SCTN: Sparse Convolution-Transformer Network for Scene Flow Estimation

Bing Li, Cheng Zheng, Silvio Giancola, Bernard Ghanem

King Abdullah University of Science and Technology
{bing.li, cheng.zheng, silvio.giancola, Bernard.Ghanem}@kaust.edu.sa

Abstract

We propose a novel scene flow estimation approach to capture and infer 3D motions from point clouds. Estimating 3D motions for point clouds is challenging, since a point cloud is unordered and its density is significantly non-uniform. Such unstructured data poses difficulties in matching corresponding points between point clouds, leading to inaccurate flow estimation. We propose a novel architecture named Sparse Convolution-Transformer Network (SCTN) that equips the sparse convolution with the transformer. Specifically, by leveraging the sparse convolution, SCTN transfers irregular point cloud into locally consistent flow features for estimating continuous and consistent motions within an object/local object part. We further propose to explicitly learn point relations using a point transformer module, different from exiting methods. We show that the learned relation-based contextual information is rich and helpful for matching corresponding points, benefiting scene flow estimation. In addition, a novel loss function is proposed to adaptively encourage flow consistency according to feature similarity. Extensive experiments demonstrate that our proposed approach achieves a new state of the art in scene flow estimation. Our approach achieves an error of 0.038 and 0.037 (EPE3D) on FlyingThings3D and KITTI Scene Flow respectively, which significantly outperforms previous methods by large margins.

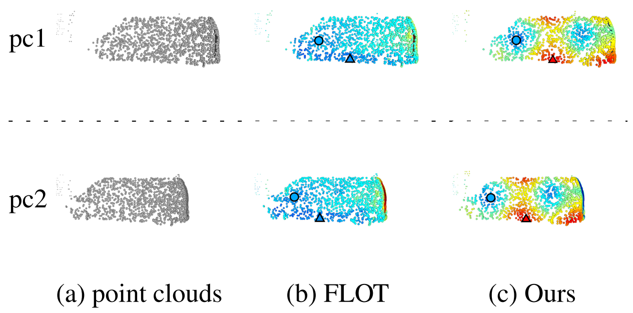

Figure 1: Illustrating the advantage of our SCTN in feature extraction, where first and second rows indicate the two point clouds, (b) and (c) visualize their features extracted by FLOT and our SCTN, respectively. Circles ◯ or triangles △ in (b)(c) indicate a pair of corresponding points between the point clouds, respectively. ◯ and △ are not corresponding to each other, however, their features extracted by FLOT are improperly similar, which are less discriminative and would lead to inaccurate predicted flows. In contrast, our SCTN extracts locally consistent while discriminative features.

Introduction

Understanding 3D dynamic scenes is critical to many real-world applications such as autonomous driving and robotics. Scene flow is the 3D motion of points in a dynamic scene, which provides low-level information for scene understanding (Vedula et al. 1999; Liu, Qi, and Guibas 2019; Geiger et al. 2013). The estimation of the scene flow can be a building block for more complex applications and tasks such as 3D object detection (Shi et al. 2020), segmentation (Thomas et al. 2019) and tracking (Qi et al. 2020). However, many previous scene flow methods estimate the 3D motion from stereo or RGB-D images. With the increasing popularity of point cloud data, it is desirable to estimate 3D motions directly from 3D point clouds.

Recent methods e.g.,(Wu et al. 2020; Liu, Qi, and Guibas 2019; Wei et al. 2021; Puy, Boulch, and Marlet 2020; Wang et al. 2021; Li et al. 2021) propose deep neural networks to learn scene flow from point clouds in an end-to-end way, which achieves promising estimation performance. However, estimating scene flow from point clouds is still challenging. In particular, existing methods (Liu, Qi, and Guibas 2019; Puy, Boulch, and Marlet 2020) extract feature for each point by aggregating information from its local neighborhood. However, such extracted features are not discriminative enough to matching corresponding points between point clouds (see Figure 1), leading to inaccurate flow estimation, since the feature extraction of these methods ignore two facts. First, the density of points is significantly non-uniform within a point cloud. It is non-trivial to learn point features that are simultaneously favorable for both points from dense regions and those from sparse regions. Second, due to Lidar and object motions, the density of points within an object often varies at the temporal dimension, leading that the geometry patterns of corresponding local regions are inconsistent between consecutive point clouds. As a result, extracted features, that are aggregated from only local regions without modeling point relations, is insufficient for scene flow estimation.

Another challenge lies in that most previous methods have to train a model on the synthetic dataset to estimate scene flow for real-world data. However, there exists domain shift between the synthetic dataset and the real-world one. For example, most objects in the synthetic dataset are rigid and undergo rigid motions, while real-world data contains many non-rigid objects whose motions are consistent within local object parts, rather than the whole object. Consequently, the performance of the trained model is degraded when handling real-world data. Yet, most recent work (Liu, Qi, and Guibas 2019; Puy, Boulch, and Marlet 2020; Wu et al. 2020) do not explicitly constrain the estimated flows. We would like to enforce the predicted flows in a local region of an object to be *consistent* and *smooth* in 3D space.

In this work, we resort to feature representations and loss functions to estimate accurate scene flow from point clouds. In particular, we explore novel feature representations (information) that would help to infer an accurate and locally consistent scene flows. We therefore propose a Sparse Convolution-Transformer Network (SCTN) which incorporates the merits of dual feature representations provided by our two proposed modules. In particular, to address the issue of spatially non-uniform and temporally varying density of dynamic point clouds, we propose a Voxelization-Interpolation based Feature Extraction (VIFE) module. VIFE extracts features from voxelized point clouds rather than original ones, and then interpolates features for each point, which encourages to generate locally consistent flow features. To furture improve discriminability of extracted features from the VIFE module, we propose to additionally model point relations in the feature space, such that extracted features capture important contextual information. Inspired by impressive performance of transformer in object detection tasks (Carion et al. 2020), we propose a Point Transformer-based Feature Extraction (PTFE) module to explicitly learn point relations based on transformer for capturing complementary information.

In addition, we propose a spatial consistency loss function with a new architecture that equips stop-gradient for training. The loss adaptively controls the flow consistency according to the similarity of point features. Our experiments demonstrate that our method significantly outperforms state-of-the art approaches on standard scene flow datasets: FlyingThings3D (N.Mayer et al. 2016a) and KITTI Scene Flow (Menze, Heipke, and Geiger 2018).

