GAMAFLOW: ESTIMATING 3D SCENE FLOW VIA GROUPED ATTENTION AND GLOBAL MOTION AGGREGATION

Zhiqi Li, Xiaosong Yang, Jianjun Zhang

National Centre for Computer Animation, Bournemouth University, UK

ABSTRACT

The estimation of 3D motion fields, known as scene flow estimation, is an essential task in autonomous driving and robotic navigation. Existing learning-based methods either predict scene flow through flow-embedding layers or rely on local search methods to establish soft correspondences. However, these methods often neglect distant points which, in fact, represent the true matching elements. To address this challenge, we introduce **GAMAFlow**, a point-voxel architecture that models local motion and global motion to predict scene flow iteratively. In particular, **GAMAFlow** integrates the advantages of (i) the point Transformer with **G**rouped **A**ttention and (ii) global **M**otion **A**ggregation to boost the efficacy of point-voxel correlation. Such an approach facilitates learning long-distance dependencies between current frame and next frame. Experiments illustrate the performance gains achieved by GAMAFlow compared to existing works on both FlyingThings3D and KITTI benchmarks.

Index Terms— Scene Flow Estimation, Attention Model, Point-Voxel Correlation, 3D Perception

1. INTRODUCTION

As an analog of optical flow [1], 3D scene flow has attracted increasing research attention in recent years [2]. It is a crucial primitive to various visual perception and understanding tasks like motion segmentation [3], object tracking [4], and trajectory prediction [5]. With point cloud data being widely applied in robotics and autonomous driving, many scene flow estimation methods [6, 7, 8, 9, 10] have been proposed to predict 3D displacements between two consecutive point cloud frames directly through deep neural networks.

Given the current state-of-the-art in scene flow estimation, we found that prediction errors primarily stemmed from occlusions and invisible long-distance agents, which poses great challenges in discriminating multi-scaled motion fields. From prior work, we have the following observations: (1) Voxel-based representations can efficiently encode multi-scale features of 3D point clouds, which are then used for object detection or segmentation. However, the downside of voxel-based representation is that it degrades localization quality due to the coarse voxelization. (2) Point-based representation could preserve accurate point positions with flexible receptive fields, which benefits flow estimation without heavy computation overhead. In light of these, the recent work PV-RAFT [9] integrates the voxel-based and point-based feature learning strategy. Meanwhile, SCTN [11] combines the point feature extracted through Transformers with voxel feature extracted via sparse 3D convolution. Despite yielding promising results, this integration of voxel-based and point-based feature representations poses two problems. The voxel-to-point encoding through voxel set abstraction operations introduces significant computational overhead, which is further exacerbated by the multi-stage point feature abstraction. On the other hand, the pooling operation in the voxel branch fails to fully harness the valuable dense points, resulting in little performance improvement for faraway or small objects with sparse points.

To handle these problems, we present a Transformer-based scene flow estimation paradigm with point-voxel correlations. First, we propose to augment point features via point Transformer layer. Specifically, we employ grouped vector attention to propagate point-wise features and guide the learning of discriminative patterns at voxel level. Secondly, we enhance the local motion feature through a context-based global motion aggregation (GMA) module. Extensive visualizations showcase the superiority of **GAMAFlow** on both datasets.

2. METHODS

Overview: The whole pipeline of our method is depicted in Fig. 1. It takes two consecutive point clouds $\mathcal{X} \in \mathbb{R}^{N\times 3}$ and $\mathcal{Y} \in \mathbb{R}^{M\times 3}$ as the input. **GAMAFlow** aims to predicting a set of flow vectors $\left\{v_i \in \mathbb{R}^3\right\}_{i=1}^{N}$ that describe the motion field, which means the source point cloud $\mathcal{X}$ is expected to move to $y_i = (x_i + v_i) \in \mathcal{Y}$. The model first proceeds by extracting point features F_X, F_Y from $\mathcal{X}$ and $\mathcal{Y}$ (Sec 2.1). Next, as introduced in Sec 2.2, a global motion aggregator is applied to enhance the motion features, followed by a Gated Recurrent Unit (GRU) to iteratively generate the flow vector. Finally, the flow vectors are progressively updated between the translated point cloud $\mathcal{Q}_t = \mathcal{X} + V_t$ and the target point cloud $\mathcal{Y}$. After T

Thanks to the Chinese Scholarship Council (CSC) for funding the first author under #202006830020. This work was supported in part by Baskerville: a national accelerated compute resource under the EPSRC Grant EP/T022221/1, and The Alan Turing Institute under EPSRC grant EP/N510129/1.

