IHNet: Iterative Hierarchical Network Guided by High-Resolution Estimated Information for Scene Flow Estimation

Yun Wang* Cheng Chi* Min Lin Xin Yang✉
Hubei Key Laboratory of Smart Internet, School of Electronic Information and Communications, Huazhong University of Science and Technology, Wuhan
{wangyun, D202281113, xinyang2014}@hust.edu.cn chengchi.hust@gmail.com

Abstract

Scene flow estimation, which predicts the 3D displacements of point clouds, is a fundamental task in autonomous driving. Most methods have adopted a coarse-to-fine structure to balance computational efficiency with accuracy, particularly when handling large displacements. However, inaccuracies in the initial coarse layer's scene flow estimates may accumulate, leading to incorrect final estimates. To alleviate this, we introduce a novel Iterative Hierarchical Network——IHNet. This approach circulates high-resolution estimated information (scene flow and feature) from the preceding iteration back to the low-resolution layer of the current iteration. Serving as a guide, the high-resolution estimated scene flow, instead of initializing the scene flow from zero, provides a more precise center for low-resolution layer to identify matches. Meanwhile, the decoder's feature at the high-resolution layer can contribute essential movement information. Furthermore, based on the recurrent structure, we design a resampling scheme to enhance the correspondence between points across two consecutive frames. By employing the previous estimated scene flow to fine-tune the target frame's coordinates, we can significantly reduce the correspondence discrepancy between two frame points, a problem often caused by point sparsity. Following this adjustment, we continue to estimate the scene flow using the newly updated coordinates, along with the reencoded feature. Our approach outperforms the recent state-of-the-art method WSAFlowNet by 20.1% on FlyingThings3D and 56.0% on KITTI scene flow datasets according to EPE3D metric. The code is available at https://github.com/wangyunlhr/IHNet.

1. Introduction

Scene flow estimation on point clouds, which predicts point-wise 3D motions from two consecutive point clouds, is a fundamental task in autonomous driving applications and robotics. It supplies spatiotemporal motion and matching information for a range of high-level tasks such as multi-object tracking [36, 30], pose estimation [2], motion segmentation [20, 5, 4], etc. In recent years, due to advancements in deep learning, neural networks have emerged as a popular method for processing point clouds. FlowNet3D [16] proposed an end-to-end scene flow estimation approach that leveraged PointNet++ [22] and introduced two new learnable layers, namely flow embedding layer and set upconv layer. However, the capacity of the flow embedding layer (cost volume) is determined by the matching search region, which is constrained to a single resolution level due to the computational cost.

To strike a balance between accuracy and computational efficiency while accounting for large displacements, numerous methods employed a coarse-to-fine structure for the scene flow estimation task, such as PointPWC-Net [35] and HALFlow [29]. The coarse-to-fine structure is a classical "encoder-decoder" architecture, consisting of multiple layers. The pipeline first constructs feature pyramid by encoding the input and estimates at the coarsest resolution to capture large displacements. Layer-by-layer refinement is then performed from the coarser to finer level. During the refinement stage, once the scene flow has been estimated at a coarser level, it is up-sampled and then passed to the finer level by warping. That is, the search center for matching is determined based on the previous estimated results. What's more, the estimated information from coarser level will be used to estimate scene flow for finer level.

However, early estimation can lead to errors and result in the search area missing the true matching point, yielding inaccurate scene flow estimation. To address this, we propose an Iterative Hierarchical Network (IHNet) based on the observation that the accuracy of downsampled estimation results from finer to coarser level is significantly higher than that of original coarser level results (initialized from zero), This is shown in Fig. 1. IHNet aims to mitigate the error caused by mismatching in coarser resolution by propagating

* Equal contribution. ✉ Corresponding author.

the high-resolution estimated information from the previous iteration to low-resolution levels of the current iteration. In this way, IHNet leverages higher accuracy results from the finer level at the previous iteration, improving performance through each iteration.

Another significant challenge in scene flow estimation task is the issue of poor correspondence between two consecutive frames, stemming from the inherent sparsity of point clouds. Due to the absence of an exact point-to-point correspondence, matching errors arise, ultimately resulting in inaccurate scene flow estimation. To address this issue, some methods [3, 1] designed novel cost volumes, such as point-to-patch, patch-to-patch cost volume. By matching larger regions (i.e. patches) instead of individual points, the influence of this challenge can be mitigated to a certain extent. Focusing on the irregular data structure, RCP [9] designed a two-stage recurrent network. They first utilized a point-wise optimization to extract the regular information. Subsequently, the information was input into a recurrent network based on GRU [23] for further regularization.

To address poor correspondence problem, we adopt another solution, which is to adjust the point coordinates. Based on the iterative hierarchical structure, we propose a novel resampling scheme to modify the coordinates of the points applied to the second iteration and beyond. We first use the estimated scene flow to warp the source frame points, and then find the nearest neighbor of warped points in the target frame to regard as the new target frame point coordinates. Subsequent processing is performed based on the features encoded from the new points. The resampling scheme is effective because it eliminates the points in the target frame that are not particularly relevant to the source frame.

In summary, the main contributions as follows:

- We design an iterative hierarchical network guided by high-resolution estimated information based on a coarse-to-fine structure WSAFlowNet [32]. It leverages the high accuracy estimated information and improves the performance during each iteration.
- To address the problem of poor correspondence between two adjacent frames, we propose a resampling scheme to adjust point clouds' coordinates and recode the features as the input.
- Our proposed network achieves state-of-the-art performance on the FlyingThings3D and KITTI benchmarks.

2. Related work

Deep learning on Point Clouds. Recently, deep learning methods have been heavily employed on point cloud data.

