

# HCRF-Flow: Scene Flow from Point Clouds with Continuous High-order CRFs and Position-aware Flow Embedding

Ruibo Li<sup>1,2</sup>, Guosheng Lin<sup>1,2\*</sup>, Tong He<sup>3</sup>, Fayao Liu<sup>4</sup>, Chunhua Shen<sup>3</sup>

<sup>1</sup>S-Lab, Nanyang Technological University, Singapore

<sup>2</sup>School of Computer Science and Engineering, Nanyang Technological University, Singapore

<sup>3</sup>University of Adelaide, Australia <sup>4</sup>Institute for Infocomm Research, A\*STAR, Singapore

E-mail: ruibo001@e.ntu.edu.sg, gslin@ntu.edu.sg

## Abstract

Scene flow in 3D point clouds plays an important role in understanding dynamic environments. Although significant advances have been made by deep neural networks, the performance is far from satisfactory as only per-point translational motion is considered, neglecting the constraints of the rigid motion in local regions. To address the issue, we propose to introduce the motion consistency to force the smoothness among neighboring points. In addition, constraints on the rigidity of the local transformation are also added by sharing unique rigid motion parameters for all points within each local region. To this end, a high-order CRFs based relation module (Con-HCRFs) is deployed to explore both point-wise smoothness and region-wise rigidity. To empower the CRFs to have a discriminative unary term, we also introduce a position-aware flow estimation module to be incorporated into the Con-HCRFs. Comprehensive experiments on *FlyingThings3D* and *KITTI* show that our proposed framework (HCRF-Flow) achieves state-of-the-art performance and significantly outperforms previous approaches substantially.

## 1. Introduction

Scene flow estimation [37] aims to provide dense or semi-dense 3D vectors, representing the per-point 3D motion in two consecutive frames. The information provided has proven invaluable in analyzing dynamic scenes. Although significant advances have been made in the 2D optical flow, the counterpart in 3D point cloud is far more challenging. This is partly due to the irregularity and sparsity of the data, but also due to the diversity of the scene.

As pointed out in [17], most of the structures in the visual world are rigid or at least nearly so. Many previous top-performing approaches [13, 5, 45, 26] simplify this task as a

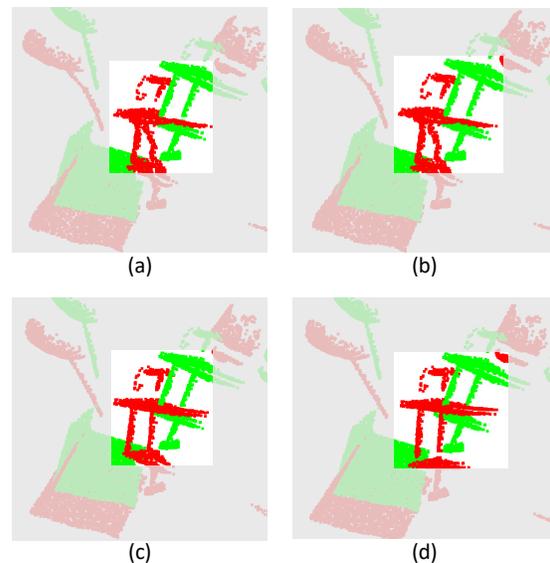


Figure 1. The warped point cloud at the next frame based on different scene flow. Green points represent the point cloud at frame  $t$ . Red points are the warped results at frame  $t + 1$  by adding scene flow back to corresponding green points. (a) scene flow produced by FlowNet3D [13]; (b) scene flow produced by FlowNet3D and refined by a conventional CRF; (c) scene flow produced by FlowNet3D and refined by our continuous high order CRFs; (d) ground truth scene flow. The local structure of the warped point cloud is distorted in FlowNet3D and the conventional CRF but preserved in our method.

regression problem by estimating a point-wise translational motion. Although promising results have been achieved, the performance is far from satisfactory as the potential rigid motion constraints existing in the local region are ignored. As shown in Fig. 1(a), the results generated by the FlowNet3D [13] are deformed and fail to maintain the local geometric smoothness. A straightforward remedy is to utilize pair-wise regularization to smooth the flow prediction. However, ignoring the potential rigid transformations

\*Corresponding author: G. Lin. (e-mail: gslin@ntu.edu.sg)

makes it hard to maintain the underlying spatial structure, as presented in Fig. 1(b).

To address this issue, we propose a novel framework termed HCRF-Flow, which consists of two components: a position-aware flow estimation module (PAFE) for per-point translational motion regression and a continuous high-order CRFs module (Con-HCRFs) for the refinement of the per-point predictions by considering both spatial smoothness and rigid transformation constraints. Specifically, in Con-HCRFs, a pairwise term is designed to encourage neighboring points with similar local structure to have similar motions. In addition, a novel high order term is designed to encourage each point in a local region to take a motion obeying the shared rigid motion parameters, i.e., translation and rotation parameters, in this region.

In point cloud scene flow estimation, it is challenging to aggregate the matching costs, which are calculated by comparing one point with its softly corresponding points. To encode this knowledge into the embedding features, we propose a position-aware flow embedding layer in the PAFE module. In the aggregation step, we introduce a pseudo matching pair that is applied to calculate the difference of the matching cost. For each softly corresponding pair, both its position information and the matching cost difference will be considered to output weights for aggregation.

Our main contributions can be summarized as follows:

- We propose a novel scene flow estimation framework HCRF-Flow by combining the strengths of DNNs and CRFs to perform a per-point translational motion regression and a refinement with both pairwise and region-level regularization;
- Formulating the rigid motion constraints as a high order term, we propose continuous high-order CRFs (Con-HCRFs) to model the interaction of points by imposing point-level and region-level consistency constraints.
- We present a novel position-aware flow estimation layer to build reliable matching costs and aggregate them based on both position information and matching cost differences.
- Our proposed HCRF-Flow significantly outperforms the state-of-the-art on both FlyingThing3D and KITTI Scene Flow 2015 datasets. In particular, we achieve Acc3DR scores of 95.07% and 94.44% on FlyingThing3D and KITTI, respectively.

## 1.1. Related work

**Scene flow from RGB or RGB-D images** Scene flow is first proposed in [37] to represent the three-dimensional motion field of points in a scene. Many works [8, 25, 32, 36, 38, 39, 40, 20, 16, 29, 6] try to recover scene flow from stereo

RGB images or monocular RGB-D images. The local rigidity assumption has been applied in scene flow estimation from images. [39, 40, 20, 16] directly predict the rigidity parameter of each local region to produce scene flow estimates. [38, 29, 6] add a rigidity term into the energy function to constrain the scene flow estimation. Compared with them, our method is different in the following aspects: 1) our method formulates the rigidity constraint as a high order term in Con-HCRFs. It encourages the region-level rigidity of point-wise scene flow rather than directly computing rigidity parameters. Thus, our Con-HCRFs can be easily added to other point cloud scene flow estimation methods as a plug-in module to improve the rigidity of their predictions; 2) our method targets irregular and unordered point cloud data instead of well organized 2D images.

