise scene flow annotations is expensive and time-consuming. To avoid the expensive point-level annotations, some efforts have been dedicated to weakly-supervised and self-supervised scene flow estimation [9, 23, 46, 60]. For example, both Rigid3DSceneFlow [9] and LiDARSceneFlow [7] propose a weakly-supervised scene flow estimation framework, which only take the ego-

*Corresponding authors
Yaqi Shen, Le Hui, Jin Xie, and Jian Yang are with PCA Lab, Key Lab of Intelligent Perception and Systems for High-Dimensional Information of Ministry of Education, and Jiangsu Key Lab of Image and Video Understanding for Social Security, School of Computer Science and Engineering, Nanjing University of Science and Technology, China.

motion and background masks as inputs. Especially, they utilize the DBSCAN clustering algorithm [8] to segment the foreground points into local regions with flow rigidity constraints. In addition, RigidFlow [31] first utilizes the off-line oversegmentation method [32] to decompose the source point clouds into some fixed supervoxels, and then estimates the rigid transformations for supervoxels as pseudo scene flow labels for model training. In summary, these clustering based methods utilize offline clustering algorithms with hand-crafted features (i.e., coordinates and normals) to generate the superpoints and use the consistent flow constraints on these fixed superpoints for scene flow estimation. However, for some complex scenes, the offline clustering methods may cluster points with different flow patterns into the same superpoints. Figure 1(a) shows that [32] falsely clusters points with the entirely different flow into the same superpoint colored in purple (highlighted by the dotted circle). Thus, applying flow constraints to the incorrect and fixed superpoints for flow estimation will mislead the model to generate false flow results.

To address this issue, we propose an iterative end-to-end superpoint guided scene flow estimation framework (dubbed as "SPFlowNet"), which consists of an online superpoint generation module and a flow refinement module. Our pipeline jointly optimizes the flow guided superpoint generation and superpoint guided flow refinement for more accurate flow prediction (Figure 1(b)). Specifically, we first utilize farthest point sampling (FPS) to obtain the initial superpoint centers, including the coordinate, flow, and feature information. Then, we use the superpoint-level and point-level flow information in the previous iteration to obtain the matching points of points and superpoint centers. With the pairs of points and superpoint centers, we can learn the soft point-to-superpoint association map. And we utilize the association map to adaptively aggregate the coordinates, features, and flow values of points for superpoint center updating. Next, based on the updated superpoint-wise flow values, we reconstruct the flow of each point via the generated association map. Furthermore, we encode the consistency between the reconstructed flow of pairwise point clouds. Finally, we feed the reconstructed flow along with the consistency encoding into a gated recurrent unit to refine the point-level flow. Extensive experiments on several benchmarks show that our approach achieves state-of-the-art performance.

Our main contributions are summarized as follows:

- We propose a novel end-to-end self-supervised scene flow estimation framework, which iteratively generates dynamic superpoints with similar flow patterns and refines the point-level flow with the superpoints.
- Different from other offline clustering based methods, we embed the online clustering into our model to dynamically segment point clouds with the guidance from pseudo flow labels generated at the last iteration.
- A superpoint guided flow refinement layer is introduced to refine the point-wise flow with superpoint-level flow information, where the superpoint-wise flow patterns are adaptively aggregated into the point-level with the learned association map.
- Our self-supervised scene flow estimation method outperforms state-of-the-art methods by a large margin.

2. Related Work

Supervised scene flow estimation on point clouds. The scene flow describes the 3D displacements of points between two temporal frames [52]. Estimating scene flow from stereo videos and RGB-D images has been investigated for many years [16, 19, 22, 50]. Recently, with the development of 3D sensor, directly estimating scene flow on point clouds has drawn the interest of many researchers. There are some supervised scene flow estimation methods [3, 25, 53, 55, 58, 59]. FlowNet3D [34] is the first end-to-end scene flow estimation framework on point clouds with a flow embedding layer to capture the local correlation between source and target point clouds and a set upconv layer to propagate the flow embedding from the coarse scale to the finer scale for flow regression. Except for FlowNet3D, some other methods also involve multiscale analysis, such as [6, 12, 54–56]. Among them, Bi-PointFlowNet [6] propagates the features of two frames bidirectionally at different scales to obtain bidirectional correlations, which achieves promising performance. To explicitly encode the rigid motion, HCRF-Flow [29] uses [32] to segment the scenes into supervoxels and takes supervoxels as rigid objects for flow refinement with conditional random fields. Nevertheless, the above methods build local correlations within a limited search area, which fail to accurately estimate the large displacements. Therefore, FLOT [45] and SCTN [27] adopt optimal transport to build global correlation. In contrast, CamLiFlow [33] takes two consecutive synchronized camera and Lidar frames as inputs to estimate the optical flow and scene flow simultaneously and builds multiple bidirectional connections between its 2D and 3D branches to fuse the information of two modalities. Unlike other methods that focus on a pair of point clouds, SPCM-Net [14], MeteorNet [35], and [18] take a sequence of point clouds as input. Specifically, SPCM-Net computes spatiotemporal cost volumes between pairwise two frames and utilizes an order-invariant recurrent unit to aggregate the correlations across time. Although these supervised scene flow estimation methods achieve adorable performance, they need dense supervision for model training, while acquiring point-wise annotations is expensive.

Self-supervised scene flow estimation on point clouds. To address this drawback, there are some self-supervised and weakly-supervised methods [15,28,36,37,42,61]. The self-supervised methods [41,44,51] utilize the cycle-consistency loss and nearest neighbor loss for model training. Besides, PointPWC-Net [60] combines the nearest neighbor loss with a flow smoothness loss and a Laplacian regularization loss as the self-supervised loss. [30] generates pseudo labels by optimal transport and refines the generated pseudo labels with the random walk. The generated pseudo labels are used for unsupervised model optimization. The follow-up RigidFlow [31] utilizes optimization-based point cloud oversegmentation method [32] to split point clouds into a set of supervoxels and then calculates the rigid transformation as pseudo flow labels. Rigid3DSceneFlow [9] and LiDARSceneFlow [7] get rid of the requirement for expensive point-wise flow supervision with binary background masks as well as ego-motion and utilize the DBSCAN clustering algorithm [8] to segment the foreground points for flow rigidity constraints. LiDARSceneFlow expands [9] with a Gated Recurrent Unit (GRU) for flow refinement. The previous methods based on offline clustering mainly employ hand-crafted features (i.e., coordinates and normals) to offline cluster superpoints, which may cluster points with different motion patterns into the same clusters and further lead to worse results with rigidity constraints on the incorrect clusters. Our method attempts to dynamically cluster point clouds into superpoints and then refines the point-wise flow with superpoint-level flow information. In this way, our model can jointly optimize the superpoint generation and flow refinement for more accurate results. Additionally, other self-supervised methods [1, 11, 24] also achieve promising performance.