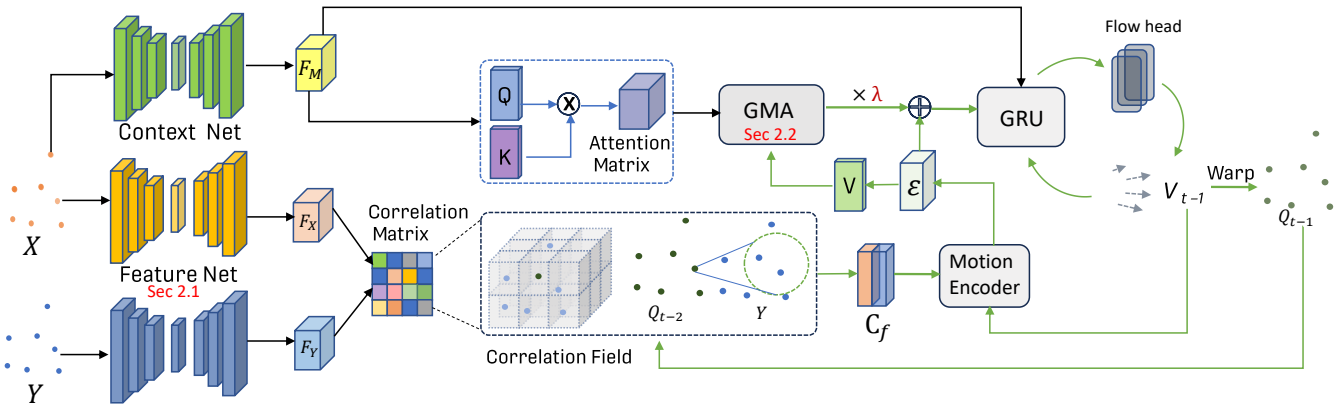


Fig. 1: The pipeline of **GAMAFlow**: The input of GAMAFlow consists of two point clouds with only 3D coordinate information. The correlation field is constructed from point-level features computed by feature net and voxel-level features. GMA is performed on the context feature F_M and the local motion feature $\mathcal{E}$. The concatenation of the local motion feature, context feature, and global motion feature from GMA is fed into GRU to update the hidden state iteratively. The output of GRU is passed through a flow head to produce the residual flow V_{t-1} and the translated point cloud Q_{t-1} for the next iteration.

iterations, the final scene flow is obtained by a flow refinement module. These steps are detailed below.

2.1. Point Feature Extraction via Grouped Attention

Previous attention-based approaches [11, 6] compute features directly on the whole point cloud, ignoring the existence of multi-scale motion fields in the scene. Considering this issue, we build our feature net and context net upon PointTransformerV2 [12] to learn point feature at multiple scales. The core purpose of the feature net is to produce 128-dimensional point-wise features for the input point clouds, denoted as $F_X \in \mathbb{R}^{N\times D}, F_Y \in \mathbb{R}^{M\times D}$ in Fig.1. Our feature net consists of point Transformer blocks employed at four different resolutions. Each block begins with a downsampling layer, followed by grouped attention for feature aggregation. Next, we decode the features and scale them up to match the original point set.

To do this, we use partition-based pooling to divide the point cloud into L subsets. For instance, an individual point set is denoted as $\mathcal{S}_i = (\mathcal{P}_i, \mathcal{F}_i)$, where $x_i = (p_i, f_i)$ belongs to $\mathcal{S}_i$. Feature f_i is updated via maxpooling operation and point position p_i is updated via meanpooling. The updated subset $\mathcal{S}' = (\boldsymbol{p}'_i, \boldsymbol{f}'_i)_{i=1}^{n'}$ are leveraged in the next stage. The channel dimension of each feature encoding layer follows: $3 \rightarrow 48 \rightarrow 96 \rightarrow 192 \rightarrow 384$. We then pass the fused point set $\mathcal{S}'$ through a decoding layer for feature propagation via inverse distance weighted averaging with the 3 nearest neighbors.