**Deep scene flow from point clouds** Some approaches [4, 35] estimate scene from point cloud via traditional techniques. Recently, inspired by the success of deep learning for point clouds, more works [5, 1, 13, 45, 26, 23, 14, 42] have employed DNNs in this field. [13] estimates scene flow based on PointNet++ [28]. [5] proposes a sparse convolution architecture for scene flow learning, [45] designs a coarse-to-fine scene flow estimation framework, and [26] estimates the point translation by point matching. Despite achieving impressive performance, these methods neglect the rigidity constraints and estimate each point’s motion independently. Although the rigid motion for each point is computed in [1], the per-point rigid parameter is independently regressed by a DNN without fully considering its geometry constraints. Unlike previous methods, we design a novel Con-HCRFs to explicitly model both spatial smoothness and rigid motion constraints.

**Deep learning on 3D point clouds** Many works [27, 28, 34, 18, 44, 15, 41, 7] focus on learning directly on raw point clouds. PointNet [27] and PointNet++ [28] are the pioneering works which use shared Multi-Layer-Perception (MLP) to extract features and a max pooling to aggregate them. [7, 41] use the attention mechanism to produce aggregation weights. [7, 15] encode shape features from local geometry clues to improve the feature extraction. Inspired by these works, we propose a position-aware flow embedding layer to dynamically aggregate matching costs based on both position representations and matching cost differences for better matching cost aggregation.

**Conditional random fields (CRFs)** CRFs are a type of probabilistic graphical models, which are widely used to model the effects of interactions among examples in numerous vision tasks [10, 12, 2, 48, 46]. In point cloud processing, previous works [47, 33, 3] apply CRFs to discrete labeling tasks for spatial smoothness. In contrast to the CRFs in these works, the variables of Con-HCRFs are defined in a continuous domain, and two different relations are modeled by Con-HCRFs at the point level and the region level.

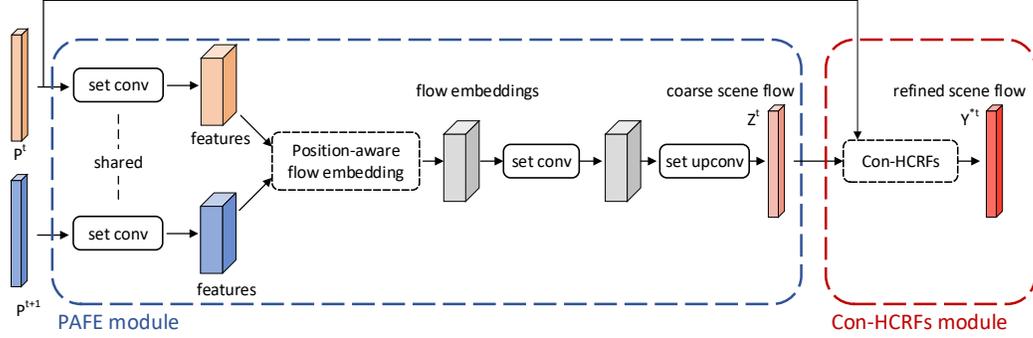


Figure 2. HCRF-Flow architecture. Our HCRF-Flow consists of two components: a PAFE module to produce per-point initial scene flow and a Con-HCRFs module to refine the initial scene flow. Our proposed position-aware flow embedding layer is employed in the PAEF module to encode motion information. We build two different architectures of the PAFE module: one is designed by considering only single-scale feature (similar to FlowNet3D [27]). The other one introduces a pyramid architecture (similar to PointPWC-Net [45]).

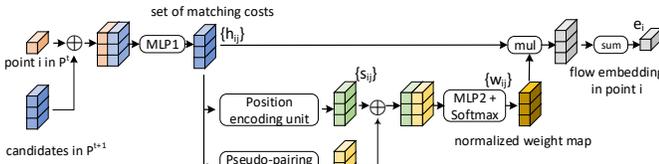


Figure 3. The details of the position-aware flow embedding layer.

## 2. HCRF-Flow

### 2.1. Overview

In the task of point cloud scene flow estimation, the inputs are two point clouds at two consecutive frames:  $\mathbf{P}^t = \{\mathbf{p}_i^t | i = 1, \dots, n_t\}$  at frame  $t$  and  $\mathbf{P}^{t+1} = \{\mathbf{p}_j^{t+1} | j = 1, \dots, n_{t+1}\}$  at frame  $t + 1$ , where  $\mathbf{p}_i^t, \mathbf{p}_j^{t+1} \in \mathbb{R}^3$  are 3D coordinates of individual points. Our goal is to predict the 3D displacement for each point  $i$  in  $\mathbf{P}^t$ , which describes the motion of each point  $i$  from frame  $t$  to frame  $t + 1$ . Unless otherwise stated, we use boldfaced uppercase and lowercase letters to denote matrices and column vectors, respectively.

As shown in Fig. 2, the HCRF-Flow consists of two components: a PAFE module for per-point flow estimation and a Con-HCRFs module for refinement. For the PAFE module, we try two different architectures, a single-scale architecture referring to FlowNet3D [27] and a pyramid architecture referring to PointPWC-Net [45]. To mix the two point clouds, in the PAFE module, we propose a novel position-aware flow embedding layer to build reliable matching costs and aggregate them to produce flow embeddings that encode the motion information. For better aggregation, we use the position information and the matching cost difference as clues to generate aggregation weights. Sec. 2.2 introduces the details about this layer. In the Con-HCRFs module, we propose novel continuous high order CRFs to refine the coarse scene flow by encouraging both point-level and region-level consistency. More details are given in Sec. 3.

### 2.2. Position-aware flow embedding layer

As shown in Fig. 3, the position-aware flow embedding layer aims to produce flow embedding  $e_i$  for each point  $\mathbf{p}_i$  in  $\mathbf{P}^t$ . For each point  $\mathbf{p}_i$ , we first find neighbouring points  $\mathcal{N}_e(i)$  around  $\mathbf{p}_i^t$  in  $t + 1$  frame  $\mathbf{P}^{t+1}$ . Then, following [13], the matching cost between point  $\mathbf{p}_i^t$  and a softly corresponding point  $\mathbf{p}_j^{t+1}$  in  $\mathcal{N}_e(i)$  are addressed as:

$$h_{ij} = h(\mathbf{f}_i^t, \mathbf{f}_j^{t+1}, \mathbf{p}_j^{t+1} - \mathbf{p}_i^t), \quad (1)$$

where  $\mathbf{f}_i^t$  and  $\mathbf{f}_j^{t+1}$  are the features for  $\mathbf{p}_i^t$  and  $\mathbf{p}_j^{t+1}$ , respectively.  $h$  is a concatenation of its inputs followed by a MLP. After obtaining the matching costs for point  $\mathbf{p}_i^t$ , two sub-branches are followed: position encoding unit and pseudo-pairing unit, to produce weights for aggregation, as shown in Fig. 3.

**Pseudo-pairing unit** When aggregating the matching costs, this unit is designed to automatically select prominent ones by assigning them more weights. To this end, we compare each matching pair with a pseudo stationary pair and use the difference as a clue to measure the importance of this matching pair. The pseudo stationary pair represents the situation when this point does not move, *i.e.* the softly corresponding point is itself. Based on Eq. 1, the matching cost of the pseudo stationary pair for each point  $p_i$  can be defined as:

$$\bar{h}_i = h(\mathbf{f}_i^t, \mathbf{f}_i^t, \mathbf{p}_i^t - \mathbf{p}_i^t). \quad (2)$$

The matching cost difference between each matching pair and this pseudo pair can be expressed as:

$$\mathbf{u}_{ij} = h_{ij} - \bar{h}_i. \quad (3)$$

In subsequent aggregation procedure, the matching cost difference  $\mathbf{u}_{ij}$  will be treated as a feature to produce aggregation weights for each matching cost.