Contributions. Our contributions are fourfold:

1. Instead of designing convolution kernels, our VIFE module leverages simple operators – voxelization and interpolation for feature extraction, showing such smoothing operator is effective to extract local consistent while discriminative features for scene flow estimation.
2. Our PTFE module shows that explicitly modeling point relations can provide rich contextual information and is helpful for matching corresponding points, benefiting scene flow estimation. We are the first to introduce transformer for scene flow estimation.
3. We propose a new consistency loss equipping stop-gradient-based architecture that helps the model trained on synthetic dataset well adapt to real data, by controlling spatial consistency of estimated flows.
4. We propose a novel network that outperforms the state-of-the-art methods with remarkable margins on both FlyingThings3D and KITTI Scene Flow benchmarks.

Related Work

Optical Flow. Optical flow estimation is defined as the task of predicting the pixels motions between consecutive 2D video frames. Optical flow is a fundamental tool for 2D scene understanding, that have been extensively studied in the literature. Traditional methods (Horn and Schunck 1981; Black and Anandan 1993; Zach, Pock, and Bischof 2007; Weinzaepfel et al. 2013; Brox, Bregler, and Malik 2009; Ranftl, Bredies, and Pock 2014) address the problem of estimating optical flow as an energy minimization problem, that does not require any training data. Dosovitskiy et al. (Dosovitskiy et al. 2015) proposed a first attempt for an end-to-end model to solve optical flow based on convolution neural network (CNN). Inspired by this work, many CNN-based studies have explored data-driven approaches for optical flow (Dosovitskiy et al. 2015; Mayer et al. 2016; Ilg et al. 2017; Hui, Tang, and Loy 2018; Hui and Loy 2020; Sun et al. 2018; Teed and Deng 2020b).

Scene Flow from Stereo and RGB-D Videos. Estimating scene flow from stereo videos have been studied for years (Chen et al. 2020; Vogel, Schindler, and Roth 2013; Wedel et al. 2008; Ilg et al. 2018; Jiang et al. 2019; Teed and Deng 2020a). Many works estimate scene flow by jointly estimating stereo matching and optical flow from consecutive stereo frames (N.Mayer et al. 2016b). Similar to optical flow, traditional methods formulate scene flow estimation as an energy minimization problem (Huguet and Devernay 2007; Wedel et al. 2008). Recent works estimate scene flow from stereo video using neural networks (Chen et al. 2020). For example, networks for disparity estimation and optical flow are combined in (Ilg et al. 2018; Ma et al. 2019). Similarly, other works (Quiroga et al. 2014; Sun, Sudderth, and Pfister 2015) explore scene flow estimation from RGB-D video.

Scene Flow on Point Clouds. Inspired by FlowNet (Dosovitskiy et al. 2015), FlowNet3D (Liu, Qi, and Guibas 2019) propose an end-to-end network to estimate 3D scene flow from raw point clouds. Different from traditional methods (Dewan et al. 2016; Ushani et al. 2017), FlowNet3D (Liu, Qi, and Guibas 2019) is based on PointNet++ (Qi et al. 2017b), and propose a flow embedding layer to aggregate the information from consecutive point clouds and extract scene flow with convolutional layers. FlowNet3D++ (Wang et al. 2020) improves the accuracy of FlowNet3D by incorporating geometric constraints. HPLFlowNet (Gu et al. 2019) projects point clouds into permutohedral lattices, and then estimates scene flow using Bilateral Convolutional Layers. Inspired by the successful optical flow method PWC-Net (Sun et al. 2018), PointPWC (Wu et al. 2020) estimates scene flow in a coarse-to-fine fashion, introducing cost volume, upsampling, and warping

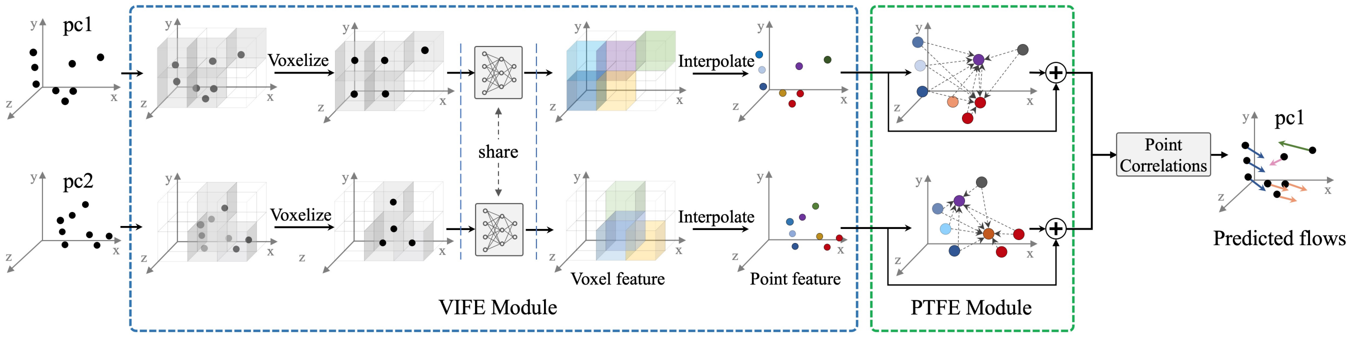

Figure 2: Overall framework of our SCTN approach. Given two consecutive point clouds, the Voxelization-Interpolation Feature Extraction (VIFE) extracts features from voxelized point clouds and then projects back the voxel features into point features. These point features are fed into the Point Transformer Feature Extraction (PTFE) module to explicitly learn point relations. With fused features of VIFE and PTFE module, SCTN computes point correlations between the point clouds and predicts flows.

modules for the point cloud processing. The most related recent work to our approach is FLOT (Puy, Boulch, and Marlet 2020). FLOT addresses the scene flow as a matching problem between corresponding points in the consecutive clouds and solve it using optimal transport. Our method differs from FLOT (Puy, Boulch, and Marlet 2020) in two aspects. First, our method explicitly explores more suitable feature representation that facilitate the scene flow estimation. Second, our method is trained to enforce the consistency of predicted flows for local-region points from the same object, which is ignored in FLOT (Puy, Boulch, and Marlet 2020). Recently, Gojcic et al. (Gojcic et al. 2021) explore weakly supervised learning for scene flow estimation using labels of ego motions as well as ground-truth foreground and background masks. Other works (Wu et al. 2020; Kittenplon, Eldar, and Raviv 2021; Mittal, Okorn, and Held 2020) study unsupervised/self-supervised learning for scene flow estimation on point clouds, proposing regularization losses that enforces local spatial smoothness of predicted flows. These losses are directly constraining points in a local region to have similar flows, but are not feature-aware.