Grouped vector attention allows efficient learning of spatial features as well as context features across different regions of the point clouds, which is given by

$$\mathbf{A}_{ij} = \beta(\gamma(\mathbf{q}_i, \mathbf{k}_j)),\ \mathbf{f}_i^a = \sum_{\mathbf{x}_j}^{\mathcal{S}(\mathbf{p}_i)} \sum_{l=1}^{g} \sum_{m=1}^{c/g} Softmax\left(\mathbf{A}_i\right)_{jl} \mathbf{v}_j^{lc/g+m}, \tag{1}$$

where γ is a relation function and $\beta(\cdot)$ is an encoding function to produce grouped weight. The output feature $\mathbf{f}_i^a$ is aggregated by dividing the channels of the value vector into g groups, thereby reducing the number of parameters.

Context Net: We encode a context feature $\mathcal{F}_M \in \mathbb{R}^{N\times D}$ to enrich context information during the flow estimation process. The context net shares the same structure with the feature net.

2.2. Global Motion Aggregation Module

We notice that the flow estimation is significantly degraded or even fails when dealing with large motions, which frequently occurs in non-local regions. To mitigate this issue, we introduce an enhanced global motion aggregation module, which reduces the number of isolated points and the need for masking operation [16]. We posit that temporal coherence exists between two consecutive point clouds, which can be utilized to build long-distance correlations among two point clouds.

To this end, we encode motion feature $\mathcal{E} \in \mathbb{R}^{N\times D_m}$ using the previously estimated flow vector V_{t-1} and the correlation feature $\mathbf{C_f}$ generated by the point-voxel correlation field. Let θ, ϵ, σ denote the projection functions to calculate query, key, and value vector. The aggregated motion feature is denoted as

$$\hat{\mathcal{E}}_i = \mathcal{E}_i + \lambda \sum_{j=1}^{N} h\left(\theta(\mathcal{F}_M), \epsilon(\mathcal{F}_M)\right) \sigma\left(\mathcal{E}_j\right), \tag{2}$$

where λ is a hyperparameter initialized to zero. The attention matrix calculated by $h(\cdot)$ is utilized for aggregating the value vector that represents temporal coherence.

$$h\left(\mathbf{q}_i, \mathbf{k}_j\right) = \frac{\exp\left(\mathbf{q}_i^\top \mathbf{k}_j / \sqrt{D}\right)}{\sum_{j=1}^{N} \exp\left(\mathbf{q}_i^\top \mathbf{k}_j / \sqrt{D}\right)}. \tag{3}$$

Method	FlyingThings3D				KITTI			
	EPE3D ↓	**Acc3DS** ↑	**Acc3DR** ↑	**Outliers** ↓	**EPE3D** ↓	**Acc3DS** ↑	**Acc3DR** ↑	**Outliers** ↓
PointPWC-Net [13]	0.059	0.738	0.928	0.342	0.069	0.728	0.888	0.265
FLOT [14]	0.052	0.732	0.927	0.357	0.056	0.755	0.908	0.242
FlowStep3D [8]	0.046	0.816	0.961	0.217	0.055	0.805	0.925	0.149
SCTN [11]	0.038	0.847	0.968	0.268	0.037	0.873	0.959	0.179
PV-RAFT [9]	0.046	0.817	0.957	0.292	0.056	0.823	0.937	0.216
PT-FlowNet [15]	0.031	0.914	0.981	0.175	0.023	0.958	0.979	0.121
Ours	**0.027**	**0.935**	**0.986**	**0.162**	**0.022**	**0.963**	**0.983**	0.122

Table 1: Quantitative evaluation on Flyingthings3D and KITTI datasets. Lower values are better for the error metrics including EPE3D and Outliers. Higher values are better for the accuracy metrics including Acc3DS and Acc3DR.

The final output is the concatenation $[\mathcal{E}, \hat{\mathcal{E}}]$. Intuitively, concatenation allows the network to flexibly merge motion vectors influenced by contextual attributes, possibly introducing an element of uncertainty in its encoding process before decoding the combined motion vector.