Point cloud oversegmentation. Point cloud oversegmentation semantically clusters points into superpoints. Recently, some optimization-based superpoint oversegmentation methods are proposed [13, 32]. Among them, [32] converts the point cloud oversegmentation into a subset selection problem and develops a heuristic algorithm to solve it. In contrast, SPNet [21] is the first end-to-end superpoint generation network. Due to low computational cost, superpoints are used for many down-stream tasks, such as point cloud segmentation [4,5,20,48]. In this paper, we introduce superpoints into scene flow estimation based on SPNet. Different from that SPNet focuses on generating superpoints in a single point cloud, our model utilizes the bidirectional flow information at the previous iteration to guide superpoint generation for pairwise point clouds.

3. Method

In this section, we illustrate our superpoint guided scene flow estimation (SPFlowNet) framework in detail. As shown in Figure 2, SPFlowNet consists of a flow guided superpoint generation module and a superpoint guided flow refinement module. It takes two consecutive point clouds $\mathbf{P} = \{\mathbf{p}_i \in \mathbb{R}^3 \mid i = 1, 2, \dots, n\}$ and $\mathbf{Q} = \{\mathbf{q}_j \in \mathbb{R}^3 \mid j = 1, 2, \dots, m\}$ as inputs and outputs the flow $\mathbf{F}^t = \{\mathbf{F}^{p,t}, \mathbf{F}^{q,t}\}$ at the t-th iteration for point clouds $\mathbf{P}$ and $\mathbf{Q}$, respectively. Note that the iteration subscript $t = 0$ means that our model is in the initialization stage.

3.1. Initialization

Initial flow. Firstly, we utilize the feature encoder used in FLOT [45] to extract the features for point clouds $\mathbf{P}$ and $\mathbf{Q}$. The local features of $\mathbf{P}$ and $\mathbf{Q}$ can be denoted as $\mathbf{X} \in \mathbb{R}^{n \times d}$ and $\mathbf{Y} \in \mathbb{R}^{m \times d}$, where d is the dimension of the feature. Then, we calculate the global correlation $\mathbf{W} \in \mathbb{R}^{n \times m}$ between the point clouds $\mathbf{P}$ and $\mathbf{Q}$, where $\mathbf{W}$ can be formulated as the dot product between their features. Next, we apply the Sinkhorn algorithm [47] to it for the final correlation map $\mathbf{W}$. The initial flow $\mathbf{f}_i^{p,0} \in \mathbf{F}^{p,0}$ for each point $\mathbf{p}_i \in \mathbf{P}$ can be defined as

$$\mathbf{f}_i^{p,0} = \frac{\sum_{j=1}^{m} w_{i,j}\mathbf{q}_j}{\sum_{j=1}^{m} w_{i,j}} - \mathbf{p}_i \tag{1}$$

Similarly, we can obtain the initial flow $\mathbf{F}^{q,0}$ for point clouds $\mathbf{Q}$ by taking the same operations as $\mathbf{P}$ on $\mathbf{Q}$.

Initial superpoint center. We obtain L ($L \ll n$ and $L \ll m$) initial superpoint centers $\mathbf{SP}^0 = \{\mathbf{SP}^{p,0}, \mathbf{SP}^{q,0}\}$ for point clouds $\mathbf{P}$ and $\mathbf{Q}$ by employing the FPS algorithm in the coordinate space. $\mathbf{SP}^{p,0}$ and $\mathbf{SP}^{q,0}$ denote the initial superpoint centers for pairwise point clouds $\mathbf{P}$ and $\mathbf{Q}$, respectively. Each superpoint center includes the coordinate, flow, and descriptor information, denoted by $\mathbf{SC}^0$, $\mathbf{SF}^0$, and $\mathbf{SD}^0$, respectively.

3.2. Flow Guided Superpoint Generation

The scene flow estimation methods [9,31] usually exploit the offline clustering methods [8, 32] to decompose the point clouds into a collection of clusters and employ the flow rigidity constraints on the fixed clusters. However, the offline clustering methods usually generate false clusters, where the points with different flow patterns exist in the same cluster, as shown in Figure 1(a). Therefore, an online flow guided superpoint generation module is embedded in our framework, in which the point clouds are dynamically divided into superpoints. Due to the joint end-to-end optimization with the consequent flow refinement module, our model can relieve the above problem to some extent.

Point-to-Superpoint association calculation. Our method attempts to generate superpoints that satisfy the following requirements: (1) The points of the same superpoint are with similar flow patterns; (2) They are also close to the superpoint centers in the coordinate space; (3) Their features are semantically similar with each other. Thus, we

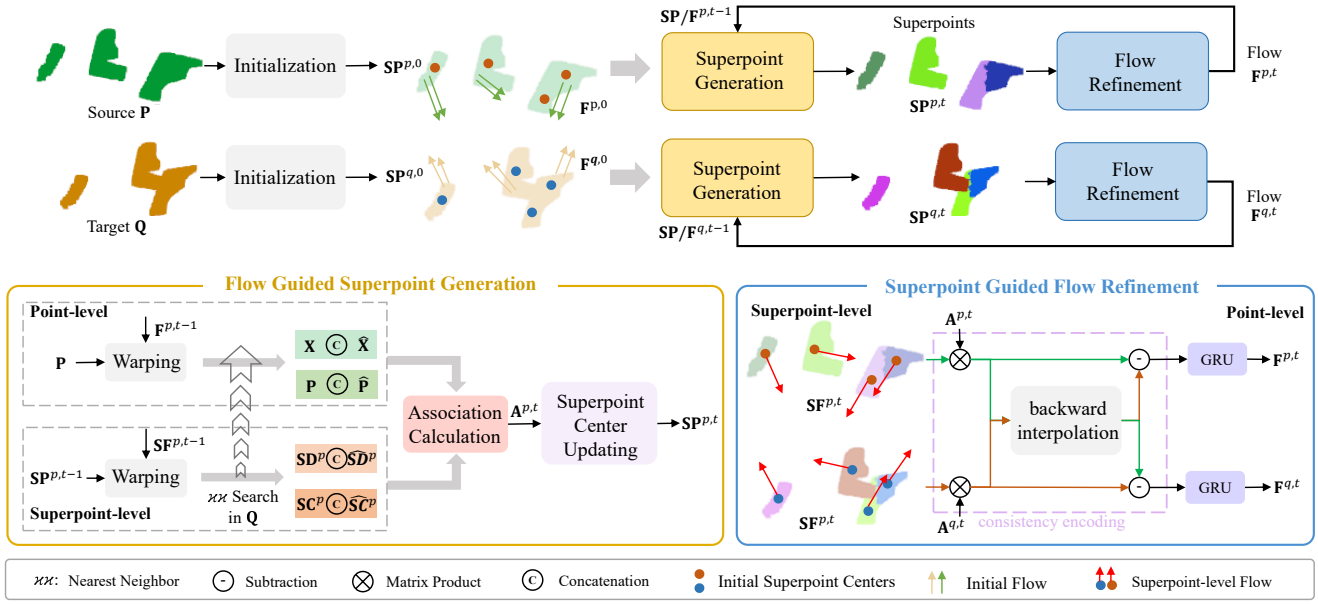


Figure 2. An overview of SPFlowNet. Given two consecutive point clouds $\mathbf{P}$ and $\mathbf{Q}$, we first calculate the initial flow $\mathbf{F}^0 = \{\mathbf{F}^{p,0}, \mathbf{F}^{q,0}\}$ and the initial superpoint centers $\mathbf{SP}^0 = \{\mathbf{SP}^{p,0}, \mathbf{SP}^{q,0}\}$ at the initialization stage ($t = 0$). Then, our model iteratively performs the flow guided superpoint generation module and the superpoint guided flow refinement module for scene flow estimation. In the end, we can obtain final flow results after several iterations. Specifically, at the t-th iteration, the flow guided superpoint generation module clusters points into dynamic superpoints $\mathbf{SP}^t = \{\mathbf{SP}^{p,t}, \mathbf{SP}^{q,t}\}$ with the pseudo superpoint-level and point-level flow labels generated at the previous iteration. With the generated superpoints, the superpoint guided flow refinement module feeds the superpoint-level flow and consistency encoding into GRU to obtain the updated point-level flow $\mathbf{F}^{p,t}$ and $\mathbf{F}^{q,t}$.

