Supplementary Material: PointPWC-Net: Cost Volume on Point Clouds for (Self-)Supervised Scene Flow Estimation

Anonymous ECCV submission

Paper ID 2950

In the supplementary material, we provide more details on the network structures of the feature pyramid network and the scene flow predictor in Sec.1. In Sec.2, we conduct additional ablation experiments. Besides, we also provide more visualization of scene flow results on the KITTI Scene Flow 2015 [3, 2] dataset in Sec.3. Finally, we analyze the typical error types of PointPWC-Net on KITTI in Sec. 4.

1 Network Details

Fig.1 shows the architecture for feature pyramid network. Fig.2 shows the scene flow predictor network. At the most coarse level, the scene flow predictor only takes the feature from the first point cloud and the cost volume as input. At the rest of the level, the scene flow predictor takes the feature from the first point cloud, the cost volume, the upsampled flow from previous layer, and the upsampled feature of the second last layer from previous level's scene flow predictor as input.

2 Additional Experiments

Concatenated coarser features. In feature pyramid network, we concatenate the feature on level l with the upsampled feature from level $l+1$, which gives a larger receptive field. In scene flow predictor, we take the feature from the second last layer of scene flow predictor from level $l+1$ as an input to the scene flow predictor in level l. In Table 1, we show the effectiveness of these features. Both features can improve our scene flow results significantly.

Runtime breakdown. In order to study the runtime of each critical part of our model, we include the runtime breakdown in Table. 2.

3 More Visual Results on KITTI

Fig.3 provide more visualization on KITTI dataset of scene flow results by PointPWC-Net. In Fig.3, the model is trained with self-supervised loss on FlyingThings3D [1], and directly tested on KITTI Scene Flow 2015 [3, 2] without any finetune. From Fig.3, we can see that our model can recover scene flow not only for the rigid objects, such as cars, but also for the non-rigid objects, such as bushes, trees, etc.

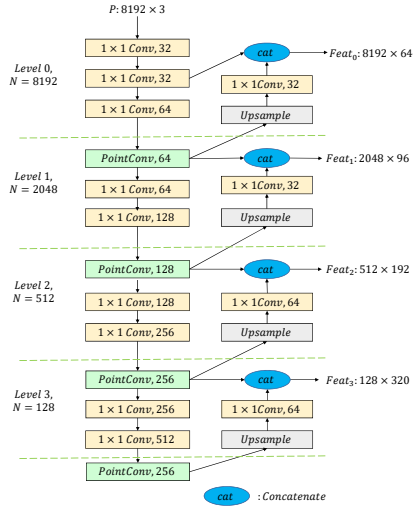

Fig. 1. Feature Pyramid Network. The first point cloud and the second point cloud are encoded using the same network with shared weights. For each point cloud, we use PointConv [4] to convolve and downsample by factor of 4. The $1 \times 1 Conv$s are used to increase the representation power and efficiency. The final feature of level l is concatenated with the upsampled feature from level $l+1$, which contains feature with a larger receptive field.

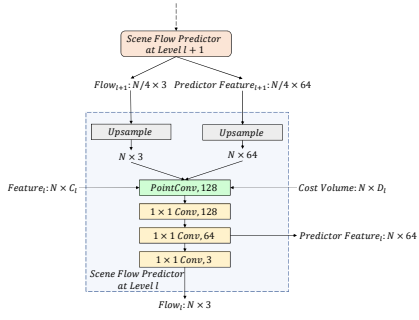

Fig. 2. Scene Flow Predictor. The scene flow predictor takes the feature from the first point cloud, the cost volume, the upsampled flow from previous layer, and the upsampled feature of the second last layer from previous level's scene flow predictor as input. The output is the estimated flow in current level and the feature in the second last layer.