**Position encoding unit** Further, to improve the ability of our aggregation, we incorporate the position representations into the aggregation procedure as a significant factor in producing soft weights. Specifically, inspired by [31] and [7],

for each matching pair  $\mathbf{p}_i^t$  and  $\mathbf{p}_j^{t+1}$ , we utilize 3D Euclidean distance, the absolute and relative coordinates as position information to encode the position representation  $\mathbf{s}_{ij}$ , which can be expressed as:

$$\mathbf{s}_{ij} = M_s(\mathbf{p}_i^t \oplus \mathbf{p}_j^{t+1} \oplus (\mathbf{p}_i^t - \mathbf{p}_j^{t+1}) \oplus \|\mathbf{p}_i^t - \mathbf{p}_j^{t+1}\|), \quad (4)$$

where  $M_s(\cdot)$  is a MLP to map the position information into the position representation,  $\oplus$  is the concatenation operation, and  $\|\cdot\|$  computes the Euclidean distance between the two points.

Given the matching cost difference and the position representation, we design a shared function  $M_a(\cdot)$  to produce a unique weight vector for each matching cost for aggregation. Specifically, the function  $M_a(\cdot)$  is composed of a MLP followed by a *softmax* operation to normalize the weights across all matching costs in a set. The normalized weight for each matching cost  $\mathbf{h}_{ij}$  can be written as:

$$\mathbf{w}_{ij} = M_a(\mathbf{u}_{ij} \oplus \mathbf{s}_{ij}). \quad (5)$$

Therefore, for each point  $\mathbf{p}_i$ , according to the learned aggregation weights, the final flow embedding for point  $\mathbf{p}_i^t$  can be expressed as:

$$\mathbf{e}_i = \sum_{j \in \mathcal{N}_e(i)} \mathbf{w}_{ij} \odot \mathbf{h}_{ij}, \quad (6)$$

where  $\odot$  is the element-wise multiplication and  $\mathcal{N}_e(i)$  is the set of softly corresponding points for point  $\mathbf{p}_i^t$  in next frame.

### 3. Continuous High-Order CRFs

In this section, we introduce the details of our continuous high order CRFs. We first formulate the problem of scene flow refinement. Then, we describe the details of three kinds of potential functions involved in the Con-HCRFs. Lastly, we discuss how to utilize mean field theory to approximate the Con-HCRFs distribution and obtain the final iterative inference algorithm.

#### 3.1. Overview

Take a point cloud  $\mathbf{P}$  with  $n$  points, indexed  $1, 2, \dots, n$ . In the scene flow refinement, we attempt to assign every point a refined 3D displacement based on the initial scene flow produced by the PAFE module. Let  $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_n]$  be a matrix of 3D displacements corresponding to all  $n$  points in point cloud  $\mathbf{P}$ , where each  $\mathbf{y}_i \in \mathbb{R}^3$ . Following [12, 30], we model the conditional probability distribution with the following density function  $\Pr(\mathbf{Y}|\mathbf{P}) = \frac{1}{Z(\mathbf{P})} \exp(-E(\mathbf{Y}|\mathbf{P}))$ . Here  $E$  is the energy function and  $Z$  is the partition function defined as:  $Z(\mathbf{P}) = \int_{\mathbf{Y}} \exp(-E(\mathbf{Y}|\mathbf{P})) d\mathbf{Y}$ .

Different from conventional CRFs, the Con-HCRFs proposed in this paper contains a novel high order potential that performs rigid motion constraints in each local region. Specifically, the energy function is defined as:

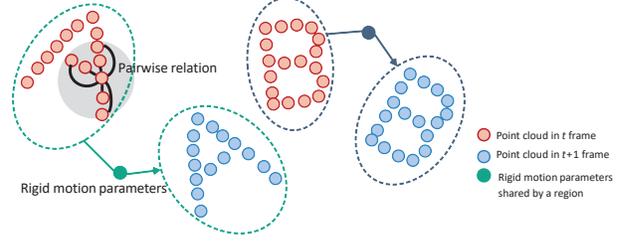


Figure 4. Illustration of Con-HCRFs. Red points and blue points represent the point cloud in frame  $t$  and frame  $t + 1$ , respectively. The black lines and gray background represent the pairwise relations and the neighborhood in the pairwise term. The dashed boxes cover rigid regions. An arrow with a point represents the rigid motion of a region with rigid motion parameters shared by all the points in it. The rigid motion constraints compose the high order term in Con-HCRFs.

$$E(\mathbf{Y}|\mathbf{P}) = \sum_i \psi^U(\mathbf{y}_i, \mathbf{P}) + \sum_{i,j \in \mathcal{N}(i)} \psi^B(\mathbf{y}_i, \mathbf{y}_j, \mathbf{P}) + \sum_{\mathcal{V} \in \mathbf{V}_{set}} \sum_{i \in \mathcal{V}} \psi^{SV}(\mathbf{y}_i, \mathbf{Y}_{\mathcal{V}-i}, \mathbf{P}), \quad (7)$$

where  $\mathcal{N}(i)$  represents the set of neighboring points of center point  $i$ ;  $\mathbf{V}_{set}$  represents the set of rigid regions in the whole point cloud and  $\mathbf{Y}_{\mathcal{V}-i}$  is a matrix composed by the scene flow of points belonging to the region  $\mathcal{V}$  with the point  $i$  excluded. And the point set without the point  $i$  is denoted as  $\mathcal{V} - i$ . The unary term  $\psi^U$  encourages the refined scene flow to be consistent with the initial scene flow. The pairwise term  $\psi^B$  encourages neighboring points with similar local structure to take similar displacements. The high order term  $\psi^{SV}$  encourages points belonging to the same rigid region to share the same rigid motion parameters. In this paper, we use over-segmentation method to segment the entire point cloud into a series of supervoxels and treat each supervoxel as the rigid region in the high order term. An illustration is shown in Fig. 4. We drop the conditioning on  $\mathbf{P}$  in the rest of this paper for convenience.

#### 3.2. Potential functions

**Unary potential** The unary potential is constructed from the initial scene flow by considering  $L^2$  norm:

$$\psi^U(\mathbf{y}_i) = \|\mathbf{y}_i - \mathbf{z}_i\|_2^2, \quad (8)$$

where  $\mathbf{z}_i$  represents the initial 3D displacement at point  $i$  produced by PAFE module;  $\|\cdot\|_2$  denotes the  $L^2$  norm of a vector.