follow SPNet [21] to build the soft association map between points and superpoint centers by adaptively learning the bilateral weights from both the coordinate and feature spaces. Different from SPNet, we introduce the previously iterated flow information at both point level and superpoint level to obtain the corresponding point/superpoint center via the bidirectional warping operation (source $\rightarrow$ target and target $\rightarrow$ source). Thus, we employ pairs of points and superpoint centers in the source and target point clouds to learn the similarity across the source and target while SPNet does not consider the pairs of corresponding points to learn the similarity. Note that following SPNet, we only calculate the association weights between each point and its K-nearest superpoint centers ($K \ll L$) in the coordinate space, which is more efficient.

We take the source point cloud $\mathbf{P}$ as an example to illustrate the point-to-superpoint association map calculation. Specifically, for the i-th point in source point cloud $\mathbf{P}$, we use the Euclidean distance in the coordinate space to select the attended K superpoint centers $\mathcal{N}_i$, where $\mathcal{N}_i = \{\mathbf{sc}_{i,k}^{p,t-1} \in \mathbb{R}^3, \mathbf{sf}_{i,k}^{p,t-1} \in \mathbb{R}^3, \mathbf{sd}_{i,k}^{p,t-1} \in \mathbb{R}^d\}_{k=0}^{K}$ includes the coordinate, flow, and feature information of the K-nearest superpoint centers. At the t-th iteration, the association $a_{i,k}^{p,t} \in \mathbf{A}^{p,t}$ between the i-th point and the k-th superpoint center in source point cloud $\mathbf{P}$ is defined as

$$
\begin{aligned}
a_{i,k}^{p,t} &= \mathrm{MLP}\left(\mathbf{u}_{i,k}\right) + \mathrm{MLP}\left(\mathbf{g}_{i,k}\right) \\
\mathbf{u}_{i,k} &= \left(\mathbf{x}_i || \hat{\mathbf{x}}_i^{t-1}\right) - \left(\mathbf{sd}_{i,k}^{p,t-1} || \hat{\mathbf{sd}}_{i,k}^{p,t-1}\right) \\
\mathbf{g}_{i,k} &= \left(\mathbf{p}_i || \hat{\mathbf{p}}_i^{t-1}\right) - \left(\mathbf{sc}_{i,k}^{p,t-1} || \hat{\mathbf{sc}}_{i,k}^{p,t-1}\right)
\end{aligned}
\tag{2}
$$

where $||$ is the concatenation, $\mathbf{u}_{i,k} \in \mathbb{R}^{1\times(2*d)}$ and $\mathbf{g}_{i,k} \in \mathbb{R}^{1\times(2*3)}$ represent the differences between the i-th point and the k-th superpoint center in feature and coordinate spaces, respectively. Besides, $\mathrm{MLP}(\cdot)$ denotes a multi-layer perceptron followed by a sum-pooling operation, which is used to map the above difference information to association weights in both coordinate and feature spaces.

In Equation (2), we also utilize the feature and coordinate information of their corresponding points generated by the predicted point-level and superpoint-level flow in the previous iteration. The corresponding point $(\hat{\mathbf{p}}_i^{t-1}, \hat{\mathbf{x}}_i^{t-1})$ and superpoint center $(\hat{\mathbf{sc}}_{i,k}^{p,t-1}, \hat{\mathbf{sd}}_{i,k}^{p,t-1})$ for point $\mathbf{p}_i$ and superpoint center $\mathbf{sc}_{i,k}^{p,t-1}$ are defined as

$$
\begin{aligned}
\hat{\mathbf{p}}_i^{t-1} &= \mathbf{p}_i + \mathbf{f}_i^{p,t-1}, \hat{\mathbf{sc}}_{i,k}^{p,t-1} = \mathbf{sc}_{i,k}^{p,t-1} + \mathbf{sf}_{i,k}^{p,t-1} \\
\hat{\mathbf{x}}_i^{t-1} &= \mathbf{Y}_{\mathrm{NN}(\hat{\mathbf{p}}_i^{t-1}, \mathbf{Q})}, \hat{\mathbf{sd}}_{i,k}^{p,t-1} = \mathbf{Y}_{\mathrm{NN}(\hat{\mathbf{sc}}_{i,k}^{p,t-1}, \mathbf{Q})}
\end{aligned}
\tag{3}
$$

where $\mathrm{NN}(\cdot, \mathbf{Q})$ is used to obtain the index of the nearest matching point in target point cloud $\mathbf{Q}$.

Next, we assign each point $\mathbf{p}_i \in \mathbf{P}$ a probability vector over its K-nearest superpoint centers by

$$a_{i,k}^{p,t} = \mathrm{softmax}\left(\left[a_{i,1}^{p,t}, ..., a_{i,K}^{p,t})\right]\right)_k \tag{4}$$

Similarly, we can obtain the association map $\mathbf{A}^{q,t}$ between the target point cloud $\mathbf{Q}$ and its superpoint centers.

Superpoint center updating. With the generated association map $\mathbf{A}^t$, we can assign each point to its K-nearest superpoint centers with the learned weights. For each superpoint center, we adaptively aggregate the coordinate, flow, and feature information of the points belonging to it to update this superpoint center via the normalized association map. Specifically, given the local feature $\mathbf{X}$, flow $\mathbf{F}^{p,t-1} = \{\mathbf{f}_i^{p,t-1} | i = 1, \ldots, n\}$ at the iteration $t-1$ and the association map $\mathbf{A}^{p,t}$ at the current t-th iteration of the source point cloud $\mathbf{P}$, the updated l-th superpoint center in source point clouds can be formulated as

$$\begin{aligned}
\mathbf{sc}_l^{p,t} &= \frac{1}{r}\sum\nolimits_{i=1}^{n} \mathbb{1}\left\{l \in \mathcal{N}_i\right\} a_{i,l}^{p,t} \mathbf{p}_i \\
\mathbf{sf}_l^{p,t} &= \frac{1}{r}\sum\nolimits_{i=1}^{n} \mathbb{1}\left\{l \in \mathcal{N}_i\right\} a_{i,l}^{p,t} \mathbf{f}_i^{p,t-1} \\
\mathbf{sd}_l^{p,t} &= \frac{1}{r}\sum\nolimits_{i=1}^{n} \mathbb{1}\left\{l \in \mathcal{N}_i\right\} a_{i,l}^{p,t} \mathbf{x}_i
\end{aligned} \tag{5}$$

where $\mathbb{1}\left\{l \in \mathcal{N}_i\right\}$ is an indicator function that equals to one if the l-th superpoint center belongs to $\mathcal{N}_i$, and zero otherwise. Besides, $r = \sum_{i=1}^{n} \mathbb{1}\left\{l \in \mathcal{N}_i\right\} a_{i,l}^{p,t}$ is the normalization factor. Similarly, we update the superpoint centers in target point cloud $\mathbf{Q}$. For brevity, we only visualize the pipeline of flow guided superpoint generation for source point cloud $\mathbf{P}$ in Figure 2.