2.3. Network Architecture

We adopt PV-RAFT [9] as our backbone, equip it with our proposed module. After feature extraction via Grouped Attention, we constructs two correlation volumes based on feature similarities. The correlation volume integrates features at **point-level** and **voxel-level**. Let $\mathcal{N}_k = \mathcal{N}(\mathcal{Y})_k$ represents the top-k nearest neighbors of $\mathcal{Q}_t$ in $\mathcal{Y}$. **Point-level** correlation feature between $\mathcal{Q}_t$ and $\mathcal{Y}$ is defined as

$$\mathbf{C}_p\left(\mathcal{Q}_t, \mathcal{Y}\right) = \gamma\left(\mathrm{MLP}(\mathrm{concat}(\mathbf{C}_M(\mathcal{N}_k), \mathcal{N}_k - \mathcal{Q}_t))\right). \quad (4)$$

Voxel-level correlation feature is defined as

$$\mathbf{C}_v\left(\mathcal{Q}_t, \mathcal{Y}\right) = \mathrm{MLP}\left(\underset{\mathbf{i}}{\mathrm{concat}}\left(\frac{1}{n_\mathbf{i}}\sum_{n_\mathbf{i}}\mathbf{C}_M\left(\mathcal{N}_r^{(\mathbf{i})}\right)\right)\right), \quad (5)$$

where n_i denotes the number of points in $\mathcal{Y}$ that located in a sub-cube of $\mathcal{Q}_t$ and $\mathcal{N}_r^{(\mathbf{i})}$ indexes all neighbor points of a sub-cube in $\mathcal{Q}_t$. In formulation 4 and 5, $\mathbf{C}_M(\mathcal{N}_k)$ represents the corresponding truncated correlation values, which is computed through the pairwise dot-product between feature vectors $\mathbf{C}_M = \mathbf{F}_q^t \cdot \mathbf{F}_y$. The combination of $\mathbf{C}_v$ and $\mathbf{C}_p$ is denoted as $\mathbf{C_f}$. In the current paradigm, a correlation volume serves as the fundamental module for consecutive frame point matching. Conceptually, the correlation volume at the point-level focus on local regions while correlation volume at voxel-level compensates for large displacements.

Iterative Update: With the integration of voxel features and point features, we follow PV-RAFT [9] to update scene flow estimation based on a GRU cell. The input to the GRU cell consists of three components: the contextual feature $\mathcal{F}_M$, global motion features $[\hat{\mathcal{E}}, \mathcal{E}]$, and previously predicted scene flow vector V_{t-2}. Then, we use the updated flow V_{t-1} to warp a new translated point cloud $\mathcal{Q}_{t-1}$ for the next iteration.

Refinement Step: In pursuit of enhanced performance in scene flow estimation networks [15], a subsequent refinement step is implemented to produce the final refined flow prediction V_{ref}. Unlike the pre-training stage, the refinement module only utilizes the final predicted flow vector from iterative update stage and the point feature F_Y. This refinement step promotes the smoothness and consistency of flow estimation.

2.4. Loss Terms

We trained our model in a supervised manner, where flow vectors are updated iteratively. The loss is formed as:

$$\mathcal{L}_{iter} = \sum_{t=1}^{T} w_t \left\|\left(V_t - V_{gt}\right)\right\|_1, \quad (6)$$

where V_t is the predicted flow vector from the t^{th} iteration in the first updating stage. V_{gt} denotes the ground-truth flow vector. T is the total number of iterations and the weight for t^{th} iteration is w_t. The loss of the refinement module is