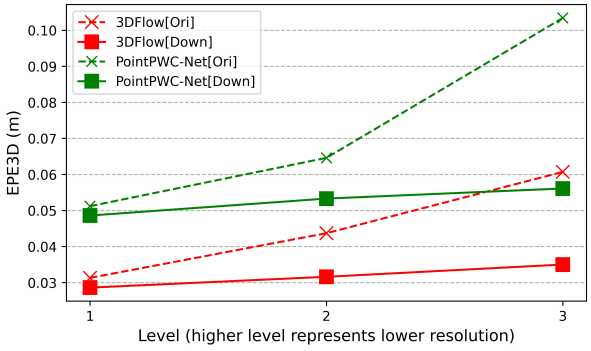

Figure 1. **Downsampled vs. Original scene flow.** We select two coarse-to-fine methods, PointPWC-Net [35] and 3DFlow [27], for demonstration. "Down" represents the downsampled scene flow results from level l_0 estimation to the corresponding resolution, while "Ori" represents the original results obtained at each resolution during the layer-by-layer estimation process. (Lower EPE3D means higher accuracy.)

Some recent works [1, 15, 28, 11] have directly used raw point clouds as network inputs. Certain works can be categorized as convolution-based methods. For instance, PointConv [34] proposed a 3D convolution kernel comprised of inverse density and weight functions in the local region, meaning that the non-uniformly sampled point clouds were re-weighted using an inverse density function. Other works can be described as Point-wise methods, using Multi-Layer Perceptron(MLP) and Maxpooling to extract the features of point clouds. For example, PointNet++ [22] introduced a hierarchical structure of set abstraction, and improved the robustness of feature learning on non-uniform density point clouds through a density-adaptive PointNet layer. This paper uses PointNet++ to obtain an excellent point cloud feature representation.

Point-based Scene Flow estimation. The concept of 3D scene flow was initially introduced by [26]. Since then, numerous works have focused on estimating scene flow using point cloud data. Some studies [6, 24, 25] have introduced traditional methods, such as those based on energy minimization. Conversely, other works [31, 16, 13, 8, 7] have taken an end-to-end learning approach to estimate scene flow. For instance, FlowNet3D [16] introduced a flow embedding layer to encode the motion of point clouds, where the scene flow can be learned. In cases of large displacement scenarios, coarse-to-fine network architectures[35, 3, 32] are indispensable. PointPWC-Net [35] proposed a novel patch-to-patch cost volume to gain a more robust and stable cost volume. Bi-PointFlowNet [3] used a bidirectional flow embedding layer to learn features of both forward and backward flows simultaneously, with the aim of leveraging contextual information for more accurate estimation. WSAFlowNet [32] proposed a weight-sharing aggregation

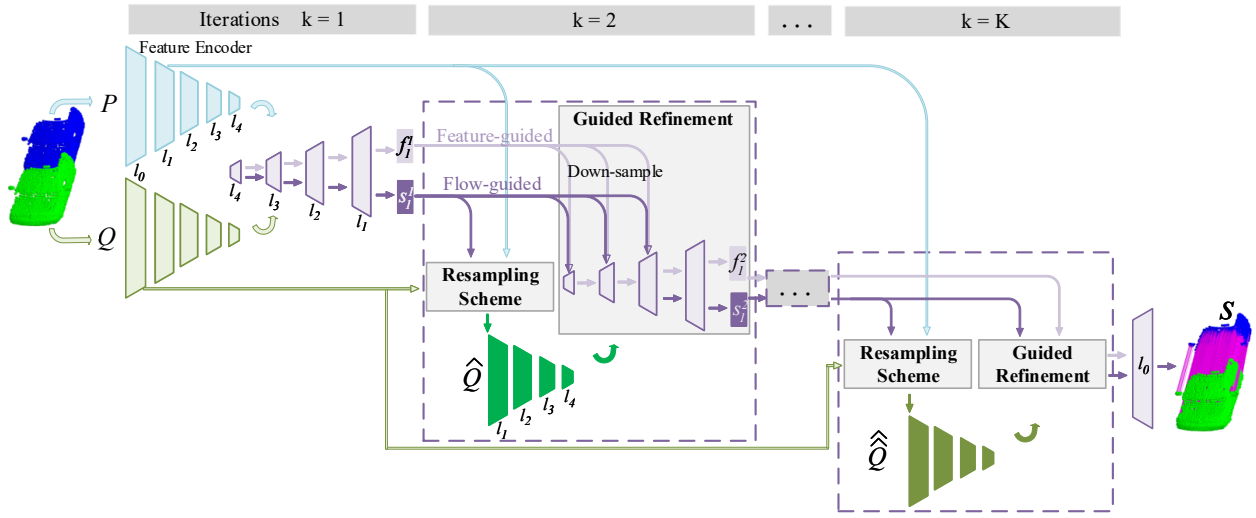

Figure 2. **IHNet Architecture.** (1) In the first iteration, the feature encoder (depicted blue and green) extracts multi-level features from the original point clouds P and Q. Subsequently, scene flow estimation (indicated in purple) is performed on these features, with refinement conducted at each layer up to the l_1 level. (2) In each subsequent iteration, the previously estimated scene flow s_1^{k-1} at the l_1 level is used to modify the coordinates Q via a resampling scheme (elaborated in § 3.3 and Fig. 3). Following this, features are extracted from the adjusted coordinates $\widehat{Q}$, and scene flow estimation is conducted based on P and $\widehat{Q}$. (3) Additionally, the estimated scene flow s_1^{k-1} and feature f_1^{k-1} from the previous iteration are down-sampled and employed as a guide for the current iteration, named guided refinement.

method at the level of feature and scene flow, which made full use of the local rigidity. In this paper, we also adopt the coarse-to-fine structure, incorporating a recurrent scheme to utilize refined flow information at a high resolution.

Correspondence on Point Cloud. Due to the sparsity of points, the correspondence between two consecutive frames is poor, leading researches to devote efforts to finding a soft correspondence. Previous work [9] demonstrated that accurate correspondence within two frames point clouds could lead to higher precision in scene flow estimation. 3DFlow [27] achieved more reliable matching by proposing a reverse reliability layer, which jointly learned backward constraints and an all-to-all flow embedding. FLOT [21] presented a method that utilized an optimal transmission tool to find the corresponding relationships in graph matching. RCP [9] employed a GRU iteration to detect soft correspondence in the target region. Different from these approaches, this paper introduces a new resampling method to enhance the correlation between the local areas of the source and target point clouds by reducing the number of mismatched points.