**Pairwise potential** The pairwise potential is constructed from  $C$  types of similarity observations to describe the relation between pairs of hidden variables  $\mathbf{y}_i$  and  $\mathbf{y}_j$ :

$$\psi^B(\mathbf{y}_i, \mathbf{y}_j) = \sum_{c=1}^C \alpha_c K_{ij}^{(c)} \|\mathbf{y}_i - \mathbf{y}_j\|_2^2, \quad (9)$$

where  $K_{ij}^{(c)}$  is a weight to specify the relation between the points  $i$  and  $j$ ;  $\alpha_c$  denotes the coefficient for each similarity measure. Specifically, we set weight  $K_{ij}^{(c)}$  depending on a Gaussian kernel  $K_{ij}^{(c)} = \exp(-\frac{\|\mathbf{k}_i^c - \mathbf{k}_j^c\|_2^2}{2\theta_c^2})$ , where  $\mathbf{k}_i^c$  and  $\mathbf{k}_j^c$  indicate the features of neighboring points  $i$  and  $j$  associating with similarity measure  $c$ ;  $\theta_c$  is the kernel's bandwidth parameter. In this paper, we use point position and surface normal as the observations to construct two Gaussian kernels.

**High order potential** For the high order potential term, we want to explore the effects of interactions among points in a supervoxel. According to the rigid motion constraint, the high order potential term in CRF can be defined as:

$$\psi^{SV}(\mathbf{y}_i, \mathbf{Y}_{\mathcal{V}-i}) = \beta \|\mathbf{y}_i - \mathbf{g}(\mathbf{p}_i, \mathbf{Y}_{\mathcal{V}-i})\|_2^2, \quad (10)$$

where  $\mathbf{g}(\mathbf{p}_i, \mathbf{Y}_{\mathcal{V}-i})$  is a displacement produced by a function  $\mathbf{g}(\cdot, \cdot)$ , where shared rigid motion parameters will be computed by  $\mathbf{Y}_{\mathcal{V}-i}$  and the displacement for point  $i$  obeying the shared parameters can be obtained by applying the shared parameters back to the original position  $\mathbf{p}_i$ .  $\beta$  is a coefficient. In the following, we will give details about the computation of  $\mathbf{g}(\mathbf{p}_i, \mathbf{Y}_{\mathcal{V}-i})$ .

In a rigid region  $\mathcal{V}$ , given point  $i$ , we denote the points in region  $\mathcal{V}$  not containing the point  $i$  as  $\mathcal{P}_{\mathcal{V}-i} = \{\mathbf{p}_j \in \mathcal{P} | j \in \mathcal{V} - i\}$  and the corresponding 3D displacements as  $\mathbf{Y}_{\mathcal{V}-i} = \{\mathbf{y}_j \in \mathbf{Y} | j \in \mathcal{V} - i\}$ . The warped positions in the next frame can be obtained by adding the scene flow back to the corresponding positions in frame  $t$ :

$$\mathbf{D}_{\mathcal{V}-i} = \{\mathbf{d}_j(\mathbf{y}_j) = \mathbf{p}_j + \mathbf{y}_j | j \in \mathcal{V} - i\}. \quad (11)$$

The possible rigid transformation from  $\mathcal{P}_{\mathcal{V}-i}$  to  $\mathbf{D}_{\mathcal{V}-i}$  can be defined as  $[\mathbf{R}, \mathbf{t}]$  where  $\mathbf{R} \in \text{SO}(3)$  and  $\mathbf{t} \in \mathbb{R}^3$ . Inspired by the work [43] in point cloud registration, we can minimize the mean-squared error  $E^{rigid}(\mathbf{R}, \mathbf{t}; \mathbf{Y}_{\mathcal{V}-i})$  to find the most suitable rigid motion parameters  $[\mathbf{R}^*(\mathbf{Y}_{\mathcal{V}-i}), \mathbf{t}^*(\mathbf{Y}_{\mathcal{V}-i})]$  to describe the motion  $\mathbf{Y}_{\mathcal{V}-i}$ :

$$E^{rigid}(\mathbf{R}, \mathbf{t}; \mathbf{Y}_{\mathcal{V}-i}) = \frac{1}{N_{\mathcal{V}-i}} \sum_{j=1}^{N_{\mathcal{V}-i}} \|\mathbf{R}\mathbf{p}_j + \mathbf{t} - \mathbf{d}_j(\mathbf{y}_j)\|_2^2, \quad (12)$$

$$[\mathbf{R}^*(\mathbf{Y}_{\mathcal{V}-i}), \mathbf{t}^*(\mathbf{Y}_{\mathcal{V}-i})] = \arg \min_{\mathbf{R}, \mathbf{t}} E^{rigid}(\mathbf{R}, \mathbf{t}; \mathbf{Y}_{\mathcal{V}-i}), \quad (13)$$

where  $N_{\mathcal{V}-i}$  is the number of points in region  $\mathcal{V}$  with the point  $i$  excluded.

Define centers of  $\mathcal{P}_{\mathcal{V}-i}$  and  $\mathbf{D}_{\mathcal{V}-i}$  as  $\bar{\mathbf{p}}_{\mathcal{V}-i} = \frac{1}{N_{\mathcal{V}-i}} \sum_{j=1}^{N_{\mathcal{V}-i}} \mathbf{p}_j$  and  $\bar{\mathbf{d}}_{\mathcal{V}-i} = \frac{1}{N_{\mathcal{V}-i}} \sum_{j=1}^{N_{\mathcal{V}-i}} \mathbf{d}_j$ , respectively. Then the cross-covariance matrix  $\mathbf{H}_{\mathcal{V}-i}$  can be written as:  $\mathbf{H}_{\mathcal{V}-i} = \sum_{j=1}^{N_{\mathcal{V}-i}} (\mathbf{p}_j - \bar{\mathbf{p}}_{\mathcal{V}-i})(\mathbf{d}_j - \bar{\mathbf{d}}_{\mathcal{V}-i})^\top$ . Using the singular value decomposition (SVD) to decompose  $\mathbf{H}_{\mathcal{V}-i} = \mathbf{U}_{\mathcal{V}-i} \mathbf{S}_{\mathcal{V}-i} \mathbf{V}_{\mathcal{V}-i}^\top$ , we can get the closed-form solutions of

$E^{rigid}(\cdot, \cdot)$ , written as:

$$\begin{aligned} \mathbf{R}^*(\mathbf{Y}_{\mathcal{V}-i}) &= \mathbf{V}_{\mathcal{V}-i} \mathbf{U}_{\mathcal{V}-i}^\top \\ \mathbf{t}^*(\mathbf{Y}_{\mathcal{V}-i}) &= -\mathbf{R}^*(\mathbf{Y}_{\mathcal{V}-i}) \bar{\mathbf{p}}_{\mathcal{V}-i} + \bar{\mathbf{d}}_{\mathcal{V}-i}. \end{aligned} \quad (14)$$

When treating the most suitable parameters  $[\mathbf{R}^*(\mathbf{Y}_{\mathcal{V}-i}), \mathbf{t}^*(\mathbf{Y}_{\mathcal{V}-i})]$  as the shared rigid motion parameters by all points in region  $\mathcal{V}$ , the displacement that satisfies the rigid motion constraints for point  $i$  is given by:

$$\mathbf{g}(\mathbf{p}_i, \mathbf{Y}_{\mathcal{V}-i}) = \mathbf{R}^*(\mathbf{Y}_{\mathcal{V}-i}) \mathbf{p}_i + \mathbf{t}^*(\mathbf{Y}_{\mathcal{V}-i}) - \mathbf{p}_i. \quad (15)$$

### 3.3. Inference

In order to produce the most probable scene flow, we should solve the MAP inference problem for  $\text{Pr}(\mathbf{Y})$ . Following [30], we approximate the original conditional distribution  $\text{Pr}(\mathbf{Y})$  by mean field theory [9]. Thus, the distribution  $\text{Pr}(\mathbf{Y})$  is approximated by a product of independent marginals, i.e.,  $Q(\mathbf{Y}) = \prod_{i=1}^n Q_i(\mathbf{y}_i)$ . By minimizing the KL-divergence between  $\text{Pr}$  and  $Q$ , the solution for  $Q$  can be written as:  $\log(Q_i(\mathbf{y}_i)) = \text{E}_{j \neq i}[\text{Pr}(\mathbf{Y})] + \text{const}$ , where  $\text{E}_{j \neq i}$  represents an expectation under  $Q$  distributions over all variable  $\mathbf{y}_j$  for  $j \neq i$ .