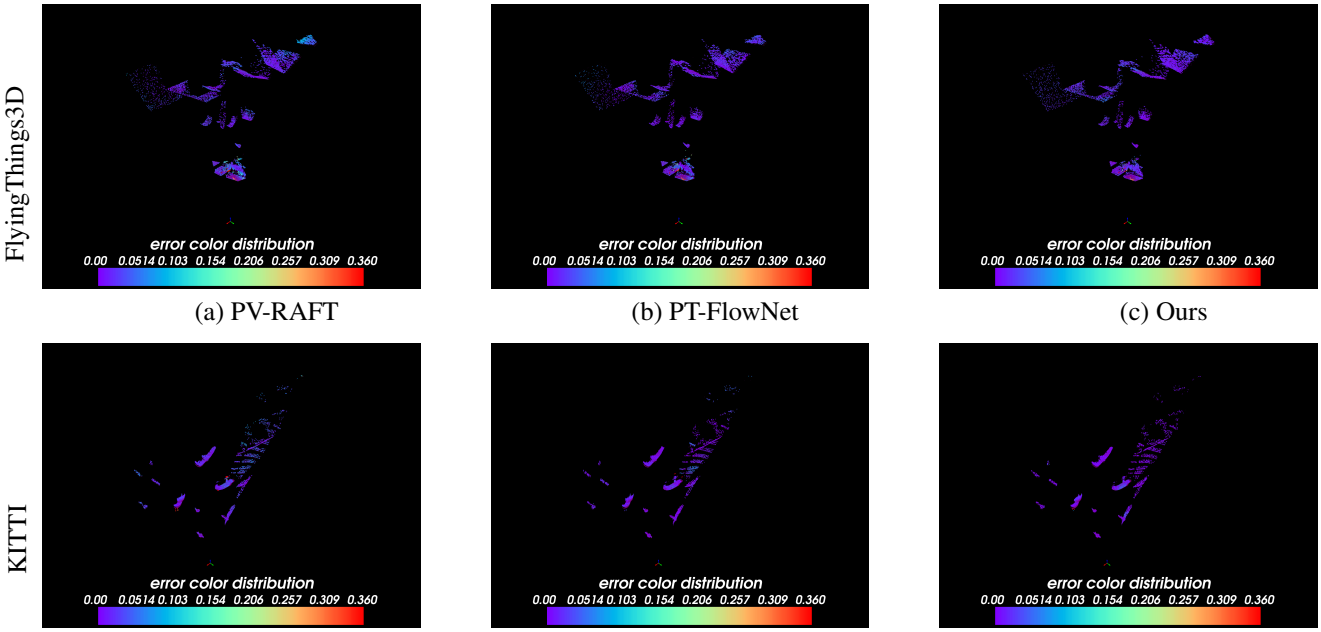
$$\mathcal{L}_{ref} = \left\|\left(\hat{V}_{ref} - V_{gt}\right)\right\|_1. \quad (7)$$

3. EXPERIMENTS

3.1. Datasets and Performance Metrics

FlyingThings3D [17] collects rendered stereo and RGB-D images from ShapeNet [18], which is the first synthetic benchmark to estimate scene flow. We follow [9] to pre-process and separate FlyingThings3D into a training set (19, 640 pairs) and a test set (3, 824 pairs). To evaluate the effectiveness of our model in a real dataset, we choose KITTI scene flow dataset [19, 20] and leverage the trained model on FlyingThings3D.

Implementation details. We implemented GAMAFlow in Pytorch. We set the number N, M of the input point clouds to 8192. The while network is first trained for 50 epochs, with another 10 epochs for the refinement step. Experiments were conducted on a machine equipped with four NVIDIA A100-SXM4-80GB GPUs.



(a) PV-RAFT (b) PT-FlowNet (c) Ours

Fig. 2: Visual comparison between PV-RAFT [9], PT-FlowNet [15], and our method on FlyingThings3D and KITTI dataset.

Evaluation Metrics. We use several metrics for comprehensive comparison. 3D end-point-error (EPE3D) is the mean L2 distance between the ground truth scene flow and predicted result. Strict accuracy (Acc3DS) is the percentage of points whose EPE3D < 0.05m or relative error $< 5\%$. Relaxed accuracy (Acc3DR) is the percentage of points whose EPE3D < 0.1m or relative error $< 10\%$. Outliers is the percentage of points whose EPE3D > 0.3m or relative error $> 10\%$.

3.2. Quantitative Analysis

Table.1 demonstrates that our model, which was trained using a synthetic dataset, exhibits strong generalization capabilities when applied to real KITTI scans. Specifically, **GAMAFlow** reduces EPE3D to 0.027m, achieved a 12% drop from PT-FlowNet [15] on FlyingThings3D. Our method also presents superior performance on KITTI in terms of Acc3DS and Acc3DR. Moreover, in Fig. 2, we visually compare the scene flow estimation results for scenes from both datasets. Colors indicate the EPE3D error distribution, with red means high error and purple means low error. It is noticeable that our method shows the minimum error.

3.3. Ablation Study

The effectiveness of key components. We conduct ablation experiments to verify the rationality of our method. Variant I is trained on PT-FlowNet [15] without the refinement step. Then we replace the core components with grouped attention for feature extraction (II) and global motion aggregation (III). Thirdly, we add the flow refinement module in our model, forming the last variant (IV). As shown in Table.2, the grouped attention module brings an improvement of 13.5% and variant III brings an improvement of 21.6% compared to variant I.

ID	GA	GMA	FR	EPE3D
I				0.037
II	✓			0.032
III	✓	✓		0.029
IV	✓	✓	✓	0.027

Table 2: Ablation study results on grouped attention module and global motion aggregation module. These experiments are conducted on FlyingThings3D.