3.3. Superpoint Guided Flow Refinement

Inspired by RAFT [49], many scene flow estimation methods [11, 26, 59] utilize a Gate Recurrent Unit (GRU) to iteratively update the predicted flow.

Gated recurrent unit. Given the hidden state $\mathbf{h}^{t-1}$ at the iteration $t-1$ and the current iteration information $\mathbf{v}^t$, the calculations of GRU can be written as

$$\begin{aligned}
\mathbf{z}^t &= \sigma\left(\mathrm{SetConv}_z\left(\mathbf{h}^{t-1} || \mathbf{v}^t\right)\right) \\
\mathbf{r}^t &= \sigma\left(\mathrm{SetConv}_r\left(\mathbf{h}^{t-1} || \mathbf{v}^t\right)\right) \\
\hat{\mathbf{h}}^t &= \tanh\left(\mathrm{SetConv}_h\left(\left(\mathbf{r}^t \odot \mathbf{h}^{t-1}\right) || \mathbf{v}^t\right)\right) \\
\mathbf{h}^t &= \left(1 - \mathbf{z}^t\right) \odot \mathbf{h}^{t-1} + \mathbf{z}^t \odot \hat{\mathbf{h}}^t
\end{aligned} \tag{6}$$

where $\odot$ is the Hadamard product, $||$ is the concatenation, and $\sigma(\cdot)$ is the sigmoid function. The $\mathrm{SetConv}$ layers are adopted from [26, 45].

The existing GRU-based methods usually concatenate the feature, flow, and flow embedding of each point as current iteration information $\mathbf{v}^t$, and regress the flow from the new hidden state $\mathbf{h}^t$. Although these methods achieve promising results, most of them only involve point-level flow information. In contrast, [7] converts GRU output into rigid flow according to pre-clustered local regions, it is limited by pre-clustered regions. Our method adaptively learns the flow association at the superpoint level and does not rely on rigid object assumption. Specifically, we encode the superpoint-level flow information into the current iteration information $\mathbf{v}^t$ to guide the new hidden state $\mathbf{h}^t$ generation. Moreover, we utilize the consistency between the reconstructed flow values from the generated superpoints of pairwise point clouds to encode the confidence into the current iteration information $\mathbf{v}^t$. Therefore, the current iteration information in our model simultaneously considers the superpoint-level flow information and its confidence.

Superpoint-level flow reconstruction. With the updated superpoint-level flow $\mathbf{SF}^{p,t}$ and $\mathbf{SF}^{q,t}$ for superpoint centers in both point clouds $\mathbf{P}$ and $\mathbf{Q}$, here we map the superpoint-level flow of K-nearest superpoint centers back onto each point in original point clouds via the learned association map $\mathbf{A}^t$ as follows

$$\widetilde{\mathbf{F}}_i^{p,t} = \sum\nolimits_{k=1}^{K} a_{i,k}^{p,t} \mathbf{sf}_k^{p,t}, \widetilde{\mathbf{F}}_i^{q,t} = \sum\nolimits_{k=1}^{K} a_{i,k}^{q,t} \mathbf{sf}_k^{q,t} \tag{7}$$

where $\widetilde{\mathbf{F}}_i^{p,t}$ and $\widetilde{\mathbf{F}}_i^{q,t}$ are the reconstructed superpoint-level flow for point clouds $\mathbf{P}$ and $\mathbf{Q}$, respectively. In this way, the reconstructed flow of each point in original point clouds adaptively aggregates the superpoint-level flow values of its K-nearest superpoint centers. Since the superpoint-level flow values capture the flow patterns of the generated superpoints, we aim to utilize the superpoint-level flow pattern to guide the point-level flow refinement.

Consistency encoding. We do a backward interpolation Ω used in [43] to propagate the reconstructed superpoint-level flow in the source point clouds to the target point clouds and vice versa. Next, we utilize the consistency between the interpolated flow and reconstructed flow to encode the confidence of the superpoint-level flow by

$$\mathbf{C}^{p,t} = \pi\left(\widetilde{\mathbf{F}}^{p,t} - \Omega(\widetilde{\mathbf{F}}^{q,t})\right), \mathbf{C}^{q,t} = \tau\left(\widetilde{\mathbf{F}}^{q,t} - \Omega(\widetilde{\mathbf{F}}^{p,t})\right) \tag{8}$$

where π and τ are the MLP layers with a sigmoid function.

Besides, we send the reconstructed superpoint-level flow $\widetilde{\mathbf{F}}^{p,t}$ into a flow embedding layer used in [26] to obtain the correlation feature $\mathbf{F}_c^{p,t}$ and a Linear layer to encode the flow feature $\mathbf{F}_e^{p,t}$. With the confidence $\mathbf{C}^{p,t}$, the current iteration information $\mathbf{v}^t$ for source point cloud $\mathbf{P}$ can be defined as

$$\mathbf{v}^t = \mathrm{SetConv_c}\left(\mathbf{F}_c^{p,t}\mathbf{C}^{p,t}\right) + \mathrm{SetConv_e}\left(\mathbf{F}_e^{p,t}\mathbf{C}^{p,t}\right) \tag{9}$$

where the $\mathrm{SetConv}$ layers are adopted from [26, 45].

We send $\mathbf{v}^t$ into GRU to obtain the new hidden state $\mathbf{h}^t$. Finally, given the new hidden state $\mathbf{h}^t$, we use a flow regressor to obtain the residual flow $\triangle\mathbf{F}^{p,t}$. Therefore, the updated flow for source point cloud $\mathbf{P}$ at the iteration t can be formulated as $\mathbf{F}^{p,t} = \widetilde{\mathbf{F}}^{p,t-1} + \triangle\mathbf{F}^{p,t}$. Similarly, we can obtain the updated flow $\mathbf{F}^{q,t}$ for target point cloud $\mathbf{Q}$.

3.4. Self-Supervised Loss Functions

At each iteration, we can obtain the estimated flow $\mathbf{F}_t = \{\mathbf{F}^{p,t}, \mathbf{F}^{q,t}\}$ for pairwise point clouds $\mathbf{P}$ and $\mathbf{Q}$. Without the ground truth scene flow, we utilize the following loss functions for model training. For simplicity, we omit the iteration subscript.