3D Deep Learning. Many works have introduced deep representation for point cloud classification and segmentation (Qi et al. 2017b; Thomas et al. 2019; Zhang, Hua, and Yeung 2019; Liu et al. 2020; Wu, Qi, and Fuxin 2019; Li et al. 2018; Lei, Akhtar, and Mian 2020; Hu et al. 2022; Zhao et al. 2021). Qi et al. (Qi et al. 2017a) propose PointNet that learns point feature only from point positions. PointNet++ (Qi et al. 2017b) extends PointNet by aggregating information from local regions. Motivated by PointNet++, many works (Zhang, Hua, and Yeung 2019; Thomas et al. 2019; Lei, Akhtar, and Mian 2020; Li et al. 2018) design various local aggregation functions for point cloud classification and segmentation. Different from these point-based convolutions, (Wu et al. 2015; Wang et al. 2017) transform point cloud into voxels, such that typical 3D convolution can be applied. However, such voxel-based convolution suffers from expensive computational and memory cost as well as information loss during voxelization. Liu et al. (Liu et al. 2019) combine PointNet (Qi et al. 2017a) with voxel-based convolution to reduce the memory consumption. For the sake of efficient learning, researchers have explored sparse convolution for point cloud segmentation (Xie et al. 2020; Choy, Gwak, and Savarese 2019) which shows impressive performance. Tang et al. (Tang et al. 2020) propose to combine PointNet with sparse convolution for large-scale point cloud segmentation. Differently, we not only leverage voxelization and interpolation for feature extraction, but also explicitly model point relations to provide complementary information.

Methodology

Problem Definition. Given two consecutive point clouds $\mathcal{P}^t$ and $\mathcal{P}^{t+1}$, scene flow estimation is to predict the 3D motion flow of each point from $\mathcal{P}^t$ to $\mathcal{P}^{t+1}$. Let p_i^t be the 3D coordinates of i-th point in $\mathcal{P}^t = \{p_i^t\}_{i=1}^{n_\mathcal{P}}$. Like previous work (Puy, Boulch, and Marlet 2020), we predict scene flow based on the correlations of each point pair between $\mathcal{P}^t$ and $\mathcal{P}^{t+1}$. Given a pair of points $p_i^t \in \mathcal{P}^t$ and $p_j^{t+1} \in \mathcal{P}^{t+1}$, the correlation of the two points is computed as follows:

$$C(p_i^t, p_j^{t+1}) = \frac{(\mathbf{F}_i^t)^T \cdot \mathbf{F}_j^{t+1}}{\|\mathbf{F}_i^t\|_2 \|\mathbf{F}_j^{t+1}\|_2} \tag{1}$$

where $\mathbf{F}_j^{t+1}$ is the feature of p_j^{t+1}, and $\|\cdot\|_2$ is the L2 norm.

Point feature $\mathbf{F}$ is the key to computing the correlation $C(p_i^t, p_j^{t+1})$ which further plays an important role in scene flow estimation. Hence, it is desirable to extract effective point features that enable point pairs in corresponding regions to achieve higher correlation values between $\mathcal{P}^t$ and $\mathcal{P}^{t+1}$. However, different from point cloud segmentation and classification that focus on static point cloud, scene flow estimation operates on dynamic ones, which poses new challenges for feature extraction in two aspects. For example, the input point clouds of scene flow are not only irregular and unordered, but also its density is spatially non-uniform and temporally varying, as discussed in previous sections.

Our goal is to extract locally consistent while discriminative features for points, so as to achieve accurate flow estimation. Different from exiting methods directly extracting point feature from the neighborhood in original point cloud, we proposed two feature extraction modules for scene flow estimation. The Voxelization-Interpolation based Feature Extraction (VIFE) is proposed to address the issue of point cloud's non-uniform density. With the features extracted by VIFE, Point Transformer based Feature Extraction (PTFE) further enhance feature discriminability by modeling point relations globally. Since the two kinds of features provide complementary information, we fuse these features, such that the fused features provide proper correspondences to predict accurate scene flows.

Overview. Figure 2 illustrates the overall framework of our approach, that takes two consecutive point clouds $\mathcal{P}^t$ and $\mathcal{P}^{t+1}$ as inputs and predict the scene flow from $\mathcal{P}^t$ to $\mathcal{P}^{t+1}$. First, the VIFE module extracts features for voxel points from voxelized point clouds, and projects back the voxel features into the original 3D points to obtain locally consistent point features. Second, our PTFE module improves the point feature representation by modeling relation between points. Third, with features extracted by VIEF and PTFE module, we calculate the correlations of points between $\mathcal{P}^t$ and $\mathcal{P}^{t+1}$, where a sinkhorn algorithm (Puy, Boulch, and Marlet 2020; Cuturi 2013; Chizat et al. 2018) is leveraged to predict the flows. We train our method with an extra regularizing loss to enforce spatial consistency of predicted flows.

Voxelization-Interpolation Based Feature Extraction

As mentioned in previous sections, the density of consecutive point clouds is spatially non-uniform and temporally varying, posing difficulties in feature extraction. To address the issue, we propose the Voxelization-Interpolation based Feature Extraction (VIFE) module. VIFE first voxelizes the consecutive point clouds into voxels. As illustrated in Figure 2, the spatial non-uniform distributions and temporally variations of points are reduced to some extent.

After that, VIFE conducts convolutions on voxel points rather than all points of the point cloud, and then interpolate features for each point. We argue that such simple operators i.e.,voxelization and interpolation, ensure points in a local neighborhood to have smoother features, ideally leading to consistent flows in space.

Voxel feature extraction. We then leverage a U-Net (Ronneberger, Fischer, and Brox 2015) architecture network to extract feature from voxelized point clouds, where convolution can be many types of point cloud convolutions such as pointnet++ used in FLOT (Puy, Boulch, and Marlet 2020). Here, we adopt sparse convolution e.g.,Minkowski Engine (Choy, Gwak, and Savarese 2019) for efficiency. More details are available in our supplementary material.

Point feature interpolation. We project back the voxel features into point feature $\mathbf{F}_i^S$ for point p_i. In particular, we interpolate the point features from the K closest voxels following equation (2). $\mathcal{N}^v(p_i)$ represents the set of K nearest neighboring non-empty voxels for the point p_i, $\mathbf{v}_k \in \mathbb{R}^C$ represents the feature of k-th closest non-empty voxel and d_{ik} the Euclidian distance between the point p_i and the center of the k-th closest non-empty voxel.

$$\mathbf{F}_i^S = \frac{\sum_{k\in\mathcal{N}^v(p_i)} d_{ik}^{-1} \cdot \mathbf{v}_k}{\sum_{k\in\mathcal{N}^v(p_i)} d_{ik}^{-1}} \tag{2}$$

We observed that close points are encouraged to have similar features, which helps our method generate consistent flows for these points. This is favorable for local object parts or rigid objects with dense densities and consistent flows (e.g.,LiDAR points on a car at close range).