3. Method

3.1. Problem definition

Our objective is to estimate the scene flow between two consecutive frames of point clouds, denoted as the source frame $P = \{p_i \in \mathbb{R}^3\}_{i=1}^{N_1}$ at time t and the target frame $Q = \{q_i \in \mathbb{R}^3\}_{i=1}^{N_2}$ at time $t+1$. The scene flow for each point $p_i \in P$ is estimated, denoted by $S = \{s(p_i) \in \mathbb{R}^3\}_{i=1}^{N_1}$. The correspondence between point clouds P and Q is poor because of the inherent sparsity of the point clouds, which leads to errors for matching.

We introduce an Iterative Hierarchical Network, which employs both iterative and hierarchical refinement. To simplify the notation, we define that the superscript will denote the iteration number, while the subscript will refer to the hierarchy level for both feature and scene flow representation.

3.2. Iterative Hierarchical Network

We propose IHNet, a novel iterative hierarchical network, inspired by the discernible fact that the accuracy of estimations in high-resolution layers surpasses that low-resolution layers, owing to the layer-by-layer refinement. Consequently, to achieve good performance, we propagate high-resolution estimated information in the preceding iteration to low-resolution layer in the current iteration. Utilizing state-of-the-art coarse-to-fine structure WSAFlowNet [32] as the baseline, we proceed to construct the iterative network. A detailed explanation of our approach is provided in the following part.

Baseline. WSAFlowNet is a coarse-to-fine structure that is equipped with the Weight-Sharing Aggregation module (WSA) and the Deformation Degree (DD) module. It begins by utilizing Farthest Point Sampling for downsampling, followed by the construction of a feature pyramid based on PointNet++ [22]. The pyramid comprises five levels, denoted as $\{l_0 - l_4\}$,with l_0 representing the high-

est resolution. Then scene flow is estimated at the coarsest level, involving warping layer, cost volume construction and scene flow estimation. The warping layer serves to adjust the search center, using up-sampled estimated scene flow from the coarser level (the initial scene flow is set to zero at the coarsest level). Following this, a cost volume (CV) is built to represent the matching information between the source point clouds P and the warped target point clouds $Q^w = Q - S_I$, where S_I symbolizes the estimated scene flow based on distance interpolation. Furthermore, a deformation degree module δ_{DD} is introduced to measure the local structure differences between the source P and the warped source point clouds $P^w = P + S$. Finally, the scene flow estimator acts as a decoder to obtain the movement feature and generate the estimated scene flow. Details of the estimator's formulation are presented below.

$$f_{l-1} = \mathrm{MLP}\left(\left[\mathrm{f(P)}, \mathrm{CV}, \delta_{DD}, \mathrm{up}\left(f_l\right), \mathrm{up}\left(s_l\right)\right]\right) \quad (1)$$

$$s_{l-1} = FC\left(f_{l-1}\right) \quad (2)$$

where f_{l-1} denotes the output feature of the estimator, capturing the movement information at the $l-1$ level. $[\cdot,\cdot]$ indicates concatenation operation. s_{l-1} denotes the estimated scene flow at that level. FC refers to a fully connected layer.

For refinement, the estimator's output feature and scene flow are propagated through WSA up-sampling, from coarser to finer level. The above processes are carried out layer by layer, continuing until it reaches the highest resolution level.

Note that, to reduce the computational cost, we have streamlined some processes in our method (elaborated in § 4.4 **Baseline-S**).

IHNet. There are several iterative refinement network based on GRU for scene flow estimation task, such as Flow-step3D [12], PV-RAFT [33]. They take advantage of the preceding iteration estimated results as the input of current iteration updater to refine. However, there are very few pyramid-based iterative methods. Primarily because a simple iteration based on the coarse-to-fine structure, where the output of the previous iteration serves as the input for the current one, can lead to significant computational overhead without substantial improvement in accuracy. This issue is verified through our experiments, as shown in Tab. 2. Hence, based on the observation that the accuracy of high-resolution estimated results is superior over low-resolution, we fully incorporate the estimated information of high-resolution into the design of the iterative network, as shown in Fig. 2. The details of the entire network are elaborated below.

To reduce computational costs, the network performs iterations between l_4 and l_1. After obtaining an initial estimated scene flow in the first iteration, we down-sample

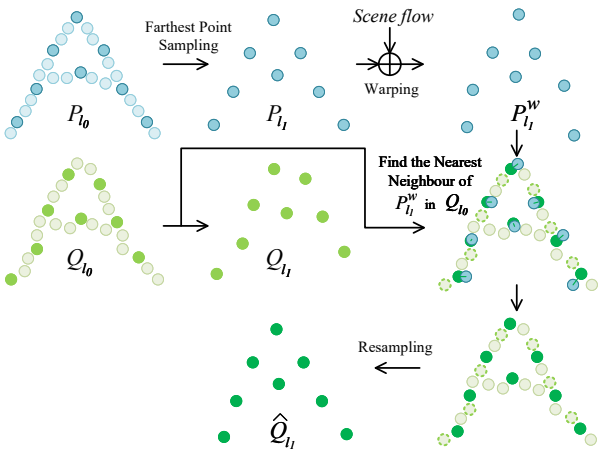

Figure 3. **Resampling scheme.** Using the scene flow S from the previous iteration, we obtain the warped source frame $P^w = P + S$, taking the warped points as centers. We then search for the nearest neighbors $N_1()$ in the target frame Q based on distance, resulting in the adjusted points $\widehat{Q} = N_1\left(P^w\right)$.

the scene flow from high-resolution l_1 to low-resolution levels $\{l_4 - l_2\}$, which are then utilized to operate warping at each level. The high-resolution scene flow, estimated from the preceding iteration, provides a more accurate matching search center for the low-resolution current iteration, making the search region more likely to cover the target matching point. Besides down-sampling scene flow, we also down-sample the estimator's output feature from l_1 to $\{l_4 - l_2\}$. These feature, coming from the previous iteration, are used as input to the estimator at each level, supplying more comprehensive movement information. Therefore, we have made modifications to the estimator, expressed as follows.