Following [30], we represent each  $\log(Q_i(\mathbf{y}_i))$  as a multivariate normal distribution, the mean field update for mean  $\mu_i$  and normalization parameter  $\sigma_i$  can be written as:

$$\sigma_i = \frac{1}{2(1 + 2 \sum_{c=1}^C \sum_{j \in \mathcal{N}(i)} \alpha_c K_{ij}^{(c)} + \beta)}, \quad (16)$$

$$\mu_i = 2\sigma_i(\mathbf{z}_i + 2 \sum_{c=1}^C \sum_{j \in \mathcal{N}(i)} \alpha_c K_{ij}^{(c)} \mu_j + \beta \mathbf{g}(\mathbf{p}_i, \mathbf{M}_{\mathcal{V}-i})), \quad (17)$$

where  $\mathbf{M}_{\mathcal{V}-i}$  is a set of mean  $\mu_j$  for all  $j \in \mathcal{V} - i$ ;  $\sigma_i$  is the diagonal element of covariance  $\Sigma_i$ . The detailed derivation of the inference algorithm can be found in supplementary. We observe that there usually exist hundreds of points in a supervoxel, which makes the rigid parameters computed on all points in the supervoxel excluding the point  $i$  vary close to the rigid parameters computed on all points in the supervoxel, i.e.,  $[\mathbf{R}^*(\mathbf{M}_{\mathcal{V}-i}), \mathbf{t}^*(\mathbf{M}_{\mathcal{V}-i})]$  is vary close to  $[\mathbf{R}^*(\mathbf{M}_{\mathcal{V}}), \mathbf{t}^*(\mathbf{M}_{\mathcal{V}})]$ . Thus, in practice, we approximate  $\mathbf{g}(\mathbf{p}_i, \mathbf{M}_{\mathcal{V}-i})$  in Eq. 17 with  $\mathbf{g}(\mathbf{p}_i, \mathbf{M}_{\mathcal{V}})$ , and the approximated mean  $\mu_i$  is:

$$\mu_i = 2\sigma_i(\mathbf{z}_i + 2 \sum_{c=1}^C \sum_{j \in \mathcal{N}(i)} \alpha_c K_{ij}^{(c)} \mu_j + \beta \mathbf{g}(\mathbf{p}_i, \mathbf{M}_{\mathcal{V}})). \quad (18)$$

After this approximation, we only need to calculate a set of rigid motion parameters for each supervoxel rather than for each point, which greatly reduces the time complexity.

In the MAP inference, since we approximate  $\text{Pr}$  with  $Q$ , an estimate of each  $\mathbf{y}_i$  can be obtained by computing the expected value  $\mu_i$  of the Gaussian distribution  $Q_i$ :

$$\mathbf{y}_i^* = \arg \max_{\mathbf{y}_i} Q_i(\mathbf{y}_i) = \mu_i. \quad (19)$$

The inference procedure of our Con-HCRFs can be clearly sketched in Algorithm 1.

---

**Algorithm 1** Mean field in Con-HCRFs

---

**Input:** Coarse scene flow  $\mathbf{Z}$ ; Coordinates of point cloud  $\mathbf{P}$ ;**Output:** Refined scene flow  $\mathbf{Y}^*$  ;**Procedure:**

- 1:  $\mu_i \leftarrow z_i$  for all  $i$ ;  
    ▶ Initialization
  - 2: **while** not converged **do**
  - 3:   Compute  $[\mathbf{R}^*(\mathcal{M}_{\mathcal{V}}), \mathbf{t}^*(\mathcal{M}_{\mathcal{V}})]$  for supervoxel  $\mathcal{V}$ ;
  - 4:    $\hat{\mu}_i \leftarrow \mathbf{R}^*(\mathcal{M}_{\mathcal{V}})\mathbf{p}_i + \mathbf{t}^*(\mathcal{M}_{\mathcal{V}}) - \mathbf{p}_i$ ;  
    ▶ Message passing from supervoxel
  - 5:    $\tilde{\mu}_i^{(c)} \leftarrow \sum_{j \in \mathcal{N}(i)} K_{ij}^{(c)} \mu_j$ ;
  - 6:    $\tilde{\sigma}_i^{(c)} \leftarrow \sum_{j \in \mathcal{N}(i)} K_{ij}^{(c)}$ ;  
    ▶ Message passing from neighboring points
  - 7:    $\bar{\mu}_i \leftarrow z_i + 2 \sum_c \alpha_c \tilde{\mu}_i^{(c)} + \beta \hat{\mu}_i$ ;
  - 8:    $\bar{\sigma}_i \leftarrow 1 + 2 \sum_c \alpha_c \tilde{\sigma}_i^{(c)} + \beta$ ;  
    ▶ Weighted summing
  - 9:    $\mu_i \leftarrow \frac{1}{\bar{\sigma}_i} \bar{\mu}_i$ ;  
    ▶ Normalizing
  - 10: **end while**
  - 11:  $\mathbf{y}_i^* \leftarrow \mu_i$  for all  $i$ .
- 

Moreover, thanks to the differentiable SVD function provided by PyTorch [24], the mean field update operation is differentiable in our inference procedure. Therefore, following [49, 46], our mean field algorithm can be fully integrated with deep learning models, which ensures the end-to-end training of the whole framework.

## 4. Experiments

In this section, we first train and evaluate our method on the synthetic FlyingThings3D dataset in Sec. 4.1, and then in Sec. 4.2 we test the generalization ability of our method on real-world KITTI dataset without fine-tuning. In Sec. 4.3, we validate the generality of our Con-HCRFs on other networks. Finally, we conduct ablation studies to analyze the contribution of each component in Sec. 4.4. Note that in the following experiments, there are two different architectures of the PAFE module, the single-scale one denoted as PAFE-S and the pyramid one denoted as PAFE. And corresponding HCRF-Flow models are denoted as HCRF-Flow-S and HCRF-Flow, respectively.