Point Transformer Based Feature Extraction

Our VIFE module adopt aggressive downsampling to obtain a large receptive field and low computation cost. However, aggressive downsampling inevitably loses some important information (Tang et al. 2020). In such case, the features of points with large information loss are disadvantageous for estimating their scene flow. To address this issue, we explicitly exploit point relations as a complementary information on top of the point feature extracted with VIFE. Recent work (Zhu et al. 2021; Zhao, Jia, and Koltun 2020; Carion et al. 2020) employ transformer and self-attention to model internal relation in the features, achieving impressive performance in image tasks such as detection and recognition. Similar trend appeared in point cloud classification and segmentation (Guo et al. 2021; Zhao et al. 2021; Engel, Belagiannis, and Dietmayer 2020) showing the effectiveness of transformer in 3D. Inspired by these work, we resort to transformer for capturing point relation information as the point feature.

In an autonomous navigation scenario, point clouds represent complete scenes, with small object such as cars and trucks, but also large structure such as buildings and walls. The scene flows in such a large scene do not only depend on the aggregation from a small region, but rather a large one. As a results, we refrain in building a transformer for the local neighborhood as it would restrict the receptive field or require deeper model (i.e.,increase the memory). Instead, our transformer module learns the relation of each point to all other points, such that the transformer can adaptively capture the rich contextual information from a complete scene.

Formally, given a point p_i, we consider every points in $\mathcal{P}$ as query and key elements. Our transformer module builds a point feature representation for p_i by adaptively aggregating the features of all points based on self-attention:

$$\mathbf{F}_i^R = \sum_{j=1}^{n_\mathcal{P}} A_{i,j} \cdot g_v(\mathbf{F}_j^S, \mathbf{G}_j) \tag{3}$$

where g_v is the a learnable function (e.g.,linear function), $A_{i,j}$ is an attention defining a weight of p_j to p_i, $\mathbf{G}_j$ is the positional encoding feature of p_j.

As pointed in literature (Zhao, Jia, and Koltun 2020; Carion et al. 2020), the positional encoding feature can provide important information for the transformer. The position encoding in recent transformer work (Zhao et al. 2021) encodes the *relative* point position to neighbors for point cloud classification or segmentation. Different from those tasks, the task of scene flow is to find correspondences between consecutive point clouds. Thus, we argue that an *absolute* position provides sufficient information to estimate

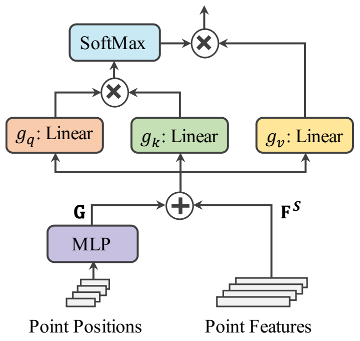

Figure 3: Details of our point transformer based feature extraction module (PTFE). $\otimes$ and $\oplus$ correspond to matrix multiplication and addition operations, respectively.

the scene flow. Therefore, given a point p, our position encoding function encodes its *absolute* position p_j:

$$\mathbf{G}_j = \phi(p_j) \tag{4}$$

where ϕ is a MLP layer. Using absolute positions reduce computational cost, compared with using relative positions.

We calculate an attention $A_{i,j}$ as the similarity between the features of p_i and p_j in an embedding space. The similarity is estimated using features and position information:

$$A_{i,j} \propto \exp\left(\frac{(g_q(\mathbf{F}_i^S, \mathbf{G}_i))^T \cdot g_k(\mathbf{F}_j^S, \mathbf{G}_j)}{c_a}\right) \tag{5}$$

where $g_q(\cdot,\cdot)$ and $g_k(\cdot,\cdot)$ are the learnable mapping functions to project feature into an embedding space, and c_a is the output dimension of $g_q(\cdot,\cdot)$ or $g_k(\cdot,\cdot)$. $A_{i,j}$ is further normalized such that $\sum_j A_{i,j} = 1$. The architecture of our transformer module is illustrated in Figure 3.

Flow Prediction

Since the point feature from our VIFE and PTFE modules provide complementary information, we fuse the two kinds of features through skip connection for each point, i.e.,$\mathbf{F}_i = \mathbf{F}_i^S + \mathbf{F}_i^R$. By feeding the fused point features into Eq. 1, we compute the correlations of all pairs $\mathbf{C}(\mathcal{P}^t, \mathcal{P}^{t+1}) = \{C(p_i^t, p_j^{t+1})\}$ between the two consecutive point clouds.

With the estimated point correlations, we adopt the Sinkhorn algorithm to estimate soft correspondences and predict flows for $\mathcal{P}^t$, following FLOT.

Training Losses

We train our model to regress the scene flow in a supervised fashion, on top of which we propose a Feature-aware Spatial Consistency loss, named "FSC loss", that enforces similar features to have similar flow. The FSC loss provides a better generalization and transfer capability between the training and testing datasets.

Supervised loss. We define in Equation (6) our supervised loss E^s that minimize the L_1-norm difference between the estimated flow and the ground truth flow for the non-occluded points. $\mathbf{u}_i^*$ and $\mathbf{u}_i$ are respectively the ground-truth and predicted motion flow for the point $p_i \in \mathcal{P}$ and m_i is a binary indicator for the non-occlusion of this point, i.e.,$m_i = 0$ if p_i is occluded, otherwise $m_i = 1$.

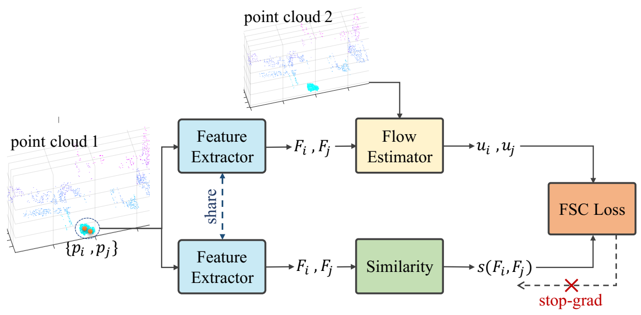

Figure 4: Stop gradient for the FSC loss. We extract the flow from a neighboring point as well as the similarity between their features. We optimize the FSC loss without back-propagating the gradients to the features from the similarity branch to avoid degenerate cases.