$$f_{l-1}^k = \mathrm{MLP}\left(\left[\mathrm{f(P)}, \mathrm{CV}\left[\mathrm{down}\left(s_1^{k-1}\right)\right], \mathrm{up}\left(f_l^k\right), \mathrm{down}\left(f_1^{k-1}\right), \mathrm{down}\left(s_1^{k-1}\right)\right]\right) \quad (3)$$

where f_{l-1}^k denotes the output feature of the estimator at the $l-1$ level in the k iteration. s_1^{k-1} denotes the estimated scene at the high-resolution l_1 level in the $k-1$ iteration. $\mathrm{CV}\left[\mathrm{down}\left(s_1^{k-1}\right)\right]$ represents cost volume construction based on the down-sampled scene flow s_1^{k-1}. $\mathrm{down}()$, $\mathrm{up}()$ represent down-sampling and up-sampling operations respectively.

By inputting $\mathrm{up}\left(f_l^k\right)$ into the estimator, we achieve the cross-level refinement. And through the utilization of $\mathrm{down}\left(f_1^{k-1}\right)$ from the preceding iteration, cross-iteration refinement is accomplished. The implementation of these features leads to a larger receptive field and more detailed information being introduced.

3.3. Resampling scheme

Due to the sparsity of point cloud data, the correspondence between P and Q is poor, which introduces errors for matching. And then it results in the incorrect estimated scene flow. To alleviate the problem of poor correspondence, recent methods employ region-based matching rather than point-to-point matching. In this way, they can find the region that contain the true target point. However, while constructing the cost volume, all points in the searching region will be involved in the computation, leading to introduce errors from irrelevant points. An alternative approach would be to adjust the coordinates of the input target point clouds directly. Thanks to the iterative hierarchical network, we are allowed for the adjustment of point coordinates in the current iteration by using the estimated scene flow from the preceding iteration. Fig. 3 demonstrates the realization of resampling scheme. The details are introduced as follows.

First, we obtain the warped source point clouds ($P^w = P + S$) by using the estimated scene flow at the high-resolution in the previous iteration. Second, we select the nearest neighbor points of the warped point clouds in the target frame and regards them as the new target point clouds $\widehat{Q}$. Finally, we re-construct feature pyramid $f(\widehat{Q})$ for the new target point clouds and execute the scene flow estimation between P and $\widehat{Q}$. Beginning with the second iteration, we modify the point coordinates in every following iteration.

The resampling scheme removes the points of the target frame that are not particularly relevant to the source frame, which improves the poor correspondence between two adjacent frames.

Comparison to related work. In the method [19], anchor points were designed to improve the self-supervised loss and mitigate the impact of incorrect forward flow predictions on the backward flow estimation. However, this did not enhance the point correspondence between two consecutive raw frames. In contrast, our resampling scheme adjusts the point coordinates of the raw frame and performs feature extraction from them for the following forward scene flow estimation. This effectively improves the point-to-point correspondence of two consecutive raw frames, making the estimated flow more accurate.

3.4. Loss

Following previous methods [35, 32], we adopt multi-scale loss L^k for each iteration. L^k can be expressed as:

$$L^k = \sum_{l=0}^{L} \gamma_l \sum_{i=1}^{N_l} \left\| \widehat{s_l^k}(p_i) - s_l^k(gt) \right\|_2 \tag{4}$$

where γ_l represents the weight for each level l. The weights are set as $\gamma_0 = 0.02$, $\gamma_1 = 0.04$, $\gamma_2 = 0.08$, $\gamma_3 = 0.16$, $\gamma_4 = 0.16$. The predicted scene flow is denoted by $\widehat{s_l^k}$. And $\| * \|_2$ refers to the L2-norm.

In addition, we supervise the estimated results for each iteration. The overall loss L_{all} is a weighted iteration loss, which the weights increase exponentially with each iteration.

$$L_{all} = \sum_{k=1}^{M} \gamma^{M-k} L^k \tag{5}$$

Where γ is set to 0.8.

4. Experiments

4.1. Datasets and Evaluation Metrics

Datasets. We assess the performance of our proposed method using both the synthetic dataset FlyingThings3D [17] and the real scene dataset KITTI Scene Flow [18] the same as [14, 3, 1, 32]. FlyingThings3D dataset contains 19,640 pairs in training set and 3,824 pairs in the test set. Our data preprocessing is based on HPLFlowNet [10]. We use disparity and optical flow to create point cloud data and then filter out any points with a depth greater than $35m$. KITTI dataset consists of 200 pairs in training set and 200 pairs in the test set. We also preprocess the dataset following HPLFlowNet [10] and generate 142 pairs point clouds in the training set, as the disparity information for the test set is unavailable. Ground points $(height < 0.3m)$ are removed.

Evaluation Metrics. For an equitable comparison, we evaluate the scene flow by following metrics, the same as [35, 3, 1, 32, 14].

•*EPE3D*: $\left\| \widehat{S}^l - S_{gt}^l \right\|_2$ the averaged end point error per point, in meters.

• *Acc3DS*: the proportion of points where *EPE3D* $< 0.05m$ or relative error $< 5\%$.

• *Acc3DR*: the proportion of points with *EPE3D* $< 0.1m$ or relative error $< 10\%$.

• *Outliers3D*: the percentage of points where *EPE3D*$>$ $0.3m$ or relative error $> 10\%$.

• *EPE2D*: the average 2D end point error, derived by projecting onto the image plane.

• *Acc2D*: the ratio of points where *EPE2D* $< 3px$ or relative error $< 5\%$.