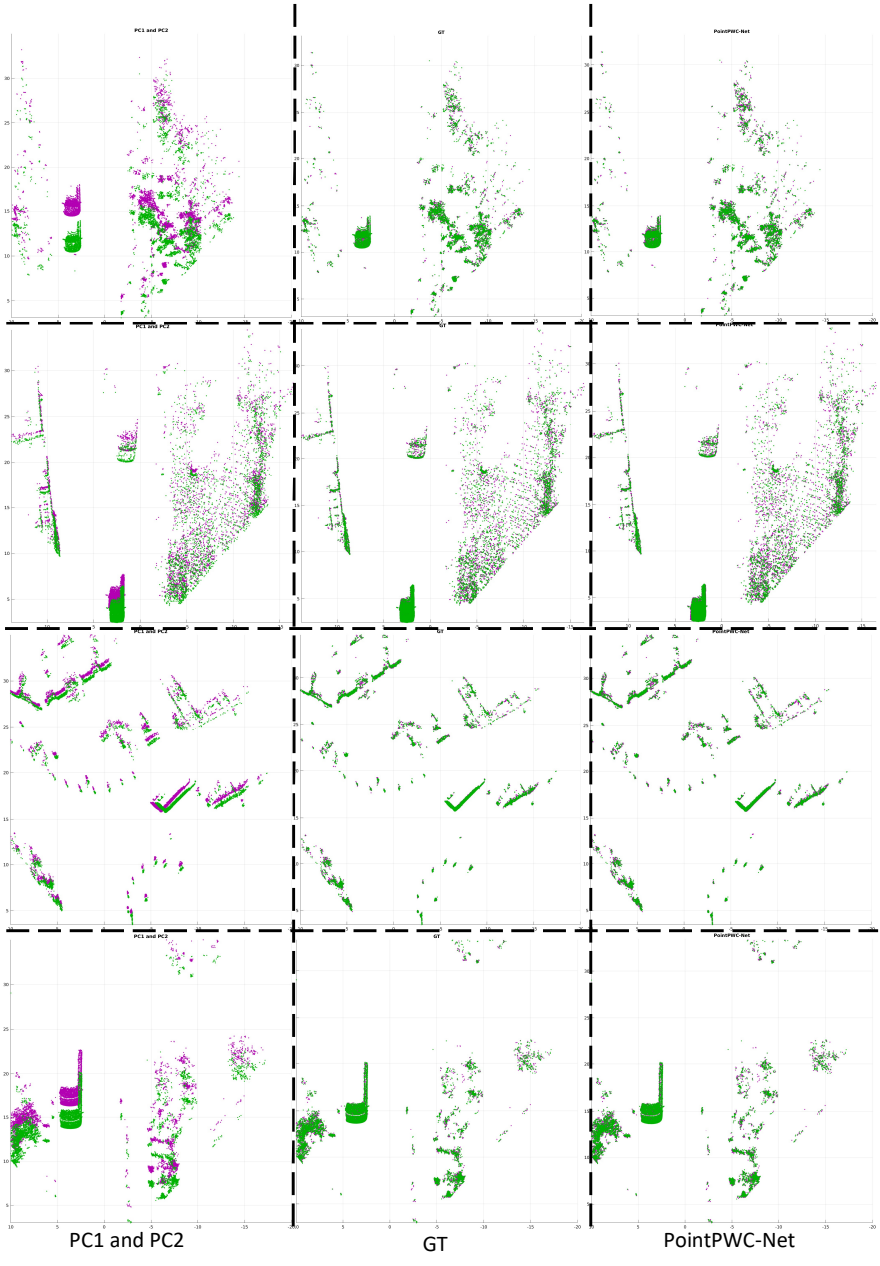

Fig. 3. Scene Flow Results on KITTI Scene Flow 2015. In (a), 2 point clouds PC1 and PC2 are presented in Magenta and Green, respectively. In (b) and (c), PC1 is warped to PC2 based on the (computed) scene flow. (b) shows the ground truth; (c) Results from PointPWC-Net that is trained with supervision on FlyingThings3D, and directly evaluate on KITTI without any fine-tune. Enlarge images for better view. (Best viewed in color)