$$E^s = \sum_{i}^{N} m_i \|\mathbf{u}_i - \mathbf{u}_i^*\| \tag{6}$$

Feature-aware Spatial Consistency (FSC) loss. Given a local region, points from the same object usually has consistent motions, resulting in similar flows. To model such phenomena, we propose a consistency loss that ensures points within an object/local object part to have similar predicted flows. Yet, object annotations are not necessarily available. Instead, we propose to control flow consistency according to feature similarity. That is, given a local region, if two points are of larger feature similarity, they are of the higher probability that belongs to the same object. In particular, given a point p_i with predicted flow $\mathbf{u}_i$ and its local neighborhood $\mathcal{N}(p_i)$, we enforce the flow $\mathbf{u}_j$ of $p_j \in \mathcal{N}(p_i)$ to be similar to $\mathbf{u}_i$, if the feature $\mathbf{F}_i$ of p_j is similar to $\mathbf{F}_j$ of p_j. Formally, we define the FSC loss as follows:

$$E^c = \sum_{i=1}^{N} \frac{1}{K} \sum_{p_j \in \mathcal{N}(p_i)} s(\mathbf{F}_i, \mathbf{F}_j) \cdot \|\mathbf{u}_i - \mathbf{u}_j\|_2 \tag{7}$$

where the similarity function $s(\mathbf{F}_i, \mathbf{F}_j)$ of p_i and p_j is defined as $1 - \exp(-(\mathbf{F}_i)^T \cdot \mathbf{F}_j / \tau)$ with τ being a temperature hyper-parameter, K is the number of points in $\mathcal{N}(p_i)$.

A naive implementation of the FSC loss would inevitably lead to degenerate cases. In particular, the FSC loss is a product of two objectives: (i) a similarity $s(\mathbf{F}_i, \mathbf{F}_j)$ between the features $\mathbf{F}_i$ and $\mathbf{F}_j$ and (ii) a difference $\|\mathbf{u}_i - \mathbf{u}_j\|_1$ between their flows. The scope of this loss is to train the flows to be similar if they have similar features. However, to minimize the FSC loss, the model would make the features $\mathbf{F}_j$ and $\mathbf{F}_i$ be orthogonal (i.e.,$\mathbf{F}_j \cdot \mathbf{F}_i = 0$), such that $s(\mathbf{F}_j, \mathbf{F}_i) = 0$ (i.e.,$E^c = 0$). Obviously, it is against our aim.

To circumvent this limitation, we propose a stop-gradient for the FSC loss, taking inspiration form recent advances in

Dataset	Method	EPE3D(m) ↓	Acc3DS ↑	Acc3DR ↑	Outliers ↓
FlyingThings3D	FlowNet3D (Liu, Qi, and Guibas 2019)	0.114	0.412	0.771	0.602
	HPLFlowNet (Gu et al. 2019)	0.080	0.614	0.855	0.429
	PointPWC (Wu et al. 2020)	0.059	0.738	0.928	0.342
	EgoFlow (Tishchenko et al. 2020)	0.069	0.670	0.879	0.404
	FLOT (Puy, Boulch, and Marlet 2020)	0.052	0.732	0.927	0.357
	SCTN (ours)	**0.038**	**0.847**	**0.968**	**0.268**
KITTI	FlowNet3D (Liu, Qi, and Guibas 2019)	0.177	0.374	0.668	0.527
	HPLFlowNet (Gu et al. 2019)	0.117	0.478	0.778	0.410
	PointPWC (Wu et al. 2020)	0.069	0.728	0.888	0.265
	EgoFlow (Tishchenko et al. 2020)	0.103	0.488	0.822	0.394
	FLOT (Puy, Boulch, and Marlet 2020)	0.056	0.755	0.908	0.242
	SCTN (ours)	**0.037**	**0.873**	**0.959**	**0.179**

Table 1: Comparison with the state-of-the-art on FlyingThings3D and KITTI. Best results in bold. Our proposed model SCTN reaches highest performances in all metrics.

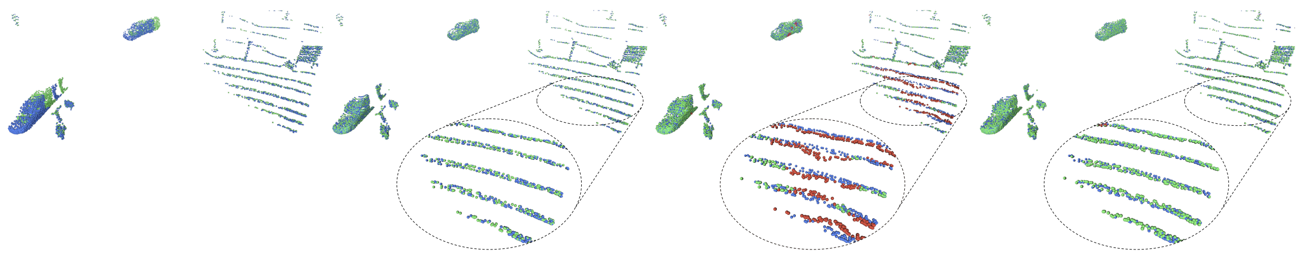
(a) Input point clouds (b) Ground-truth flows (c) FLOT (d) Ours

Figure 5: Qualitative comparison results. Green points indicate the first point cloud in (a), and blue points indicate the second point cloud in (a)(b)(c)(d). In (b)(c)(d), green points are the ones in the first point cloud warped by correctly predicted flows, while red points are the ones warped by incorrect flows (the first point cloud + incorrect scene flow whose EPE3D >0.1m).

self-supervised learning (Chen and He 2020). As illustrated in Figure 4, our architecture stops the propagation of the gradient in the branch extracting the feature similarity. By such architecture, our FSC loss avoids optimizing the features, while optimizing solely the flows similarities $\|\mathbf{u}_j - \mathbf{u}_i\|_1$ for neighboring points with similar features.

Experiments

Dataset. We conduct our experiments on two datasets that are widely used to evaluate scene flow. *FlyingThings3D* (N.Mayer et al. 2016a) is a large-scale synthetic stereo video datasets, where synthetic objects are selected from ShapeNet (Chang et al. 2015) and randomly assigned various motions. We generate 3D point clouds and ground truth scene flows with their associated camera parameters and disparities. Following the same preprocessing as in (Puy, Boulch, and Marlet 2020; Gu et al. 2019; Wu et al. 2020), we randomly sample 8192 points and remove points with camera depth greater than 35 m. We use the same $19640/3824$ pairs of point cloud (training/testing) used in the related works (Puy, Boulch, and Marlet 2020; Gu et al. 2019; Wu et al. 2020). *KITTI Scene Flow* (Menze, Heipke, and Geiger 2018; Choy, Gwak, and Savarese 2019) is a real-world Lidar scan dataset for scene flow estimation from the KITTI autonomous navigation suite. Following the preprocessing of (Gu et al. 2019), we leverage 142 point cloud pairs of 8192 points for testing. For a fair comparison, we also remove ground points by discarding points whose height is lower than -1.4m, following the setting of existing methods (Puy, Boulch, and Marlet 2020; Wu et al. 2020; Gu et al. 2019).