4.2. Experimental Setup

We conduct experiments using NVIDIA RTX 3090 GPUs. We train the model on the FlyingThings3D training dataset, and then evaluate on the FlyingThings3D test dataset and KITTI dataset, to verify the effectiveness and generalization ability. We unroll k=3 iterations for training and use as input two adjacent frame point clouds containing only point coordinates. The input is determined by randomly sampling and the size is set to 8192 points, with a

Dataset	Method	EPE3D(m)↓	Acc3D Strict↑	Acc3D Relax↑	Outliers3D↓	EPE2D↓	Acc2D↑
FlyingThings3D [17]	PointPWC-Net[35]	0.0588	0.7379	0.9276	0.3424	3.2390	0.7994
	PV-RAFT[33]	0.0461	0.8169	0.9574	0.2924	-	-
	FlowStep3D[12]	0.0455	0.8162	0.9614	0.2165	-	-
	RCP[9]	0.0403	0.8567	0.9635	0.1976	-	-
	Bi-PointFlowNet[3]	0.0280	0.9180	0.9780	0.1430	1.5820	0.9290
	3DFlow [27]	0.0281	0.9290	0.9817	0.1458	1.5229	0.9279
	WSAFlowNet [32]	0.0239	0.9391	0.9821	0.1103	1.3703	0.9358
	Ours	**0.0191**	**0.9601**	**0.9865**	**0.0715**	**1.0918**	**0.9563**
KITTI [18]	PointPWC-Net[35]	0.0694	0.7281	0.8884	0.2648	3.0062	0.7673
	PV-RAFT[33]	0.0560	0.8226	0.9372	0.2163	-	-
	FlowStep3D[12]	0.0546	0.8051	0.9254	0.1492	-	-
	RCP[9]	0.0481	0.8491	0.9448	0.1228	-	-
	Bi-PointFlowNet[3]	0.0300	0.9200	0.9600	0.1410	1.0560	0.9490
	3DFlow[27]	0.0309	0.9047	0.9580	0.1612	1.1285	0.9451
	WSAFlowNet [32]	0.0277	0.9209	0.9613	0.1350	0.9773	0.9574
	Ours	**0.0122**	**0.9779**	**0.9892**	**0.0913**	**0.4993**	**0.9862**

Table 1. **Quantitative results on FlyingThings3D and KITTI Scene Flow datasets.** All listed approaches are only trained on FlyingThings3D training dataset in a fully-supervised manner. Bold indicates the optimal outcomes.

Baseline-S	Basic-iter	Flow-guided	Feature-guided	Resampling scheme	EPE3D(m)↓	Acc3D Strict↑	Acc3D Relax↑	Outliers3D↓	EPE2D↓	Acc2D↑	Time(ms)
✓					0.0272	0.9260	0.9792	0.1379	1.5408	0.9235	**70.46**
✓	✓				0.0265	0.9278	0.9789	0.1317	1.4976	0.9248	106.86
✓	✓	✓			0.0247	0.9392	0.9806	0.1129	1.4198	0.9364	105.77
✓	✓	✓	✓		0.0200	0.9579	0.9857	0.0823	1.1390	0.9543	110.71
✓	✓	✓	✓	✓	**0.0191**	**0.9601**	**0.9865**	**0.0715**	**1.0918**	**0.9563**	130.91

Table 2. **Ablation studies on FlyingThings3D dataset.** "✓" denotes using this strategy. Bold indicates the optimal outcomes.

batch size of 8. To speed up, we adopt a two-stage training process. In the first stage, our model is trained on a quarter of the training set (4910 pairs), with a learning rate set to 0.001 and a decay rate of 0.7 for every 20 epochs. This preliminary training phase continues for 100 epochs. In the second stage, after loading the pre-trained weights, the model is fine-tuned on the complete training set. The learning rate is set to 0.00024, and the decay rate is the same as before. The fine-tuning extends for 160 epochs. The parameters of Adam optimizer are set to $\beta_1 = 0.9$, $\beta_2 = 0.99$, weight decay = 0.0001. Besides conducting ablation experiments regarding the number of iterations, we evaluate the results of the experiment with k=3 iterations. And to verify the generalization ability of our method, we test the model on the KITTI dataset without fine-tuning.

4.3. Evaluation on FlyingThings3D and KITTI

As shown in Tab. 1, our method outperforms recent SOTA work on all evaluation metrics on FlyingThings3D dataset, thereby demonstrating the effectiveness of our proposed method. Our proposed method achieves a 20.1% improvement over recent SOTA method WSAFlowNet [32] on EPE3D metric. To verify the generalization ability, we evaluate our model without fine-tuning on KITTI Scene Flow dataset. Our method outperforms WSAFlowNet by 56.0% on EPE3D metric on KITTI dataset. It proves that our method is highly effective in real-world scenarios. In summary, utilizing the Guided Refinement and Resampling Scheme achieves great performance on both synthetic and real scene datasets. The visualization results (Fig. 7) demonstrate that our method achieves higher accuracy than recent SOTA methods [32, 3] on FlyingThings3D and KITTI datasets. We also provide the local details for easy observation. Despite encountering challenging areas, we still realize favorable results, as indicated by fewer red points, which correspond to fewer errors.

4.4. Ablation Studies

Baseline-S. We adopt WSAFlowNet [32] as our baseline. In order to reduce computational expense, we streamline certain processing steps. Specifically, the estimator's Multilayer Perceptrons (MLPs) have been simplified from an original configuration of five layers [128, 128, 96, 64, 64] down to just two layers [96, 64]. Additionally, we remove one MLP layer within the feature encoder and eliminate the deformation degree module. The baseline-S (Parameters reduced from $3.7M$ to $2.0M$) experimental results are shown in Tab. 2.