**Evaluation metrics.** Let  $\mathbf{Y}^*$  denote the predicted scene flow, and  $\mathbf{Y}_{gt}$  be the ground truth scene flow. The evaluate metrics are computed as follows. **EPE3D(m)**: the main metric,  $\|\mathbf{Y}^* - \mathbf{Y}_{gt}\|_2$  average over each point. **Acc3DS(%)**: the percentage of points whose EPE3D  $< 0.05\text{m}$  or relative error  $< 5\%$ . **Acc3DR(%)**: the percentage of points whose EPE3D  $< 0.1\text{m}$  or relative error  $< 10\%$ . **Outliers3D(%)**: the percentage of points whose EPE3D  $> 0.3\text{m}$  or relative error  $> 10\%$ . **EPE2D(px)**: 2D End Point Error, which is a

common metric for optical flow. **Acc2D(%)**: the percentage of points whose EPE2D  $< 3\text{px}$  or relative error  $< 5\%$ .

### 4.1. Results on FlyingThings3D

FlyingThings3D [19] is a large-scale synthetic dataset. We follow [5] to build the training set and the test set. Our method takes  $n = 8,192$  points in each point cloud as input. We train our models on one quarter of the training set (4,910 pairs) and evaluate on the whole test set (3,824 pairs).

Referring to PointPWC-Net [45], we build a pyramid PAFE module, PAFE, and corresponding HCRF-Flow framework, HCRF-Flow. Note that, compared with original architecture in [45], there are three adjustments in our PAFE: 1) we replace the MLPs in level 0 with a set conv [13]; 2) we replace all PointConvns [44] with set convs [13]; 3) we replace cost volume layers [45] with our position-aware flow embedding layers. In Con-HCRFs, we utilize the algorithm proposed in [11] for supervoxel segmentation. During training, we first train our PAFE with a multi-scale loss function used in [45]. Then we add the Con-HCRFs to PAFE for fine-tuning. More implementation details are in supplementary.

The quantitative evaluation results on the Flyingthings3D are shown in Table 1. We compare our method with four baseline models: FlowNet3D [13], HPLFlowNet [5], PointPWC-Net [45], and FLOT [26]. As shown in Table 1, our PAFE module outperforms the above four methods. Further, adding Con-HCRFs and fine-tuning on FlyingThings3D, the final method, HCRF-Flow, achieves the best performance on all metrics. Qualitative results are shown in Fig. 5.

### 4.2. Generalization results on KITTI

KITTI Scene Flow 2015 [22, 21] is a well-known dataset for 3D scene flow estimation. In this section, in order to evaluate the generalization ability of our method, we train our model on FlyingThings3D dataset and test on KITTI Scene Flow 2015 without fine-tuning. And the desired supervoxel size and the bandwidth parameters in KITTI are the same as those in FlyingThings3D.

Following [13, 5], we evaluate on all 142 scenes in the training set and remove the points on the ground by height ( $< 0.3\text{m}$ ) for a fair comparison. The quantitative evaluation results on the KITTI are shown in Table 1. Our method outperforms the competing methods, which represents the good generalization ability of our method on real-world data. Fig. 5 shows the qualitative results.

### 4.3. Generality of Con-HCRFs on other models

In this section, we study the generalization ability of Con-HCRFs by applying it to the other scene flow estimation models as a post-processing module. We evaluate the performance of our proposed Con-HCRFs with

Table 1. Evaluation results on FlyingThings3D and KITTI Scene Flow 2015. Our model outperforms all baselines on all evaluation metrics. Especially, the good performance on KITTI shows the generalization ability of our method.

Dataset	Method	EPE3D↓	Acc3DS↑	Acc3DR↑	Outliers3D↓	EPE2D↓	Acc2D↑
FlyingThings3D	FlowNet3D [13]	0.0886	41.63	81.61	58.62	4.7142	60.10
	HPLFlowNet [5]	0.0804	61.44	85.55	42.87	4.6723	67.64
	PointPWC-Net [45]	0.0588	73.79	92.76	34.24	3.2390	79.94
	FLOT [26]	0.0520	73.20	92.70	35.70	-	-
	Ours (PAFE module)	0.0535	78.90	94.93	30.51	2.8253	83.46
	Ours (HCRF-Flow)	<b>0.0488</b>	<b>83.37</b>	<b>95.07</b>	<b>26.14</b>	<b>2.5652</b>	<b>87.04</b>
KITTI	FlowNet3D [13]	0.1069	42.77	79.78	41.38	4.3424	57.51
	HPLFlowNet [5]	0.1169	47.83	77.76	41.03	4.8055	59.38
	PointPWC-Net [45]	0.0694	72.81	88.84	26.48	3.0062	76.73
	FLOT [26]	0.0560	75.50	90.80	24.20	-	-
	Ours (PAFE module)	0.0646	80.29	93.47	20.24	2.4829	80.80
	Ours (HCRF-Flow)	<b>0.0531</b>	<b>86.31</b>	<b>94.44</b>	<b>17.97</b>	<b>2.0700</b>	<b>86.56</b>

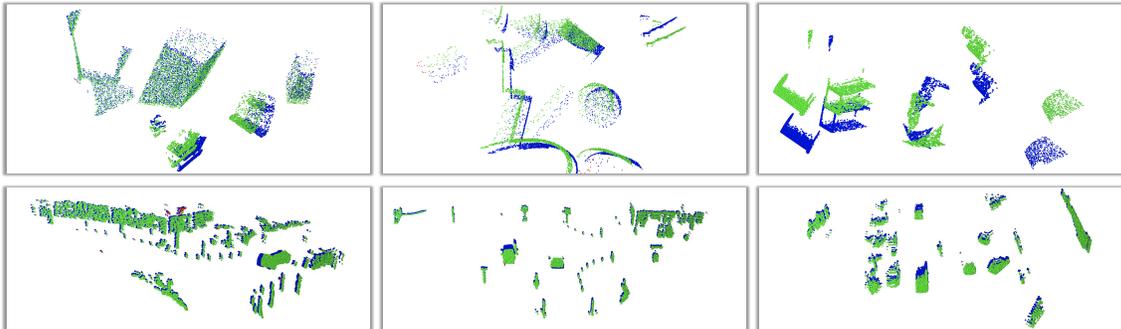


Figure 5. Qualitative results on FlyingThings3D (top) and KITTI (bottom). Blue points are point cloud at frame  $t$ . Green points are the warped results at frame  $t + 1$  for the points, whose predicted displacements are measured as correct by Acc3DR. For the incorrect predictions, we use the ground-truth scene flow to replace them. And the ground truth warped results are shown as red points.

Table 2. Generalization results of Con-HCRFs on FlowNet3D and FLOT models.  $\Delta$  denotes the difference in metrics with respect to each original model. Although built upon strong baselines, our proposed Con-HCRFs boost the performance of each baseline by a large margin on the two datasets.

Dataset	Method	Acc3DS↑	$\Delta$ Acc3DS
FlyingThings3D	FlowNet3D [13]	41.63	0.0
	+ Con-HCRFs	47.01	+ 5.38
	FLOT [26]	73.20	0.0
	+ Con-HCRFs	78.63	+ 5.43
KITTI	FlowNet3D [13]	42.77	0.0
	+ Con-HCRFs	46.90	+ 4.13
	FLOT [26]	75.50	0.0
	+ Con-HCRFs	85.44	+ 9.94

FlowNet3D [13] and FLOT [26], which have shown strong capability on both challenging synthetic data from FlyingThings3D and real Lidar scans from KITTI. The results are presented in Table 2. Although built upon strong baselines, our proposed Con-HCRFs boost the performance of each baseline by a large margin on both datasets, demonstrating the strong robustness and generalization.