Chamfer loss. Following [26, 60], we warp the source $\mathbf{P}$ with the predicted flow $\mathbf{F}^p$ and minimize the Chamfer Distance between the warped source $\boldsymbol{P}'$ and target $\mathbf{Q}$ by

$$L_{ch}(\mathbf{P}', \mathbf{Q}) = \sum_{\mathbf{q}_j \in \boldsymbol{Q}} \min_{\mathbf{p'}_i \in \mathbf{P}'} \|\mathbf{q}_j - \mathbf{p'}_i\|_2 + \sum_{\mathbf{p'}_i \in \mathbf{P}'} \min_{\mathbf{q}_j \in \mathbf{Q}} \|\mathbf{p'}_i - \mathbf{q}_j\|_2 \tag{10}$$

Smoothness loss. Following [26, 60], we also constrain the predicted scene flow values within a small local region to be similar. The smoothness loss is defined as

$$L_s = \sum_{\mathbf{p}_i \in \boldsymbol{P}} \frac{1}{|\mathcal{N}'_i|} \sum_{\mathbf{p}_j \in \mathcal{N}'_i} \|\mathbf{f}^p_i - \mathbf{f}^p_j\|_2 \tag{11}$$

where $\mathcal{N}'_i$ is the neighborhood around $\mathbf{p}_i \in \boldsymbol{P}$.

Consistency loss. We enforce the backward-interpolated flow of the target point clouds to be consistent with the predicted flow of the source point clouds and vice versa.

$$L_c = \|\mathbf{F}^p - \Omega\left(\mathbf{F}^q\right)\|_2 + \|\mathbf{F}^q - \Omega\left(\mathbf{F}^p\right)\|_2 \tag{12}$$

where Ω is backward interpolation.

The combined loss for self-supervised training can be written as

$$L = L_{ch} + \alpha L_s + \beta L_c \tag{13}$$

where α and β are the regularization parameters.

4. Experiment

4.1. Experimental Setups

Datasets. To validate the effectiveness of our proposed scene flow estimation framework, we conduct extensive experiments on two benchmarks, the FlyingThings3D [38] and the KITTI Scene Flow [39, 40]. There are two versions of datasets. The first version of the datasets is prepared by HPLFlowNet [12]. We denote these datasets without occluded points $\mathrm{FT3D_s}$ and $\mathrm{KITTI_s}$, respectively. $\mathrm{FT3D_s}$ contains 19640 training examples and 3824 pairs in the test set. We only use one-quarter of the training data (4910 pairs). $\mathrm{KITTI_s}$ is a real-world scene flow dataset with 200 pairs for which 142 are used for testing without any fine-tuning. The second version of the datasets is prepared by FlowNet3D [34]. This version of datasets includes the occluded points, which are denoted $\mathrm{FT3D_o}$ and $\mathrm{KITTI_o}$, respectively. $\mathrm{FT3D_o}$ contains 19999 training examples and 2003 pairs in the test set. $\mathrm{KITTI_o}$ consists of 150 test examples. Besides, following Self-Point-Flow [30], we also split the $\mathrm{KITTI_o}$ dataset into $\mathrm{KITTI_f}$ with 100 pairs and $\mathrm{KITTI_t}$ with 50 pairs for evaluation. Moreover, [30] also extracts another self-supervised training dataset with 6026 pairs from the original KITTI dataset, denoted as $\mathrm{KITTI_r}$.

Implementation details. Our model is implemented with Pytorch and all experiments are executed on a NVIDIA TITAN RTX GPU. For the experiments on point clouds without occlusions, we train our model on synthetic $\mathrm{FT3D_s}$ training data and evaluate it on both $\mathrm{FT3D_s}$ test data and $\mathrm{KITTI_s}$ dataset. we feed randomly sampled 8192 points as inputs to our model, just like [31, 45] and other compared methods, and train it with a batch size of 2. Besides, we set the superpoint number and iteration number to 128 and 3, respectively. For occluded experiments, like [31], we also train our model on $\mathrm{KITTI_r}$ dataset and test it on $\mathrm{KITTI_o}$ and $\mathrm{KITTI_t}$. The size of the input point clouds is set to 2048. Here, we set the batch size, iteration number, and superpoint number to 4, 3, and 30, respectively. The initial learning rate used in all experiments is 0.001 and our model is optimized with the ADAM optimizer. We multiply the learning rate by 0.7 at epochs 40, 55, and 70 and train our model for 100 epochs.

Evaluation metrics. We test our model with four evaluation metrics used in [12, 34], including End Point Error (EPE), Accuracy Strict (AS), Accuracy Relax (AR), and Outliers (Out). We denote the estimated scene flow and ground truth scene flow as $\mathbf{F}$ and $\mathbf{F}_{gt}$, respectively. EPE(m): $||\mathbf{F} - \mathbf{F}_{gt}||_2$ averaged over all points. AS(%): the percentage of points whose EPE $<0.05m$ or relative error $<5\%$. AR(%): the percentage of points whose EPE $<0.1m$ or relative error $<10\%$. Out(%): the percentage of points whose EPE $>0.3m$ or relative error $>10\%$.

4.2. Results

Performance on point clouds without occlusions. We train our self-supervised model on $\mathrm{FT3D_s}$ training data and evaluate it on both $\mathrm{FT3D_s}$ test data and $\mathrm{KITTI_s}$ dataset. And we compare our model with the recent state-of-the-art self-supervised scene flow estimation methods, including Ego-Motion [51], PointPWC-Net [60], SLIM [2], Self-Point-Flow [30], FlowStep3D [26], RCP [11], PDF-Flow [15], and RigidFlow [31]. The results are reported in Table 1. From the results, it can be found that our model can

outperform all compared self-supervised methods in terms of the four metrics on the FT3D_s test data. Especially, our model brings 8.72% gains for metric AS. For the KITTI_s dataset, our model brings substantial improvements on all metrics. To be specific, our model outperforms the second best method RCP [11] by 8.68% and 6.58% on metrics AS and AR, respectively. Besides, it is worth noting that our model can even achieve an EPE metric of $3.62cm$, which is much lower than the EPE ($6.19cm$) of recent RigidFlow.

We also compare our model with some supervised methods, such as FlowNet3D [34] and FLOT [45], etc. As shown in Table 1, our self-supervised model achieves comparable performance with supervised HPLFlowNet [12] on FT3D_s dataset. Without any fine-tuning on KITTI_s dataset, our model can even outperform the supervised methods listed in Table 1, which proves that our model has better generalization ability. For real scenes, most local regions are with similar flow patterns. Thanks to dynamically clustering mechanism, our model clusters points with similar flow pattern into the same clusters and encodes the superpoint-level flow into the GRU for flow refinement, thereby leading to satisfactory performance on real scenes.

