3D Understanding of Deformable Linear Objects: Datasets and Transferability Benchmark

Bare Luka Zagar[1*], Mingyu Liu[1*], Tim Hertel[1*], Ekim Yurtsever[2], Alois Knoll[1]

[1]Technical University of Munich, 85748 Garching b. München, Germany

[2] The Ohio State University, Columbus, OH 43212, USA

bare.luka.zagar@tum.de

[*]Authors contributed equally.

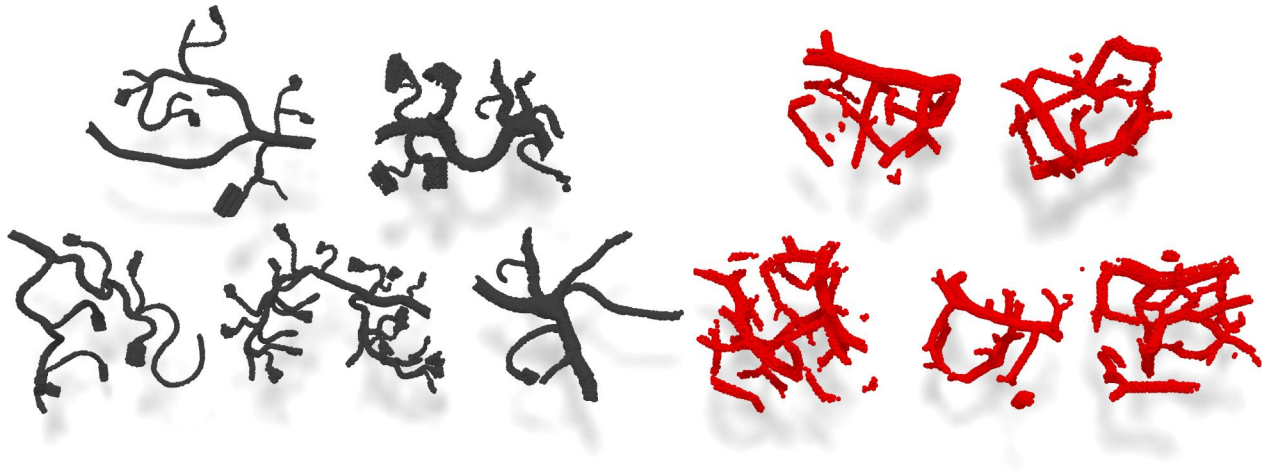

Figure 1. Examples of point clouds of complex deformable linear objects. **Left:** Wiring harness point cloud samples. **Right:** Blood vessel point cloud samples.

1. PointWire Dataset[1]

The PointWire dataset samples with annotations are given in the attached video - *PointWire.mov*. Additionally, we show several PointVessel samples in Fig. 1.

1.1. Skeleton Comparison

We compared four different 3D skeleton algorithms: ROSA[2] [10], SkelTre[3] [1], L1-Medial[4] [5], Laplacian-based contraction[5] [2] qualitatively, as shown in Fig. 2. It is evident that the Laplace-based contraction methods [2] gives visually the most accurate results.

1.2. Blender Animation

The animated wiring harnesses in Blender are provided in the attached video - *PointWire_Blender.mov*.

2. PointVessel Dataset[6]

The original synthetic blood vessel dataset introduced by [11] is generated by using the vascular tree generation methods from [8, 9]. Examples of our PointVessel point cloud dataset are given in Fig. 1.

3. Experimental Setting

In this study, we construct benchmarks for our proposed PointWire and PointVessel datasets with the following six

[1]The dataset used for the segmentation experiments is attached in the zip file *PointWire_Part.zip*.

[2]https://github.com/taiya/rosa

[3]https://github.com/MarcSchotman/skeletons-from-poincloud

[4]https://github.com/facontidavide/PointCloudProcessing

[5]https://github.com/meyerls/PC-Skeletor

[6]The dataset used for the segmentation experiments is attached in the zip file *PointVessel_Part.zip*

Method	bs	lr	epoch	opt	sched
PointNet++ [6]	8	1e-3	200	Adam	step
DGCNN [12]	8	2.5e-2	200	SGD	cos
PCT [4]	8	2.5e-3	200	SGD	cos
CurveNet [14]	8	1.25e-2	200	SGD	cos
DeltaConv [13]	8	5e-2	200	SGD	cos
RepSurf [7]	8	5e-2	200	SGD	mstep

Table 1. Hyperparameters of benchmarks. bs is batch size, lr is learning rate, opt is the optimizer that used in each method, and sched is the scheduler. For scheduler, step is StepLR, msetp is MultiStepLR and cos is CosineAnnealingLR.

Augmentation	Rotation	Perturbation	Jitter	Shift	Random scaling
Parameter Range	$-180° \sim 180°$	sigma=0.06	clip=0.01 clip=0.18	range=0.1	low=0.8 high=1.25

Table 2. Augmentation settings for robustness evaluation. We utilize the same data augmentation setting for both PointWire and PointVessel.

sota approaches, PointNet++[7] [6], DGCNN[8] [12], PCT[9] [4], CurveNet[10] [14], DeltaConv[11] [13] and RepSurf[12] [7]. All experiments were conducted using one NVIDIA GeForce RTX 3090 (24Gb) with CPU AMD Ryzen Threadripper 2950X 16-Core Processor. The hyperparameters of each model are shown in Tab. 1. To establish fair benchmarks, we set the batch size for all experiments to eight, and the learning rates of each network were modified based on the Linear Scaling Rule [3]. All models were trained for 200 epochs and we evaluated the best checkpoint, which had the highest mean accuracy on the validation set, on the test set. Furthermore, the optimizer of PointNet++ [6] was Adam with the StepLR as the scheduler. All the other models [12] [4] [14] [13] [7] were trained with Stochastic Gradient Descent (SGD) using the CosineAnnealingLR scheduler, except RepSurf, whose scheduler was StepLR.