Evaluation Metrics. To evaluate the performance of our approach, we adopt the standard evaluation metrics used in the related methods (Puy, Boulch, and Marlet 2020; Wu et al. 2020), described as follows: The *EPE3D (m)* (3D end-point-error) is calculated by computing the average L_2 distance between the predicted and GT scene flow, in meters. This is our main metric. The *Acc3DS* is a strict version of the accuracy which estimated as the ratio of points whose EPE3D $<$ 0.05 m or relative error $<$5%. The *Acc3DR* is a relaxed accuracy which is calculated as the ratio of points whose EPE3D $<$0.10m or relative error $<$10%. The *Outliers* is the ratio of points whose EPE3D $>$0.30m or relative error $>$10%.

Implementation Details. We implement our method in PyTorch (Paszke et al. 2019). We train our method on FlyingThing3D then evaluate on FlyingThing3D and KITTI. We minimize a cumulative loss $E = E^s + \lambda E^c$ with $\lambda = 0.30$ a weight that scale the losses. We use the Adam optimizer (Kingma and Ba 2014) with an initial learning rate of 10^{-3}, which is dropped to 10^{-4} after the 50^{th} epoch. First, we train for 40 epochs only using the supervised loss. Then we continue the training for 20 epochs with both the supervision loss and the FSC loss, for a total on 60 epochs. We

VIFE	PTFE	FSC loss	EPE3D(m) ↓	Acc3DS ↑
✓			0.045	0.835
✓		✓	0.042	0.853
✓	✓		0.040	0.863
✓	✓	✓	**0.037**	**0.873**

Table 2: Ablation for SCTN. We further analyse the performances of our three components: VIFE, PTFE and FSC loss on KITTI. We highlight in bold the best performances.

use a voxel size of resolution 0.07m.

Runtime. We evaluate the running time of our method. Table 3 reports the evaluated time compared with recent state-of-the-art methods. FLOT (Puy, Boulch, and Marlet 2020) is the most related work to our method, since we both adopt point-wise correlations to generate predicted flows. Our method consumes lower running time than FLOT, although the transformer module is equipped.

Quantitative Evaluation

We compare our approach with recent deep-learning-based methods including FlowNet3D (Liu, Qi, and Guibas 2019), HPLFlowNet (Gu et al. 2019), PointPWC (Wu et al. 2020) and FLOT (Puy, Boulch, and Marlet 2020). These methods are state-of-the-art in scene flow estimation from point cloud data and do not leverge any additional labels such as ground-truth ego motions or instance segmentation.

Results on FlyingThings3D. We train and evaluate our model on the FlyThings3D datasets. As shown in Table 1, our method outperforms all methods in every metrics by a significant margin. It is worth noting that our method obtains an EPE3D metric below 4cm, with a relative improvement of 26.9% and 35.5% over the most recent methods FLOT (Puy, Boulch, and Marlet 2020) and PointPWC (Wu et al. 2020), respectively. The performance shows that our method is effective in predicting flows with high accuracy.

Results on KITTI without Fine-tune. Following the common practice (Puy, Boulch, and Marlet 2020; Wu et al. 2020), we train our model on FlyingThings3D and directly test the trained model on KITTI Scene Flow dataset, without any fine-tuning, to evaluate the generalization capability of our method. We report in Table 1 the highest accuracy of scene flow estimation on KITTI Scene Flow dataset for our SCTN method. Again, we reduce the EPE3D metric below 4cm, with a 33.9% relative improvement over FLOT (Puy, Boulch, and Marlet 2020). In the Acc3DS metrics, our method outperforms both FLOT (Puy, Boulch, and Marlet 2020) and PointPWC (Wu et al. 2020) by 13.5% and 16.6% respectively. These results highlight the capability of our method to generalize well on real-world datasets.

Qualitative Evaluation

To qualitatively evaluate the quality of our scene flow predictions, we visualize the predicted scene flow and ground-truth one in Figure 5. Since FLOT (Puy, Boulch, and Marlet 2020) is most related to our method, we compare the qualitative performances of our SCTN with FLOT (Puy, Boulch, and Marlet 2020).

Method	Runtime (ms)
FLOT (Puy, Boulch, and Marlet 2020)	389.3
SCTN (ours)	242.7

Table 3: Running time comparisons. The runtime of FLOT and our SCTN are evaluated on a single GTX2080Ti GPU. We used the official implementation of FLOT.

Points in a local region from the same object usually have similar ground-truth flows. Yet, FLOT introduces prediction errors in local regions, highlighting the inconsistency in the scene flow predictions. For example, FLOT inaccurately predicts scene flow for some regions in the background, even though those points have similar flows, as shown in Figure 5. In contrast, our method is more consistent in the prediction for points in the same object, achieving better performance, e.g.,for the background objects with complex structure.

Ablation Study

To study the roles of the proposed VIFE, PTFE and FSC loss, we ablate each proposed component of our model and evaluate their performance on KITTI. For all the experiments, we follow the same training procedure than in the main results. Table 2 reports the evaluation results.

VIFE module. Table 2 shows that our approach with the sole VIEF convolution module already outperforms the state-of-the-art methods listed in Table 1. Different from existing methods directly applying convolution on original point clouds, our VIFE extracts feature from voxelized point cloud, which reduces the non-uniform density of point cloud, while ensuring that points in a local region have consistent features, to some extent. The results show that such features are favorable for scene flow estimation.

PTFE module. Compared with only using VIFE module, adding PTFE improves both metrics on KITTI as reported in the third row in Table 2. For example, EPE3D is improved by 11.1%, compared with only using the VIEF module. Our PIFE module explicitly learns point relations, which provides rich contextual information and helps to match corresponding points even for objects with complex structures.

FSC loss. Table 2 shows that adding the FSC loss helps to achieve better scene flow estimation on KITTI. Our FSC loss improves the generalization capability of our method.

Conclusion

We present a Sparse Convolution-Transformer Network (SCTN) for scene flow estimation. Our SCTN leverages the VIFE module to transfer irregular point cloud into locally smooth flow features for estimating spatially consistent motions in local regions. Our PTFE module learns rich contextual information via explicitly modeling point relations, which is helpful for matching corresponding points and benefits scene flow estimation. A novel FSC loss is also proposed for training SCTN, improving the generalization ability of our method. Our approach achieves state-of-the-art performances on FlyingThings3D and KITTI datasets.