Table 1. Concatenated coarser features. The upsampled feature at level l is the feature upsampled from the level $l+1$ followed by a $1 \times 1 Conv$. The predictor feature for level l is upsampled from the second last layer of the scene flow predictor in level $l+1$. Adding just the upsampled feature improves the performance by 9.5%, and then with both the performance improves by 34.8%

Upsampled Feature	Predictor Feature	EPE3D↓
-	-	0.0902
✓	-	0.0816
✓	✓	**0.0588**

Table 2. Runtime breakdown. The runtime of each critical component in PointPWC-Net

Component	Feature Pyramid	Cost Volume	Upsample Warping	Scene Flow Predictor
Runtime(ms)	43.2	22.7	24.5	27.0

4 Typical Error Types

We summaries three typical error types of our PointPWC-Net for KITTI dataset. To visualize errors, we use blue, green and red to represent the first point cloud, the warped points which are correctly predicted, and the wrongly predicted points, respectively, as shown in Fig. 4. The first error type is when the object is a straight line or a plane. In this case, it is hard for the network to construct a cost volume with strong discernment, as shown in Fig.4 **A** and **C**. The second one is that it is hard to find good correspondences between consecutive frames due to the strong deformation of local shapes, as shown in Fig.4 **B**. The third case is that the ground points are not removed properly, as shown in Fig.4 **D**. By using a better ground removal strategy, we can further improve our results.

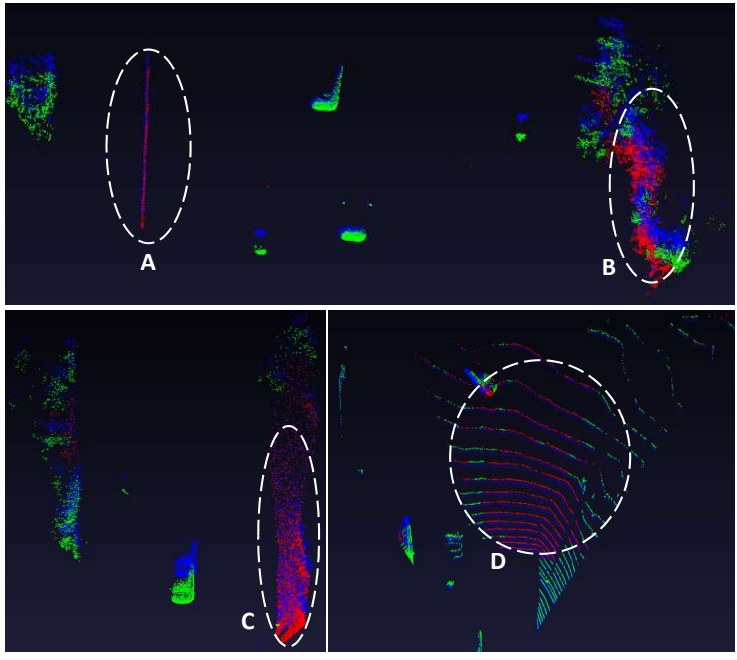

Fig. 4. Typical error types for KITTI. The blue points are from the first point cloud P. The green points are the warped points $P_w = P+SF$ according to the correctly predicted flow. The "correctness" is measured by Acc3DR. The red points are wrongly predicted. **A** and **C** are the ambiguity in 3D point clouds, which are straight lines or plane walls. **B** is the messy bushes, whose features do not have strong correspondences. **D** is the case when the ground points are not removed cleanly.

References

1. Mayer, N., Ilg, E., Hausser, P., Fischer, P., Cremers, D., Dosovitskiy, A., Brox, T.: 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. pp. 4040–4048 (2016)
2. Menze, M., Heipke, C., Geiger, A.: Joint 3d estimation of vehicles and scene flow. In: ISPRS Workshop on Image Sequence Analysis (ISA) (2015)
3. Menze, M., Heipke, C., Geiger, A.: Object scene flow. ISPRS Journal of Photogrammetry and Remote Sensing (JPRS) (2018)
4. Wu, W., Qi, Z., Fuxin, L.: Pointconv: Deep convolutional networks on 3d point clouds. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 9621–9630 (2019)