#### 4.4. Ablation studies

In this section, we provide a detailed analysis of every component in our method. All experiments are conducted

in the FlyThings3D dataset. Besides the pyramid models, PAFE and HCRF-Flow, for comprehensive analysis, we also evaluate the performance of each component when used in single-scale models, PAFE-S and HCRF-Flow-S. These models are designed referring to FlowNet3D [13].

**Ablation for position-aware flow embedding layer.** We explore the effect of the aggregation strategy on our position-aware flow embedding layer. This strategy is introduced to dynamically aggregate the matching costs considering position information and matching cost differences. As shown in Table 3, for the two baselines, when applying both the pseudo-pairing unit and the position encoding unit to the flow embedding layer, the performance in Acc3DS can be improved by around 8. Moreover, to verify the effectiveness of the pseudo pair, we design a naive dynamic aggregation unit, denoted as **NDA**, which directly produces weights from matching costs rather than the matching cost difference between each matching pair and this pseudo pair. As shown in Table 3, after replacing **PP** with **NDA**, the improvement in Acc3DS decreases from 6.79 to 2.62. Thus, the pseudo-pairing unit is a better choice for this task.

**Ablation for Con-HCRFs.** To ensure the spatial smoothness and the local rigidity of the final predictions, we propose continuous CRFs with a novel high order term. The ablation results for Con-HCRFs are presented in Table 4.

Table 3. Ablation study for position-aware flow embedding layer. **MP**: Maxpooling. **PP**: pseudo-pairing unit. **PE**: position encoding unit. **NDA**: naive dynamic aggregation unit.  $\Delta$  denotes the difference in metrics with respect to each baseline model.

Method	MP	PP	PE	NDA	Acc3DS $\uparrow$	$\Delta$ Acc3DS
Single-scale baseline	✓				41.63	0.00
+NDA				✓	44.25	+ 2.62
+PP		✓			48.42	+ 6.79
+PP+PE (PAFE-S module)		✓	✓		<b>50.08</b>	+ 8.45
Pyramid baseline	✓				69.94	0.0
+PP+PE (PAFE module)		✓	✓		<b>78.90</b>	+ 8.96

Table 4. Ablation study for Con-HCRFs. **Unary**: unary term. **Pair**: pairwise term. **High-order**: our proposed high order term. **naive Region**: a naive regional term designed as a reference to verify the effectiveness of our high order term.  $\Delta$  denotes the difference in metrics with respect to PAFE or PAFE-S module, whose details are introduced in Table 3.  $\dagger$  means jointly optimizing Con-HCRFs and PAFE modules.

Method	Acc3DS $\uparrow$	$\Delta$ Acc3DS
PAFE-S module	50.08	0.00
+ (Unary+Pair)	50.24	+ 0.16
+ (Unary+Pair+naive Region)	48.10	- 1.98
+ (Unary+Pair+High-order)/Con-HCRFs	54.51	+ 4.43
+ (Unary+Pair+High-order)/Con-HCRFs $\dagger$	<b>56.29</b>	+ 6.21
PAFE module	78.90	0.00
+ (Unary+Pair+High-order)/Con-HCRFs	81.39	+ 2.49
+ (Unary+Pair+High-order)/Con-HCRFs $\dagger$	<b>83.37</b>	+ 4.47

With the help of a pairwise term, denoted as (Unary+Pair), the performance gains a slight improvement due to the fact that this pairwise term aims at spatial smoothness but ignores the potential rigid motion constraints. Our proposed Con-HCRFs module, which formulates the rigid motion constraints as its high order term, boosts the performance by a large margin for PAFE-S and PAFE modules. After jointly optimizing Con-HCRFs and PAFE modules, we observe further improvement.

**Can we replace the rigid motion constraints by a region-level smoothness in a supervoxel?** We want to explore whether the rigid motion constraint is a good approach to model the relations among points in a supervoxel. Instead of sharing unique rigid motion parameters, a straightforward approach is to encourage the points among a rigid region to share the same motion, i.e., encourage region-level smoothness in a supervoxel. To this end, we design a naive regional term as:  $\psi^{naive}(\mathbf{y}_i, \mathbf{Y}_{\mathcal{V}}) = \beta \|\mathbf{y}_i - \mathbf{g}^{naive}(\mathbf{p}_i, \mathbf{Y}_{\mathcal{V}})\|^2$ , where  $\mathbf{g}^{naive}(\mathbf{p}_i, \mathbf{Y}_{\mathcal{V}})$  is an average of  $\mathbf{Y}_{\mathcal{V}}$  over all points in  $\mathcal{V}$ . The results are shown in Table 4, denoted as (Unary+Pair+naive Region). As it only enforces spatial smoothness in a region and fails to model suitable dependencies among points in this rigid region, this kind of CRFs is ineffective and even worsen the performance. In contrast, when applying our proposed Con-HCRFs, the final scene flow shows significant improvements.

**Speed analysis of Con-HCRFs** Table 5 reports the average runtime of each component of Con-HCRFs tested on a

Table 5. Time consumption of Con-HCRFs.

Component	Supervoxel	Pairwise term	High order term	Total
Time (ms)	115.1	12.3	100.8	228.2

Table 6. The impact of point number of each supervoxel on our method.

Desired point number	80	100	140	200	PAFE-S
EPE3D $\downarrow$	0.0804	0.0788	<b>0.0782</b>	0.0790	0.0815

single 1080ti GPU. As shown in Table 5, the Con-HCRFs takes 0.2s to process a scene with 8192 points. The speed of Con-HCRFs is similar to DenseCRF [10], which also takes about 0.2s to process a 320x213 image. Additionally, due to the approximate computation that we apply in the high order term, this term only takes 0.1s for a scene. In contrast, the time for the term will dramatically increase from 0.1s to 14s, if the rigid motion parameters are calculated for each point instead of supervoxel. The large gap of runtime shows that the approximation discussed in Sec. 3.3 can significantly boost the efficiency of our Con-HCRFs.

**Impact of supervoxel sizes.** To illustrate the sensitivity to supervoxel sizes, we test our method when facing supervoxels with different point numbers. As shown in Table 6, the method achieves the best performance when the desired point number of each supervoxel is set to a range of 140 to 200.

## 5. Conclusions

In this paper, we have proposed a novel point cloud scene flow estimation method, termed HCRF-Flow, by incorporating the strengths of DNNs and CRFs to perform translational motion regression on each point and operate refinement with both pairwise and region-level regularization. Formulating the rigid motion constraints as a high order term, we propose a novel high-order CRF based relation module (Con-HCRFs) considering both point-level and region-level consistency. In addition, we design a position-aware flow estimation layer for better matching cost aggregation. Experimental results on FlyingThings3D and KITTI datasets show that our proposed method performs favorably against comparison methods. We have also shown the generality of our Con-HCRFs on other point cloud scene flow estimation methods.

## 6. Acknowledgements

This research was conducted in collaboration with SenseTime. This work is supported by A\*STAR through the Industry Alignment Fund - Industry Collaboration Projects Grant. This work is also supported by the National Research Foundation, Singapore under its AI Singapore Programme (AISG Award No: AISG-RP-2018-003), and the MOE Tier-1 research grants: RG28/18 (S) and RG22/19 (S).