Acknowledgments

This work was supported by the King Abdullah University of Science and Technology (KAUST) Office of Sponsored Research through the Visual Computing Center (VCC) funding. We thank Hani Itani for his constructive suggestions and help.

References

Black, M. J.; and Anandan, P. 1993. A framework for the robust estimation of optical flow. In *ICCV*, 231–236. IEEE.

Brox, T.; Bregler, C.; and Malik, J. 2009. Large displacement optical flow. In *CVPR*, 41–48. IEEE.

Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; and Zagoruyko, S. 2020. End-to-end object detection with transformers. In *ECCV*, 213–229.

Chang, A. X.; Funkhouser, T.; Guibas, L.; Hanrahan, P.; Huang, Q.; Li, Z.; Savarese, S.; Savva, M.; Song, S.; Su, H.; Xiao, J.; Yi, L.; and Yu, F. 2015. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], arXiv preprint.

Chen, X.; and He, K. 2020. Exploring Simple Siamese Representation Learning. arXiv:2011.10566.

Chen, Y.; Van Gool, L.; Schmid, C.; and Sminchisescu, C. 2020. Consistency Guided Scene Flow Estimation. In *ECCV*, 125–141. Springer.

Chizat, L.; Peyré, G.; Schmitzer, B.; and Vialard, F.-X. 2018. Scaling algorithms for unbalanced optimal transport problems. *Mathematics of Computation*, 87(314): 2563–2609.

Choy, C.; Gwak, J.; and Savarese, v. 2019. 4D Spatio-Temporal ConvNets: Minkowski Convolutional Neural Networks. In *CVPR*, 3075–3084.

Cuturi, M. 2013. Sinkhorn distances: lightspeed computation of optimal transport. In *NeurIPS*, volume 2, 4.

Dewan, A.; Caselitz, T.; Tipaldi, G. D.; and Burgard, W. 2016. Rigid scene flow for 3d lidar scans. In *IROS*, 1765–1770. IEEE.

Dosovitskiy, A.; Fischer, P.; Ilg, E.; Hausser, P.; Hazirbas, C.; Golkov, V.; Van Der Smagt, P.; Cremers, D.; and Brox, T. 2015. Flownet: Learning optical flow with convolutional networks. In *ICCV*, 2758–2766.

Engel, N.; Belagiannis, V.; and Dietmayer, K. 2020. Point Transformer. arXiv:2011.00931.

Geiger, A.; Lenz, P.; Stiller, C.; and Urtasun, R. 2013. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11): 1231–1237.

Gojcic, Z.; Litany, O.; Wieser, A.; Guibas, L. J.; and Birdal, T. 2021. Weakly Supervised Learning of Rigid 3D Scene Flow. In *CVPR*, 5692–5703.

Gu, X.; Wang, Y.; Wu, C.; Lee, Y. J.; and Wang, P. 2019. HPLFlowNet: Hierarchical Permutohedral Lattice FlowNet for Scene Flow Estimation on Large-scale Point Clouds. In *CVPR*.

Guo, M.-H.; Cai, J.-X.; Liu, Z.-N.; Mu, T.-J.; Martin, R. R.; and Hu, S.-M. 2021. PCT: Point Cloud Transformer. arXiv:2012.09688.

Horn, B. K.; and Schunck, B. G. 1981. Determining optical flow. *Artificial intelligence*, 17(1-3): 185–203.

Hu, W.; Pang, J.; Liu, X.; Tian, D.; Chia-Wen, L.; and Anthony, V. 2022. Graph signal processing for geometric data and beyond: Theory and applications. *IEEE Trans. Multimedia*.

Huguet, F.; and Devernay, F. 2007. A variational method for scene flow estimation from stereo sequences. In *ICCV*, 1–7. IEEE.

Hui, T.-W.; and Loy, C. C. 2020. LiteFlowNet3: Resolving Correspondence Ambiguity for More Accurate Optical Flow Estimation. In *ECCV*, 169–184. Springer.

Hui, T.-W.; Tang, X.; and Loy, C. C. 2018. Liteflownet: A lightweight convolutional neural network for optical flow estimation. In *CVPR*, 8981–8989.

Ilg, E.; Mayer, N.; Saikia, T.; Keuper, M.; Dosovitskiy, A.; and Brox, T. 2017. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *CVPR*, 2462–2470.

Ilg, E.; Saikia, T.; Keuper, M.; and Brox, T. 2018. Occlusions, motion and depth boundaries with a generic network for disparity, optical flow or scene flow estimation. In *ECCV*, 614–630.

Jiang, H.; Sun, D.; Jampani, V.; Lv, Z.; Learned-Miller, E.; and Kautz, J. 2019. Sense: A shared encoder network for scene-flow estimation. In *ICCV*, 3195–3204.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kittenplon, Y.; Eldar, Y. C.; and Raviv, D. 2021. FlowStep3D: Model Unrolling for Self-Supervised Scene Flow Estimation. In *CVPR*, 4114–4123.

Lei, H.; Akhtar, N.; and Mian, A. 2020. Spherical kernel for efficient graph convolution on 3d point clouds. *TPAMI*.

Li, R.; Lin, G.; He, T.; Liu, F.; and Shen, C. 2021. 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)*, 364–373.

Li, Y.; Bu, R.; Sun, M.; Wu, W.; Di, X.; and Chen, B. 2018. PointCNN: Convolution on χ-transformed points. In *NeurIPS*, 828–838.

Liu, X.; Qi, C. R.; and Guibas, L. J. 2019. FlowNet3D: Learning Scene Flow in 3D Point Clouds. In *CVPR*.

Liu, Z.; Hu, H.; Cao, Y.; Zhang, Z.; and Tong, X. 2020. A closer look at local aggregation operators in point cloud analysis. In *ECCV*, 326–342. Springer.

Liu, Z.; Tang, H.; Lin, Y.; and Han, S. 2019. Point-Voxel CNN for Efficient 3D Deep Learning. In *NeurIPS*.

Ma, W.-C.; Wang, S.; Hu, R.; Xiong, Y.; and Urtasun, R. 2019. Deep Rigid Instance Scene Flow. In *CVPR*.

Mayer, N.; Ilg, E.; Hausser, P.; Fischer, P.; Cremers, D.; Dosovitskiy, A.; and Brox, T. 2016. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *CVPR*, 4040–4048.

Menze, M.; Heipke, C.; and Geiger, A. 2018. Object Scene Flow. *JPRS*.

Mittal, H.; Okorn, B.; and Held, D. 2020. Just Go With the Flow: Self-Supervised Scene Flow Estimation. In *CVPR*.