Basic-iter. We conduct an experiment on a basic iteration (k=3) that employs Baseline-S structure. It propagates the

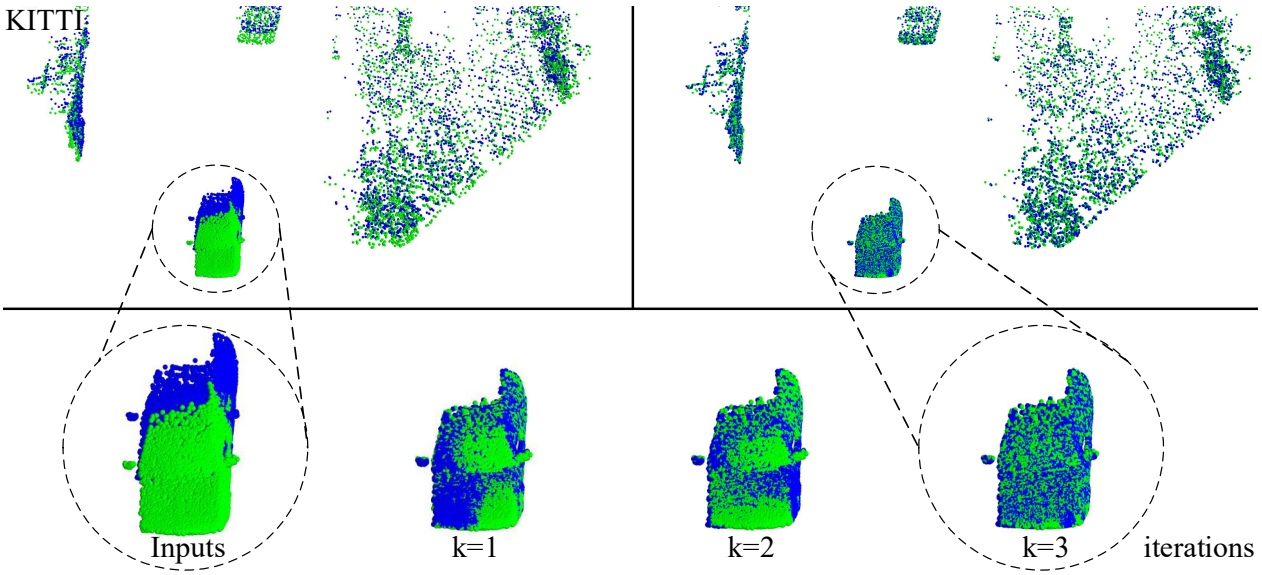

Figure 4. **Visualization results of iteration k.** On the left side of the upper part, it shows the original inputs, with blue points representing the source frame and green points representing the target frame. On the right side of the upper part, it shows the fitting degree between the warped source frame based on the estimated scene flow and the target frame. The higher the fitting degree, the higher the accuracy of the estimated scene flow. The lower part shows that as the number of iterations increases, the fitting degree between the warped source frame and the target frame becomes better and better.

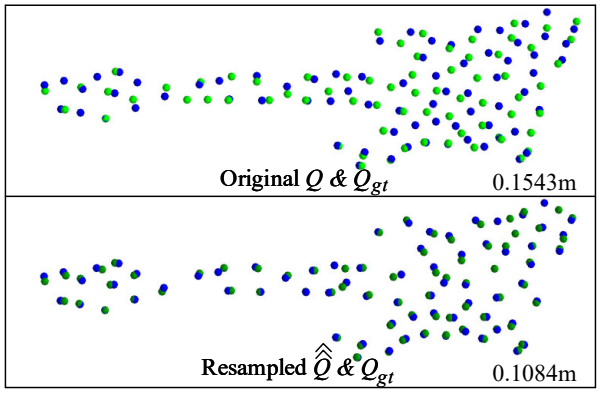

Figure 5. **Visualization results of resampling scheme.** Blue points is the Q_{g_t}, Green points is the original points Q, Deep green points is the resampled points $\hat{\hat{Q}}$. The average distance between two sets of points is denoted by the number located in the lower right corner of each row.

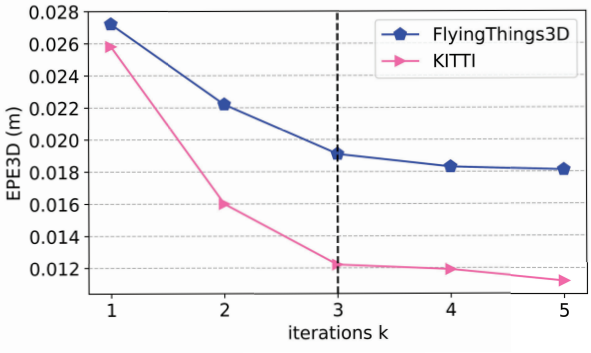

Figure 6. **Iteration k.** The model is trained on the FlyingThings3D training set with iteration k=3. We evaluate the model on both the FlyingThings3D and KITTI test set with different k. As the iteration number increases, the EPE3D metric gradually decreases.

estimated output from the finer layer l_1 in the preceding iteration to the coarsest layer l_4 in the current iteration. This is achieved by using the previously estimated scene flow for warping in the current iteration. As illustrated in the first and second rows of Tab. 2, this basic cross-iteration refinement leads to a slight improvement in performance, though it comes at the cost of taking more time.

Flow-guided. We down-sample the estimated scene flow from the finer layer l_1 to the coarser layers $\{l_4 - l_2\}$ for warping. By doing so, the higher accuracy result information is passed to the following iteration. Compared to the Basic-iter structure, this method outperforms it by 6.8% on EPE3D metric, without incurring any additional computational cost. This proves that transmitting high-resolution estimated information to all low-resolution levels is more effective.

Feature-guided. The output feature of the estimator, which represents movement information, is critical for subsequent processes. In addition to flow-guided design, we introduce feature-guided. We down-sample both the esti-