Performance on point clouds with occlusions. Following the experimental settings used in Self-Point-Flow [30] and RigidFlow [31], we train our model on KITTI_r dataset and evaluate our model on both KITTI_o and KITTI_t datasets. The results on KITTI_o and KITTI_t are shown in Tables 2 and 3, respectively. Although our model is not designed to deal with occluded cases, our model can also achieve the best performance on KITTI_o dataset. This is due to that although there is no correspondence of the occluded points, our model employs the superpoint-level flow to guide the flow refinement rather than point-level flow information, which can alleviate the occluded problem to some extent. Due to the lack of point-level flow annotations for the real scenes, the supervised FLOT and FlowNet3D are trained on synthetic FT3D_o dataset. The other two self-supervised methods [30, 31] and our model can be trained directly on unlabeled outdoor KITTI_r dataset. As shown in Table 2, our model can outperform all self-supervised methods including Self-Point-Flow and RigidFlow. To be specific, our model brings 12.7% gains on metric AS. Besides, it is worth noting that the Self-Point-Flow [30] needs additional normal and color information for pseudo label generation. Our model only needs the coordinate information of the consecutive frames of point clouds as inputs. For the KITTI_t dataset, we compare our model with JGF [41] and WWL [44]. These two methods use the pre-trained model of FlowNet3D on FT3D_o as the baseline and perform self-supervised fine-tuning on KITTI_f, and then test their model on KITTI_t dataset. Our model and RigidFlow [31] get rid of the pre-trained model on synthetic FT3D_o and only need to be trained on the unlabeled KITTI_r in a self-supervised manner. As shown in Table 3, our model can obtain 9.92% improvements on metric AR.

Methods	Sup.	EPE ↓	AS ↑	AR ↑	Out ↓
FT3D_s					
FlowNet3D [34]	Full.	0.1136	41.25	77.06	60.16
HPLFlowNet [12]	Full.	0.0804	61.44	85.55	42.87
PointPWC-Net [60]	Full.	0.0588	73.79	92.76	34.24
Ego-Motion [51]	Self.	0.1696	25.32	55.01	80.46
PoinPWC-Net [60]	Self.	0.1246	30.68	65.52	70.32
Self-Point-Flow [30]	Self.	0.1009	42.31	77.47	60.58
FlowStep3D [26]	Self.	0.0852	53.63	82.62	41.98
PDF-Flow [15]	Self.	0.075	58.9	86.2	47.0
RCP [11]	Self.	0.0765	58.58	86.02	41.42
RigidFlow [31]	Self.	0.0692	59.62	87.10	46.42
SPFlowNet (ours)	Self.	**0.0606**	**68.34**	**90.74**	**38.76**
KITTI_s					
FlowNet3D [34]	Full.	0.1767	37.38	66.77	52.71
HPLFlowNet [12]	Full.	0.1169	47.83	77.76	41.03
PointPWC-Net [60]	Full.	0.0694	72.81	88.84	26.48
FLOT [45]	Full.	0.0560	75.50	90.80	24.20
Ego-Motion [51]	Self.	0.4154	22.09	37.21	80.96
PoinPWC-Net [60]	Self.	0.2549	23.79	49.57	68.63
SLIM [2]	Self.	0.1207	51.78	79.56	40.24
FlowStep3D [26]	Self.	0.1021	70.80	83.94	24.56
PDF-Flow [15]	Self.	0.092	74.7	87.0	28.3
Self-Point-Flow [30]	Self.	0.1120	52.76	79.36	40.86
RigidFlow [31]	Self.	0.0619	72.37	89.23	26.18
RCP [11]	Self.	0.0763	78.56	89.21	18.49
SPFlowNet (ours)	Self.	**0.0362**	**87.24**	**95.79**	**17.71**

Table 1. Comparison results on the FT3D_s and KITTI_s datasets. Our model is trained on FT3D_s training part and evaluated on FT3D_s test set and KITTI_s dataset. Full. means the fully-supervised training manner. Self. represents the self-supervised training manner. Note that the best and the second-best results are emboldened and underlined, respectively.

Methods	Sup.	T. data	EPE ↓	AS ↑	AR ↑	Out ↓
FlowNet3D [34]	Full.	F_o	0.173	27.6	60.9	64.9
FLOT [45]	Full.	F_o	0.107	45.1	74.0	46.3
Self-Point-Flow [30]	Self.	K_r	0.105	41.7	72.5	50.1
RigidFlow [31]	Self.	K_r	0.102	48.4	75.6	44.2
SPFlowNet (ours)	Self.	K_r	**0.086**	**61.1**	**82.4**	**39.1**

Table 2. Comparison results on KITTI_o dataset. Our model is trained on KITTI_r and evaluated on KITTI_o dataset. T. data: training data. F_o: FT3D_o. K_r:KITTI_r.

4.3. Ablation Study

The effectiveness of key components. We conduct experiments to verify the effectiveness of key components in our

Methods	Pre-T.	T. data	EPE ↓	AS ↑	AR ↑
TGF [41]	✓	$F_o + K_f$	0.218	10.17	34.38
WWL [44]	✓	$F_o + K_f$	0.169	21.71	47.75
RigidFlow [31]		K_r	0.117	38.75	69.73
SPFlowNet (ours)		K_r	**0.089**	**53.28**	**79.65**

Table 3. Comparison results on $KITTI_t$ dataset. Our model is trained on $KITTI_r$ and evaluated on $KITTI_t$ dataset.

Methods	EPE ↓	AS ↑	AR ↑	Out ↓
w/o superpoint	0.119	55.4	72.9	45.2
w/ SPNet	0.090	60.0	80.7	40.2
w/ FGSG (ours)	**0.086**	**61.1**	**82.4**	**39.1**
w/o cons. encoding	0.103	57.7	76.1	44.3
w/o cons. loss	0.094	59.0	80.0	40.7
SPFlowNet (ours)	**0.086**	**61.1**	**82.4**	**39.1**

Table 4. Comparison results on the $KITTI_o$ dataset. All models are trained on $KITTI_r$ and evaluated on $KITTI_o$ dataset.

model. Firstly, we remove the superpoint generation and superpoint guided flow refinement modules in our model. This variant takes a GRU without superpoint guidance for flow refinement (abbr. as "w/o superpoint"). Secondly, we adopt the SPNet [21] for superpoint generation without flow guidance (abbr. as "w/ SPNet"). The model "w/ FGSG (ours)" represents our model with flow guided superpoint generation module. The results of the above three models are listed in the top part of Table 4. From the results of the variant "w/o superpoint" and the other two models with superpoints, it can be found that introducing superpoints into scene flow estimation is effective. Besides, our proposed flow guided superpoint generation module can achieve better results than SPNet, which shows that flow guidance is crucial when there is no ground truth superpoint labels. Besides, we remove the consistency encoding from our model (abbr. as "w/o cons. encoding"). Table 4 shows that the performance drops a lot without the superpoint consistency encoding, which demonstrates that the consistency between the reconstructed superpoint-level flow of pairwise point clouds is important. Finally, we also remove the consistency loss and only utilize the Chamfer loss and smoothness loss for model training (abbr. as "w/o cons. loss"). The results of our model without consistency loss are worse than with it. According to the above comparisons, it can be observed that our model is less effective without any key components.