## References

- [1] Aseem Behl, Despoina Paschalidou, Simon Donné, and Andreas Geiger. Pointflownet: Learning representations for rigid motion estimation from point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7962–7971, 2019. 2
- [2] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017. 2
- [3] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3075–3084, 2019. 2
- [4] 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. 2
- [5] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3254–3263, 2019. 1, 2, 6, 7
- [6] Michael Hornacek, Andrew Fitzgibbon, and Carsten Rother. Spheroflow: 6 dof scene flow from rgb-d pairs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3526–3533, 2014. 2
- [7] Qingyong Hu, Bo Yang, Linhai Xie, Stefano Rosa, Yulan Guo, Zhihua Wang, Niki Trigoni, and Andrew Markham. Randla-net: Efficient semantic segmentation of large-scale point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11108–11117, 2020. 2, 3
- [8] Frédéric Hugué and Frédéric Devernay. A variational method for scene flow estimation from stereo sequences. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–7. IEEE, 2007. 2
- [9] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009. 5
- [10] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In *Advances in neural information processing systems*, pages 109–117, 2011. 2, 8
- [11] Yangbin Lin, Cheng Wang, Dawei Zhai, Wei Li, and Jonathan Li. Toward better boundary preserved supervoxel segmentation for 3d point clouds. *ISPRS journal of photogrammetry and remote sensing*, 143:39–47, 2018. 6
- [12] Fayao Liu, Chunhua Shen, and Guosheng Lin. Deep convolutional neural fields for depth estimation from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5162–5170, 2015. 2, 4
- [13] Xingyu Liu, Charles R Qi, and Leonidas J Guibas. Flownet3d: Learning scene flow in 3d point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 529–537, 2019. 1, 2, 3, 6, 7
- [14] Xingyu Liu, Mengyuan Yan, and Jeannette Bohg. Meteor-net: Deep learning on dynamic 3d point cloud sequences. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9246–9255, 2019. 2
- [15] Yongcheng Liu, Bin Fan, Shiming Xiang, and Chunhong Pan. Relation-shape convolutional neural network for point cloud analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8895–8904, 2019. 2
- [16] Wei-Chiu Ma, Shenlong Wang, Rui Hu, Yuwen Xiong, and Raquel Urtasun. Deep rigid instance scene flow. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3614–3622, 2019. 2
- [17] D Man and A Vision. A computational investigation into the human representation and processing of visual information, 1982. 1
- [18] Jiageng Mao, Xiaogang Wang, and Hongsheng Li. Interpolated convolutional networks for 3d point cloud understanding. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1578–1587, 2019. 2
- [19] 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*, pages 4040–4048, 2016. 6
- [20] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3061–3070, 2015. 2
- [21] Moritz Menze, Christian Heipke, and Andreas Geiger. Joint 3d estimation of vehicles and scene flow. *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences*, 2, 2015. 6
- [22] Moritz Menze, Christian Heipke, and Andreas Geiger. Object scene flow. *ISPRS Journal of Photogrammetry and Remote Sensing*, 140:60–76, 2018. 6
- [23] Himangi Mittal, Brian Okorn, and David Held. Just go with the flow: Self-supervised scene flow estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11177–11185, 2020. 2
- [24] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 6
- [25] Jean-Philippe Pons, Renaud Keriven, and Olivier Faugeras. Multi-view stereo reconstruction and scene flow estimation with a global image-based matching score. *International Journal of Computer Vision*, 72(2):179–193, 2007. 2
- [26] Gilles Puy, Alexandre Boulch, and Renaud Marlet. Flot: Scene flow on point clouds guided by optimal transport. *arXiv preprint arXiv:2007.11142*, 2020. 1, 2, 6, 7
- [27] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 2, 3

- [28] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in neural information processing systems*, pages 5099–5108, 2017. 2
- [29] Julian Quiroga, Thomas Brox, Frédéric Devernay, and James Crowley. Dense semi-rigid scene flow estimation from rgbd images. In *European Conference on Computer Vision*, pages 567–582. Springer, 2014. 2
- [30] Kosta Ristovski, Vladan Radosavljevic, Slobodan Vucetic, and Zoran Obradovic. Continuous conditional random fields for efficient regression in large fully connected graphs. In *Twenty-Seventh AAAI Conference on Artificial Intelligence*, 2013. 4, 5
- [31] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. *arXiv preprint arXiv:1803.02155*, 2018. 3
- [32] Deqing Sun, Erik B Sudderth, and Hanspeter Pfister. Layered rgbd scene flow estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 548–556, 2015. 2
- [33] Lyne Tchapmi, Christopher Choy, Iro Armeni, JunYoung Gwak, and Silvio Savarese. Segcloud: Semantic segmentation of 3d point clouds. In *2017 international conference on 3D vision (3DV)*, pages 537–547. IEEE, 2017. 2
- [34] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6411–6420, 2019. 2
- [35] 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. 2
- [36] Levi Valgaerts, Andrés Bruhn, Henning Zimmer, Joachim Weickert, Carsten Stoll, and Christian Theobalt. Joint estimation of motion, structure and geometry from stereo sequences. In *European Conference on Computer Vision*, pages 568–581. Springer, 2010. 2
- [37] 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. 1, 2
- [38] Christoph Vogel, Konrad Schindler, and Stefan Roth. 3d scene flow estimation with a rigid motion prior. In *2011 International Conference on Computer Vision*, pages 1291–1298. IEEE, 2011. 2
- [39] Christoph Vogel, Konrad Schindler, and Stefan Roth. Piecewise rigid scene flow. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1377–1384, 2013. 2
- [40] Christoph Vogel, Konrad Schindler, and Stefan Roth. 3d scene flow estimation with a piecewise rigid scene model. *International Journal of Computer Vision*, 115(1):1–28, 2015. 2
- [41] Lei Wang, Yuchun Huang, Yaolin Hou, Shenman Zhang, and Jie Shan. Graph attention convolution for point cloud semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10296–10305, 2019. 2
- [42] 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. 2
- [43] Yue Wang and Justin M Solomon. Deep closest point: Learning representations for point cloud registration. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3523–3532, 2019. 5
- [44] Wenxuan Wu, Zhongang Qi, and Li Fuxin. Pointconv: Deep convolutional networks on 3d point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9621–9630, 2019. 2, 6
- [45] 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. 1, 2, 3, 6, 7
- [46] Dan Xu, Elisa Ricci, Wanli Ouyang, Xiaogang Wang, and Nicu Sebe. Multi-scale continuous crfs as sequential deep networks for monocular depth estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5354–5362, 2017. 2, 6
- [47] Bo Yang, Jianan Wang, Ronald Clark, Qingyong Hu, Sen Wang, Andrew Markham, and Niki Trigoni. Learning object bounding boxes for 3d instance segmentation on point clouds. In *Advances in Neural Information Processing Systems*, pages 6737–6746, 2019. 2
- [48] Chi Zhang, Guosheng Lin, Fayao Liu, Rui Yao, and Chunhua Shen. Canet: Class-agnostic segmentation networks with iterative refinement and attentive few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5217–5226, 2019. 2
- [49] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip HS Torr. Conditional random fields as recurrent neural networks. In *Proceedings of the IEEE international conference on computer vision*, pages 1529–1537, 2015. 6