4. DISCUSSIONS AND CONCLUSION

In this paper, we propose **GAMAFlow** for motion analysis between two point clouds. The core insight of our method is the integration of local motion feature and long-distance global information. Experimental results of **GAMAFlow** on the FlyingThings3D and KITTI datasets demonstrate its effectiveness. A limitation of our model is its relatively large parameter size compared to other attention-based methods, even though **GAMAFlow** brings a 41% drop on the EPE3D metric. This suggests that, utilizing lightweight model in the learning of visual representation of point could data could ease the computational bottleneck, which is an important avenue for future research. We will consider improving the overall efficiency through model compression techniques in the future.

5. REFERENCES

[1] Haofei Xu, Jing Zhang, Jianfei Cai, Hamid Rezatofighi, and Dacheng Tao, "Gmflow: Learning optical flow via global matching," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 8121–8130.

[2] Zhiqi Li, Nan Xiang, Honghua Chen, Jianjun Zhang, and Xiaosong Yang, "Deep learning for scene flow estimation on point clouds: A survey and prospective trends," in *Computer Graphics Forum*. Wiley Online Library, 2023.

[3] Stefan Andreas Baur, David Josef Emmerichs, Frank Moosmann, Peter Pinggera, Björn Ommer, and Andreas Geiger, "Slim: Self-supervised lidar scene flow and motion segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 13126–13136.

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

[5] Chaoyang Wang, Xueqian Li, Jhony Kaesemodel Pontes, and Simon Lucey, "Neural prior for trajectory estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 6532–6542.

[6] Yukang Shi and Kaisheng Ma, "Safit: Segmentation-aware scene flow with improved transformer," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 10648–10655.

[7] Xingyu Liu, Charles R Qi, and Leonidas J Guibas, "Flownet3d: Learning scene flow in 3d point clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, California, USA, 2019, pp. 529–537, IEEE.

[8] Yair Kittenplon, Yonina C Eldar, and Dan Raviv, "Flowstep3d: Model unrolling for self-supervised scene flow estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 4114–4123.

[9] Y. Wei, Z. Wang, Y. Rao, J. Lu, and J. Zhou, "Pv-raft: Point-voxel correlation fields for scene flow estimation of point clouds," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Los Alamitos, CA, USA, jun 2021, pp. 6950–6959, IEEE Computer Society.

[10] Hanlin Li, Guanting Dong, Yueyi Zhang, Xiaoyan Sun, and Zhiwei Xiong, "Rppformer-flow: Relative position guided point transformer for scene flow estimation," in *Proceedings of the 30th ACM International Conference on Multimedia*, New York, NY, USaA, 2022, MM '22, p. 4867–4876, Association for Computing Machinery.

[11] Bing Li, Cheng Zheng, Silvio Giancola, and Bernard Ghanem, "Sctn: Sparse convolution-transformer network for scene flow estimation," vol. 36, no. 2, pp. 1254–1262, 2022.

[12] Xiaoyang Wu, Yixing Lao, Li Jiang, Xihui Liu, and Hengshuang Zhao, "Point transformer v2: Grouped vector attention and partition-based pooling," in *NeurIPS*, 2022.

[13] 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*. Springer, 2020, pp. 88–107.

[14] Gilles Puy, Alexandre Boulch, and Renaud Marlet, *FLOT: Scene Flow on Point Clouds Guided by Optimal Transport*, pp. 527–544, 2020.

[15] Jingyun Fu, Zhiyu Xiang, Chengyu Qiao, and Tingming Bai, "Pt-flownet: Scene flow estimation on point clouds with point transformer," *IEEE Robotics and Automation Letters*, vol. 8, no. 5, pp. 2566–2573, 2023.

[16] Ziyang Song and Bo Yang, "Ogc: Unsupervised 3d object segmentation from rigid dynamics of point clouds," *Advances in Neural Information Processing Systems*, vol. 35, pp. 30798–30812, 2022.

[17] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox, "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4040–4048.

[18] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al., "Shapenet: An information-rich 3d model repository," *arXiv preprint arXiv:1512.03012*, 2015.

[19] Moritz Menze, Christian Heipke, and Andreas Geiger, "Joint 3d estimation of vehicles and scene flow," *ISPRS annals of the photogrammetry, remote sensing and spatial information sciences*, vol. 2, pp. 427, 2015.

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