Choices of the superpoint number L. In our superpoint generation layer, we generate L superpoints. We conduct the ablation study to choose a suitable superpoint number. We plot the results of the metrics AS and AR with different

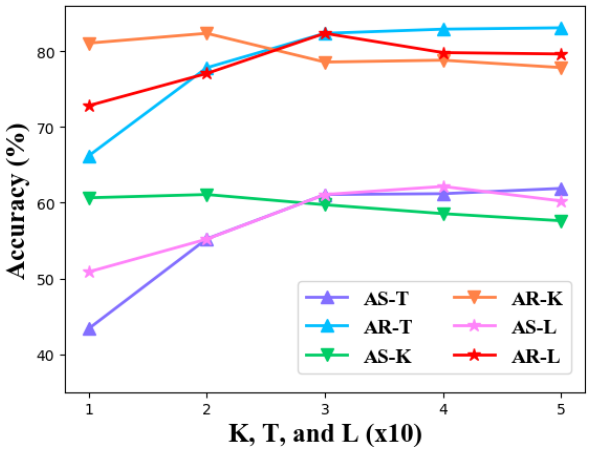


Figure 3. The ablation study results (**AS** and **AR**) of different hyper-parameters L, K, and T on the $KITTI_o$ dataset, where $L \in \{10, 20, 30, 40, 50\}$ and $K, T \in \{1, 2, 3, 4, 5\}$.

$L \in \{10, 20, 30, 40, 50\}$ in Figure 3. It can be observed that choosing $L = 30$ achieves the best results.

Impact of the K-nearest superpoint centers. To prevent a point from being clustered to a distant superpoint, we only calculate the association map between each point and its K-nearest superpoint centers. Here we explore the impact on the performance of different K. We fix other super-parameters and choice $K \in \{1, 2, 3, 4, 5\}$. The accuracy results are visualized in Figure 3. Figure 3 shows that our model achieves the best performance with $K = 2$.

Number of iterations T. Our model iteratively generates superpoints and conducts the superpoint guided flow refinement. We plot the accuracy results of our model after each iteration. From Figure 3, it can be found that $T = 3$ can obtain state-of-the-art performance. Although $T = 4, 5$ can achieve slightly high accuracy, it increases the inference time. Therefore, for a good trade-off between the accuracy and efficiency, we choose $T = 3$.

5. Conclusion

We proposed a novel end-to-end superpoint guided scene flow estimation framework. Different from other offline clustering based scene flow estimation methods, our method can simultaneously optimize the flow guided superpoint generation and superpoint guided flow refinement. Thanks to the joint end-to-end optimization, our model can gradually generate more accurate flow results. Extensive experiments on the synthetic and real LiDAR scenes demonstrate that our self-supervised model can achieve outstanding performance in the scene flow estimation task.

6. Acknowledgements

This work was supported by the National Science Foundation of China (Grant Nos. U62276144, U1713208).

References

[1] Ramy Battrawy, René Schuster, Mohammad-Ali Nikouei Mahani, and Didier Stricker. Rms-flownet: Efficient and robust multi-scale scene flow estimation for large-scale point clouds. In *ICRA*, 2022. 3

[2] Stefan Andreas Baur, David Josef Emmerichs, Frank Moosmann, Peter Pinggera, Björn Ommer, and Andreas Geiger. Slim: Self-supervised lidar scene flow and motion segmentation. In *ICCV*, 2021. 1, 6, 7

[3] Aseem Behl, Despoina Paschalidou, Simon Donné, and Andreas Geiger. Pointflownet: Learning representations for rigid motion estimation from point clouds. In *CVPR*, 2019. 2

[4] Mingmei Cheng, Le Hui, Jin Xie, and Jian Yang. Sspc-net: Semi-supervised semantic 3d point cloud segmentation network. In *AAAI*, 2021. 3

[5] Mingmei Cheng, Le Hui, Jin Xie, Jian Yang, and Hui Kong. Cascaded non-local neural network for point cloud semantic segmentation. In *IROS*, 2020. 3

[6] Wencan Cheng and Jong Hwan Ko. Bi-pointflownet: Bidirectional learning for point cloud based scene flow estimation. In *ECCV*, 2022. 2

[7] Guanting Dong, Yueyi Zhang, Hanlin Li, Xiaoyan Sun, and Zhiwei Xiong. Exploiting rigidity constraints for lidar scene flow estimation. In *CVPR*, pages 12776–12785, 2022. 1, 3, 5

[8] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, 1996. 2, 3

[9] Zan Gojcic, Or Litany, Andreas Wieser, Leonidas J Guibas, and Tolga Birdal. Weakly supervised learning of rigid 3d scene flow. In *CVPR*, 2021. 1, 3

[10] Paulo FU Gotardo, Tomas Simon, Yaser Sheikh, and Iain Matthews. Photogeometric scene flow for high-detail dynamic 3d reconstruction. In *ICCV*, 2015. 1

[11] Xiaodong Gu, Chengzhou Tang, Weihao Yuan, Zuozhuo Dai, Siyu Zhu, and Ping Tan. Rcp: Recurrent closest point for point cloud. In *CVPR*, 2022. 1, 3, 5, 6, 7

[12] Xiuye Gu, Yijie Wang, Chongruo Wu, Yong Jae Lee, and Panqu Wang. Hplflownet: Hierarchical permutohedral lattice flownet for scene flow estimation on large-scale point clouds. In *CVPR*, 2019. 2, 6, 7

[13] Stéphane Guinard and Loic Landrieu. Weakly supervised segmentation-aided classification of urban scenes from 3d lidar point clouds. In *ISPRS Workshop*, 2017. 3

[14] Pan He, Patrick Emami, Sanjay Ranka, and Anand Rangarajan. Learning scene dynamics from point cloud sequences. *IJCV*, 2022. 2

[15] Pan He, Patrick Emami, Sanjay Ranka, and Anand Rangarajan. Self-supervised robust scene flow estimation via the alignment of probability density functions. In *AAAI*, 2022. 3, 6, 7

[16] Evan Herbst, Xiaofeng Ren, and Dieter Fox. Rgb-d flow: Dense 3-d motion estimation using color and depth. In *ICRA*, 2013. 2

[17] Michael Hornacek, Andrew Fitzgibbon, and Carsten Rother. Sphereflow: 6 dof scene flow from rgb-d pairs. In *CVPR*, 2014. 1

[18] Shengyu Huang, Zan Gojcic, Jiahui Huang, and Konrad Schindler Andreas Wieser. Dynamic 3d scene analysis by point cloud accumulation. In *ECCV*, 2022. 2

[19] Frédéric Huguet and Frédéric Devernay. A variational method for scene flow estimation from stereo sequences. In *ICCV*, 2007. 1, 2

[20] Le Hui, Linghua Tang, Yaqi Shen, Jin Xie, and Jian Yang. Learning superpoint graph cut for 3d instance segmentation. In *NeurIPS*, 2022. 3

[21] Le Hui, Jia Yuan, Mingmei Cheng, Jin Xie, Xiaoya Zhang, and Jian Yang. Superpoint network for point cloud oversegmentation. In *ICCV*, 2021. 3, 4, 8

[22] Mariano Jaimez, Mohamed Souiai, Javier Gonzalez-Jimenez, and Daniel Cremers. A primal-dual framework for real-time dense rgb-d scene flow. In *ICRA*, 2015. 2

[23] Chaokang Jiang, Guangming Wang, Yanzi Miao, and Hesheng Wang. 3d scene flow estimation on pseudo-lidar: Bridging the gap on estimating point motion. *TII*, 2022. 1