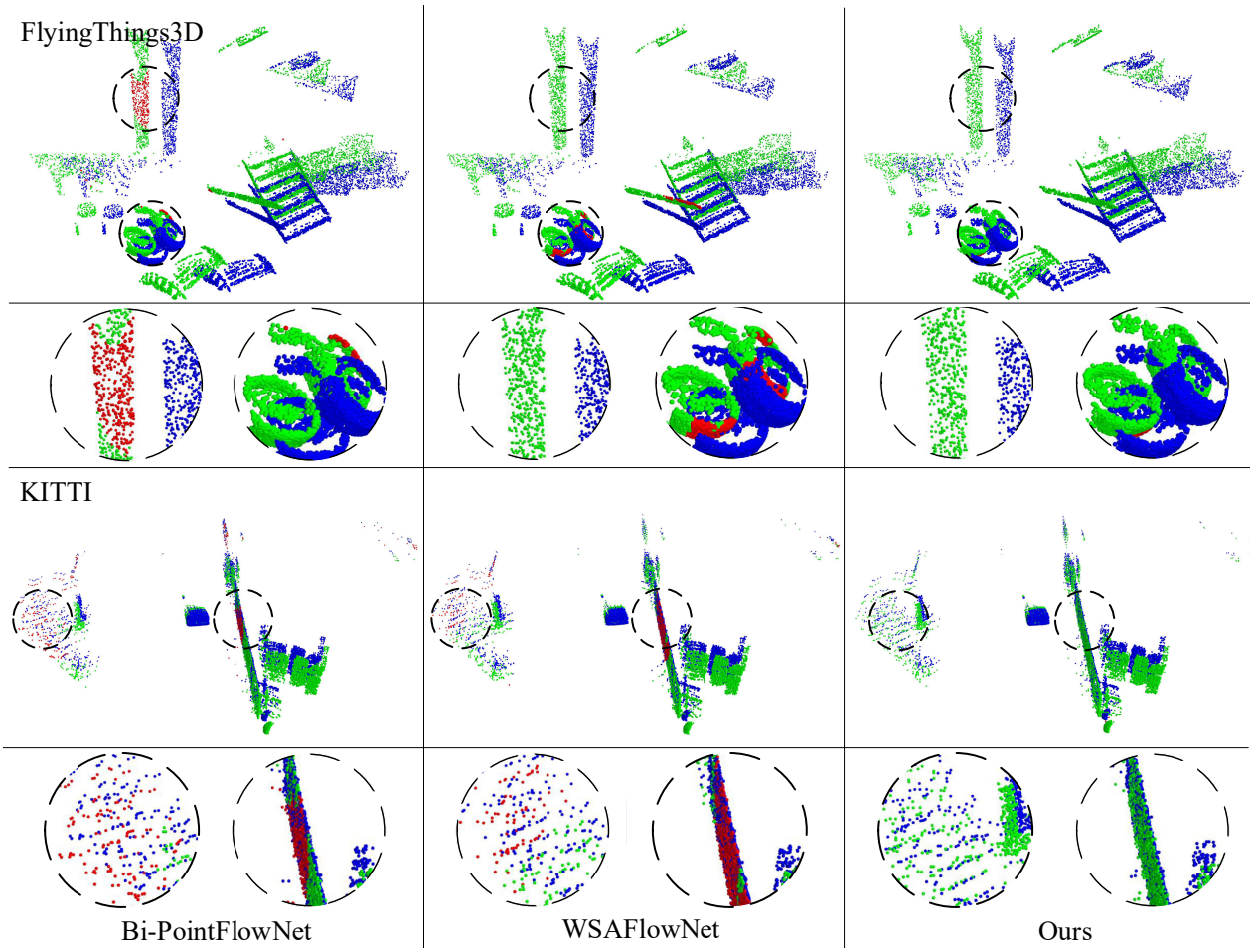

Figure 7. **Visualization Results on FlyingThings3D and KITTI scene flow datasets.** From left to right, it's Bi-PointFlowNet, WSAFlowNet, and our approach. The first two rows present the experimental results on the FlyingThings3D dataset, while the last two rows showcase the results on KITTI dataset. Blue points represent the source points. Green points indicate the warped source points utilizing the correct predicted scene flow according to the *Acc3DS* metric. Red points correspond to the warped source points utilizing the incorrect predicted scene flow according to the *Acc3DS* metric.

mated feature and scene flow from the finer layer l_1 to the coarser layers $\{l_4 - l_2\}$. The finer layer's feature can supply detailed movement information. From the comparison between rows three and four in Tab. 2, it becomes clear that the incorporation of feature-guided results in a substantial improvement in accuracy. It outperforms only flow-guided by 19.0% on EPE3D metric.

Resampling scheme. To tackle the problem of poor correspondence, we propose a resampling scheme. Utilizing this scheme, we achieve a certain degree of improvement even within our outstanding design (incorporates both flow-guided and feature-guided). Specifically, we see a further 4.5% enhancement on the EPE3D metric, demonstrating the effectiveness of adjusting target point coordinates based on the estimated scene flow.

To better illustrate the effectiveness of our resampling scheme, we provide a visualization of the resampled point coordinates in Fig. 5. We set $Q_{gt} = P + S_{gt}$ as the reference standard, which exactly corresponds to P. The average distance between the original Q point coordinates and Q_{gt} is $0.1543m$. In contrast, the average distance between the resampled $\widehat{\widehat{Q}}$ point coordinates and Q_{gt} is reduced to 0.1084m. Thus, our method significantly alleviates the poor correspondence between the two frame point clouds.

Number of iterations. Although we unroll three iterations for training, we evaluate the experiments result with different k iterations for testing. As shown in the Fig. 6, the experimental results on both FlyingThings3D and KITTI dataset improve with an increasing number of iterations. After more than three iterations, the degree of improvement in the experimental results decreases. Therefore, considering the computational cost and accuracy, we use the experi-

mental model with k=3 iterations as our final model, which exhibits good performance and generalization ability. As demonstrated in the Fig. 4, the warped point P^w and Q become progressively more aligned as the number of iterations increases, which proves the improvement of experimental performance.

5. Conclusion

We propose an Iterative Hierarchical Network based on the observation that the accuracy of estimated results in the finer layer is higher than that in the coarser layer. Consequently, we regard the high-resolution estimated information from the preceding iteration as a guide to link the following iteration. We propagate the estimated scene flow from the finer layer to the coarser layer, aiming to determine a more accurate matching center, and we also propagate the output feature of the estimator to provide additional movement information. In this way, the network achieves cross-layer and cross-iteration refinements. Furthermore, to address the poor correspondence problem between two frame input point clouds, we design a resampling scheme. We use the estimated scene flow to update the target frame's point coordinates. Compared to the raw points, the new target points are more relevant to the source points, thereby alleviating the errors introduced by the inputs. We then conduct the estimating operation based the adjusted point clouds. Extensive experimental results on the FlyingThings3D and KITTI scene flow datasets demonstrate that our IHNet achieves state-of-the-art performance.