N.Mayer; E.Ilg; P.Häusser; P.Fischer; D.Cremers; A.Dosovitskiy; and T.Brox. 2016a. A Large Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation. In *CVPR*.

N.Mayer; E.Ilg; P.Häusser; P.Fischer; D.Cremers; A.Dosovitskiy; and T.Brox. 2016b. A Large Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation. In *CVPR*.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703*.

Puy, G.; Boulch, A.; and Marlet, R. 2020. FLOT: Scene Flow on Point Clouds Guided by Optimal Transport. In *ECCV*.

Qi, C. R.; Su, H.; Mo, K.; and Guibas, L. J. 2017a. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 652–660.

Qi, C. R.; Yi, L.; Su, H.; and Guibas, L. J. 2017b. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *NeurIPS*, 5099–5108.

Qi, H.; Feng, C.; Cao, Z.; Zhao, F.; and Xiao, Y. 2020. P2B: Point-to-box network for 3D object tracking in point clouds. In *CVPR*, 6329–6338.

Quiroga, J.; Brox, T.; Devernay, F.; and Crowley, J. 2014. Dense semi-rigid scene flow estimation from rgbd images. In *ECCV*, 567–582. Springer.

Ranftl, R.; Bredies, K.; and Pock, T. 2014. Non-local total generalized variation for optical flow estimation. In *ECCV*, 439–454. Springer.

Ronneberger, O.; Fischer, P.; and Brox, T. 2015. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 234–241. Springer.

Shi, S.; Guo, C.; Jiang, L.; Wang, Z.; Shi, J.; Wang, X.; and Li, H. 2020. Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. In *CVPR*, 10529–10538.

Sun, D.; Sudderth, E. B.; and Pfister, H. 2015. Layered RGBD scene flow estimation. In *CVPR*, 548–556.

Sun, D.; Yang, X.; Liu, M.-Y.; and Kautz, J. 2018. PWC-Net: CNNs for Optical Flow Using Pyramid, Warping, and Cost Volume. In *CVPR*.

Tang, H.; Liu, Z.; Zhao, S.; Lin, Y.; Lin, J.; Wang, H.; and Han, S. 2020. Searching Efficient 3D Architectures with Sparse Point-Voxel Convolution. In *ECCV*.

Teed, Z.; and Deng, J. 2020a. RAFT-3D: Scene Flow using Rigid-Motion Embeddings. *arXiv preprint arXiv:2012.00726*.

Teed, Z.; and Deng, J. 2020b. Raft: Recurrent all-pairs field transforms for optical flow. In *ECCV*, 402–419. Springer.

Thomas, H.; Qi, C. R.; Deschaud, J.-E.; Marcotegui, B.; Goulette, F.; and Guibas, L. J. 2019. Kpconv: Flexible and deformable convolution for point clouds. In *ICCV*, 6411–6420.

Tishchenko, I.; Lombardi, S.; Oswald, M. R.; and Pollefeys, M. 2020. Self-Supervised Learning of Non-Rigid Residual Flow and Ego-Motion. *arXiv preprint arXiv:2009.10467*.

Ushani, A. K.; Wolcott, R. W.; Walls, J. M.; and Eustice, R. M. 2017. A learning approach for real-time temporal scene flow estimation from lidar data. In *ICRA*, 5666–5673. IEEE.

Vedula, S.; Baker, S.; Rander, P.; Collins, R.; and Kanade, T. 1999. Three-dimensional scene flow. In *ICCV*, volume 2, 722–729. IEEE.

Vogel, C.; Schindler, K.; and Roth, S. 2013. Piecewise rigid scene flow. In *ICCV*, 1377–1384.

Wang, H.; Pang, J.; Lodhi, M. A.; Tian, Y.; and Tian, D. 2021. FESTA: Flow Estimation via Spatial-Temporal Attention for Scene Point Clouds. In *CVPR*, 14173–14182.

Wang, P.-S.; Liu, Y.; Guo, Y.-X.; Sun, C.-Y.; and Tong, X. 2017. O-CNN: Octree-based Convolutional Neural Networks for 3D Shape Analysis. *ACM Trans. Graph.*, 36(4): 72:1–72:11.

Wang, Z.; Li, S.; Howard-Jenkins, H.; Prisacariu, V.; and Chen, M. 2020. FlowNet3D++: Geometric Losses For Deep Scene Flow Estimation. In *WACV*.

Wedel, A.; Rabe, C.; Vaudrey, T.; Brox, T.; Franke, U.; and Cremers, D. 2008. Efficient dense scene flow from sparse or dense stereo data. In *ECCV*, 739–751. Springer.

Wei, Y.; Wang, Z.; Rao, Y.; Lu, J.; and Zhou, J. 2021. PV-RAFT: Point-Voxel Correlation Fields for Scene Flow Estimation of Point Clouds. In *CVPR*.

Weinzaepfel, P.; Revaud, J.; Harchaoui, Z.; and Schmid, C. 2013. DeepFlow: Large displacement optical flow with deep matching. In *ICCV*, 1385–1392.

Wu, W.; Qi, Z.; and Fuxin, L. 2019. Pointconv: Deep convolutional networks on 3d point clouds. In *CVPR*, 9621–9630.

Wu, W.; Wang, Z. Y.; Li, Z.; Liu, W.; and Fuxin, L. 2020. PointPWC-Net: Cost Volume on Point Clouds for (Self-) Supervised Scene Flow Estimation. In *ECCV*, 88–107.

Wu, Z.; Song, S.; Khosla, A.; Yu, F.; Zhang, L.; Tang, X.; and Xiao, J. 2015. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, 1912–1920.

Xie, S.; Gu, J.; Guo, D.; Qi, C. R.; Guibas, L.; and Litany, O. 2020. PointContrast: Unsupervised Pre-training for 3D Point Cloud Understanding. In *ECCV*.

Zach, C.; Pock, T.; and Bischof, H. 2007. A duality based approach for realtime tv-l 1 optical flow. In *Joint pattern recognition symposium*, 214–223. Springer.

Zhang, Z.; Hua, B.-S.; and Yeung, S.-K. 2019. ShellNet: Efficient Point Cloud Convolutional Neural Networks Using Concentric Shells Statistics. In *ICCV*.

Zhao, H.; Jia, J.; and Koltun, V. 2020. Exploring Self-Attention for Image Recognition. In *CVPR*.

Zhao, H.; Jiang, L.; Jia, J.; Torr, P.; and Koltun, V. 2021. Point Transformer. In *ICCV*.

Zhu, X.; Su, W.; Lu, L.; Li, B.; Wang, X.; and Dai, J. 2021. Deformable DETR: Deformable Transformers for End-to-End Object Detection. In *ICLR*.