[24] Zhao Jin, Yinjie Lei, Naveed Akhtar, Haifeng Li, and Munawar Hayat. Deformation and correspondence aware unsupervised synthetic-to-real scene flow estimation for point clouds. In *CVPR*, 2022. 3

[25] Philipp Jund, Chris Sweeney, Nichola Abdo, Zhifeng Chen, and Jonathon Shlens. Scalable scene flow from point clouds in the real world. *RA-L*, 2021. 2

[26] Yair Kittenplon, Yonina C Eldar, and Dan Raviv. Flowstep3d: Model unrolling for self-supervised scene flow estimation. In *CVPR*, 2021. 5, 6, 7

[27] Bing Li, Cheng Zheng, Silvio Giancola, and Bernard Ghanem. Sctn: Sparse convolution-transformer network for scene flow estimation. In *AAAI*, 2022. 2

[28] Bing Li, Cheng Zheng, Guohao Li, and Bernard Ghanem. Learning scene flow in 3d point clouds with noisy pseudo labels. *arXiv preprint arXiv:2203.12655*, 2022. 3

[29] Ruibo Li, Guosheng Lin, Tong He, Fayao Liu, and Chunhua Shen. Hcrf-flow: Scene flow from point clouds with continuous high-order crfs and position-aware flow embedding. In *CVPR*, 2021. 2

[30] Ruibo Li, Guosheng Lin, and Lihua Xie. Self-point-flow: Self-supervised scene flow estimation from point clouds with optimal transport and random walk. In *CVPR*, 2021. 3, 6, 7

[31] Ruibo Li, Chi Zhang, Guosheng Lin, Zhe Wang, and Chunhua Shen. Rigidflow: Self-supervised scene flow learning on point clouds by local rigidity prior. In *CVPR*, 2022. 2, 3, 6, 7, 8

[32] Yangbin Lin, Cheng Wang, Dawei Zhai, Wei Li, and Jonathan Li. Toward better boundary preserved supervoxel segmentation for 3d point clouds. *ISPRS*, 2018. 2, 3

[33] Haisong Liu, Tao Lu, Yihui Xu, Jia Liu, Wenjie Li, and Lijun Chen. Camliflow: Bidirectional camera-lidar fusion for joint optical flow and scene flow estimation. In *CVPR*, 2022. 2

[34] Xingyu Liu, Charles R Qi, and Leonidas J Guibas. Flownet3d: Learning scene flow in 3d point clouds. In *CVPR*, 2019. 1, 2, 6, 7

[35] Xingyu Liu, Mengyuan Yan, and Jeannette Bohg. Meteornet: Deep learning on dynamic 3d point cloud sequences. In *CVPR*, 2019. 2

[36] Yawen Lu, Yuhao Zhu, and Guoyu Lu. 3d sceneflownet: Self-supervised 3d scene flow estimation based on graph cnn. In *ICIP*, 2021. 3

[37] Chenxu Luo, Xiaodong Yang, and Alan Yuille. Self-supervised pillar motion learning for autonomous driving. In *CVPR*, 2021. 1, 3

[38] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *CVPR*, 2016. 1, 6

[39] Moritz Menze, Christian Heipke, and Andreas Geiger. Joint 3d estimation of vehicles and scene flow. *ISPRS*, 2015. 6

[40] Moritz Menze, Christian Heipke, and Andreas Geiger. Object scene flow. *ISPRS*, 2018. 6

[41] Himangi Mittal, Brian Okorn, and David Held. Just go with the flow: Self-supervised scene flow estimation. In *CVPR*, 2020. 3, 7, 8

[42] Bojun Ouyang and Dan Raviv. Occlusion guided scene flow estimation on 3d point clouds. In *CVPRW*, 2021. 3

[43] Bojun Ouyang and Dan Raviv. Occlusion guided self-supervised scene flow estimation on 3d point clouds. In *3DV*, 2021. 5

[44] Jhony Kaesemodel Pontes, James Hays, and Simon Lucey. Scene flow from point clouds with or without learning. In *3DV*, 2020. 3, 7, 8

[45] Gilles Puy, Alexandre Boulch, and Renaud Marlet. Flot: Scene flow on point clouds guided by optimal transport. In *ECCV*, 2020. 2, 3, 5, 6, 7

[46] Yukang Shi and Kaisheng Ma. Safit: Segmentation-aware scene flow with improved transformer. In *ICRA*, 2022. 1

[47] Richard Sinkhorn. A relationship between arbitrary positive matrices and doubly stochastic matrices. *JSTOR*, 1964. 3

[48] Linghua Tang, Le Hui, and Jin Xie. Learning inter-superpoint affinity for weakly supervised 3d instance segmentation. In *ACCV*, 2022. 3

[49] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *ECCV*, 2020. 5

[50] Zachary Teed and Jia Deng. Raft-3d: Scene flow using rigid-motion embeddings. In *CVPR*, 2021. 2

[51] Ivan Tishchenko, Sandro Lombardi, Martin R Oswald, and Marc Pollefeys. Self-supervised learning of non-rigid residual flow and ego-motion. In *3DV*, 2020. 3, 6, 7

[52] Sundar Vedula, Simon Baker, Peter Rander, Robert Collins, and Takeo Kanade. Three-dimensional scene flow. In *ICCV*, 1999. 2

[53] Guangming Wang, Yunzhe Hu, Zhe Liu, Yiyang Zhou, Masayoshi Tomizuka, Wei Zhan, and Hesheng Wang. What matters for 3d scene flow network. In *ECCV*, 2022. 2

[54] Guangming Wang, Yunzhe Hu, Xinrui Wu, and Hesheng Wang. Residual 3d scene flow learning with context-aware feature extraction. *TIM*, 2022. 2

[55] Guangming Wang, Xinrui Wu, Zhe Liu, and Hesheng Wang. Hierarchical attention learning of scene flow in 3d point clouds. *TIP*, 2021. 2

[56] Haiyan Wang, Jiahao Pang, Muhammad A Lodhi, Yingli Tian, and Dong Tian. Festa: Flow estimation via spatial-temporal attention for scene point clouds. In *CVPR*, 2021. 1, 2

[57] Ke Wang and Shaojie Shen. Estimation and propagation: Scene flow prediction on occluded point clouds. *RA-L*, 2022. 1

[58] Zirui Wang, Shuda Li, Henry Howard-Jenkins, Victor Prisacariu, and Min Chen. Flownet3d++: Geometric losses for deep scene flow estimation. In *CVPR*, 2020. 2

[59] Yi Wei, Ziyi Wang, Yongming Rao, Jiwen Lu, and Jie Zhou. Pv-raft: Point-voxel correlation fields for scene flow estimation of point clouds. In *CVPR*, 2021. 1, 2, 5

[60] Wenxuan Wu, Zhi Yuan Wang, Zhuwen Li, Wei Liu, and Li Fuxin. Pointpwc-net: Cost volume on point clouds for (self-) supervised scene flow estimation. In *ECCV*, 2020. 1, 3, 6, 7

[61] Victor Zuanazzi, Joris van Vugt, Olaf Booij, and Pascal Mettes. Adversarial self-supervised scene flow estimation. In *3DV*, 2020. 3