Acknowledgements. This work was supported by the National Natural Science Foundation of China under Grants 62061160490, 62122029, and U20B2064.

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 *2022 International Conference on Robotics and Automation (ICRA)*, pages 883–889, 2022.

[2] Aseem Behl, Despoina Paschalidou, Simon Donne, and Andreas Geiger. Pointflownet: Learning representations for rigid motion estimation from point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[3] Wencan Cheng and Jong Hwan Ko. Bi-pointflownet: Bidirectional learning for point cloud based scene flow estimation. In *European Conference on Computer Vision*, pages 108–124. Springer, 2022.

[4] Cheng Chi, Peiliang Li, Xiaozhi Chen, and Xin Yang. Puamos: End-to-end point-wise uncertainty weighted aggregation for moving object segmentation. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 12456–12463. IEEE, 2022.

[5] David Deng and Avideh Zakhor. Rsf: Optimizing rigid scene flow from 3d point clouds without labels. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1277–1286, 2023.

[6] Ayush Dewan, Tim Caselitz, Gian Diego Tipaldi, and Wolfram Burgard. Rigid scene flow for 3d lidar scans. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1765–1770. IEEE, 2016.

[7] Guanting Dong, Yueyi Zhang, Hanlin Li, Xiaoyan Sun, and Zhiwei Xiong. Exploiting rigidity constraints for lidar scene flow estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12776–12785, June 2022.

[8] Zan Gojcic, Or Litany, Andreas Wieser, Leonidas J. Guibas, and Tolga Birdal. Weakly supervised learning of rigid 3d scene flow. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5692–5703, June 2021.

[9] Xiaodong Gu, Chengzhou Tang, Weihao Yuan, Zuozhuo Dai, Siyu Zhu, and Ping Tan. Rcp: Recurrent closest point for point cloud. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8216–8226, June 2022.

[10] 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 *Computer Vision and Pattern Recognition (CVPR), 2019 IEEE International Conference on*, 2019.

[11] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7233–7243, 2022.

[12] Yair Kittenplon, Yonina C. Eldar, and Dan Raviv. Flowstep3d: Model unrolling for self-supervised scene flow estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4114–4123, June 2021.

[13] Bing Li, Cheng Zheng, Silvio Giancola, and Bernard Ghanem. Sctn: Sparse convolution-transformer network for scene flow estimation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 1254–1262, 2022.

[14] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 364–373, June 2021.

[15] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16959–16968, June 2022.

[16] Xingyu Liu, Charles R. Qi, and Leonidas J. Guibas. Flownet3d: Learning scene flow in 3d point clouds. In *Pro-*

ceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2019.

[17] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[18] Moritz Menze, Christian Heipke, and Andreas Geiger. Object scene flow. *ISPRS Journal of Photogrammetry and Remote Sensing*, 140:60–76, 2018.

[19] Himangi Mittal, Brian Okorn, and David Held. Just go with the flow: Self-supervised scene flow estimation. In *CVPR*, pages 11177–11185, 2020.

[20] Jhony Kaesemodel Pontes, James Hays, and Simon Lucey. Scene flow from point clouds with or without learning. In *2020 International Conference on 3D Vision (3DV)*, pages 261–270, 2020.

[21] Gilles Puy, Alexandre Boulch, and Renaud Marlet. Flot: Scene flow on point clouds guided by optimal transport. In *European conference on computer vision*, pages 527–544. Springer, 2020.

[22] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017.

[23] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *European conference on computer vision*, pages 402–419. Springer, 2020.

[24] Arash K Ushani and Ryan M Eustice. Feature learning for scene flow estimation from lidar. In *Conference on Robot Learning*, pages 283–292. PMLR, 2018.

[25] Arash K Ushani, Ryan W Wolcott, Jeffrey M Walls, and Ryan M Eustice. A learning approach for real-time temporal scene flow estimation from lidar data. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5666–5673. IEEE, 2017.

[26] Sundar Vedula, Simon Baker, Peter Rander, Robert Collins, and Takeo Kanade. Three-dimensional scene flow. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 722–729. IEEE, 1999.

[27] Guangming Wang, Yunzhe Hu, Zhe Liu, Yiyang Zhou, Masayoshi Tomizuka, Wei Zhan, and Hesheng Wang. What matters for 3d scene flow network. In *European Conference on Computer Vision*, pages 38–55. Springer, 2022.

[28] Guangming Wang, Yunzhe Hu, Xinrui Wu, and Hesheng Wang. Residual 3-d scene flow learning with context-aware feature extraction. *IEEE Transactions on Instrumentation and Measurement*, 71:1–9, 2022.

[29] Guangming Wang, Xinrui Wu, Zhe Liu, and Hesheng Wang. Hierarchical attention learning of scene flow in 3d point clouds. *IEEE Transactions on Image Processing*, 30:5168–5181, 2021.

[30] Sukai Wang, Yuxiang Sun, Chengju Liu, and Ming Liu. Pointtracknet: An end-to-end network for 3-d object detection and tracking from point clouds. *IEEE Robotics and Automation Letters*, 5(2):3206–3212, 2020.

[31] Shenlong Wang, Simon Suo, Wei-Chiu Ma, Andrei Pokrovsky, and Raquel Urtasun. Deep parametric continuous convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2589–2597, 2018.

[32] Yun Wang, Cheng Chi, and Xin Yang. Exploiting implicit rigidity constraints via weight-sharing aggregation for scene flow estimation from point clouds. *arXiv preprint arXiv:2303.02454*, 2023.

[33] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6954–6963, June 2021.

[34] Wenxuan Wu, Zhongang Qi, and Li Fuxin. Pointconv: Deep convolutional networks on 3d point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[35] 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 *European Conference on Computer Vision*, pages 88–107. Springer, 2020.

[36] Guangyao Zhai, Xin Kong, Jinhao Cui, Yong Liu, and Zhen Yang. Flowmot: 3d multi-object tracking by scene flow association. *arXiv preprint arXiv:2012.07541*, 2020.