4. Additional Quantitative Results

Scaling-up Training Data. To investigate the impact of training data size on the model performance, we provide the experimental results with partial training data, such as 5%, 10%, and 100%, in Tab. 3. As our expectation, the performance of the model increases with more training data, while DGCNN [12] exhibits stronger capability on learning from smaller data in comparison with PCT [4].

Robustness Analysis. We investigate the robustness of existing models trained on our datasets by applying various augmentations during the evaluation process. Specifically, we assess the performance of DGCNN [12] and PCT [4], and the specific values of each augmentation are present in Tab. 2. We present the experimental results in Tab. 4. In general, the application of a broader range of augmentations negatively impacts the performance of both models. However, PCT exhibits greater robustness to these augmentations, particularly in terms of bifurcation, than DGCNN.

Model	Training Data Size	PointVessel mIoU↑	mAcc↑	OA↑	v IoU↑	b IoU↑
DGCNN [12]	5%	69.55	84.07	83.13	76.49	62.60
	10%	73.47	86.67	85.71	79.85	67.06
	100%	77.44	89.15	88.21	83.19	71.70
PCT [4]	5%	25.95	54.49	42.19	18.36	33.36
	10%	33.62	59.07	50.39	31.75	35.50
	100%	57.38	76.70	73.62	63.46	51.31

Table 3. Quantitative results with various training data sizes. We evaluate the performance of DGCNN [12] and PCT [4] by training the model with increased training data size from 1% to 100% using the PointVessel dataset. The models are evaluated on the full 100% PointVessel test set to assess the impact of training data size on model performance.

Fine-tuning Results. In Tab. 5, we illustrate the results of DGCNN [12] and PCT [4]. Specifically, we initialize the models using pretrained weights and fine-tune them on the PointVessel dataset with an initial learning rate of 2.5e-4, over 50 training epochs. All other hyperparameters remain consistent with those outlined in Tab. 1. In general, fine-tuning leads to performance improvements across all models, especially for the bifurcation category. However, there remains a performance gap between fine-tuning and training the models directly on the PointVessel dataset.

5. Additional Qualitative Results

The qualitative results of samples 034/0199 and 038/0000 from PointWire are shown in Fig. 3. Furthermore, we show qualitative results of PointVessel (sample 0125/0077 and 0130/0033) in Fig. 4.

[7] https://github.com/yanx27/Pointnet_Pointnet2_pytorch
[8] https://github.com/AnTao97/dgcnn.pytorch
[9] https://github.com/MenghaoGuo/PCT
[10] https://github.com/tiangexiang/CurveNet
[11] https://github.com/rubenwiersma/deltaconv
[12] https://github.com/hancyran/RepSurf

	Augmentations	PointWire$_{bifurcation}$					PointVessel				
		mIoU↑	**mAcc↑**	**OA↑**	**w.IoU/ACC**	**b.IoU/ACC**	**mIoU↑**	**mAcc↑**	**OA↑**	**v.IoU/ACC↑**	**b.IoU/ACC↑**
(1)	none	49.28	78.71	72.87	69.48/70.85	29.08/86.57	77.44	89.15	88.21	83.19/86.45	71.70/91.86
		45.78	60.86	75.23	73.84/80.21	17.72/41.50	57.38	76.7	73.62	63.46/67.89	51.31/85.51
(2)	(1) + rotation	49.14	78.68	72.72	69.30/70.66	29.00/86.71	77.40	89.13	88.18	83.14 /86.40	71.66/91.86
		45.63	60.84	75.00	73.59/79.90	17.68/41.78	57.36	76.65	73.6	63.45/67.93	51.27/85.36
(3)	(2) + perturbation	48.38	78.39	71.84	68.29/69.58	28.47/87.21	77.39	89.12	88.17	83.14/86.41	71.65/91.84
		45.54	60.68	74.96	73.55/79.90	17.54/41.45	57.24	76.59	73.49	63.28/67.73	51.20/85.45
(4)	(3) + jitter	47.89	78.11	71.31	67.69/68.96	28.10/87.26	75.15	88.11	86.71	81.01/84.10	69.28/92.11
		45.51	61.12	74.64	73.16/79.31	17.87/42.94	55.59	75.88	71.99	60.95/64.81	50.23/86.87
(5)	(4) + shift	47.47	77.92	70.83	67.14/68.38	27.81/87.47	75.08	88.06	86.67	80.96/84.06	69.19/92.05
		45.45	61.76	74.18	72.59/78.47	18.31/45.05	55.41	75.66	71.85	60.80/64.74	50.02/86.58
(6)	(5) + random scale	47.54	77.98	87.51	67.21/68.44	27.86/87.51	74.8	87.91	86.48	80.71/83.81	68.90/92.03
		45.34	61.89	73.93	72.30/78.09	18.38/45.68	54.45	75.00	71.01	59.67/63.58	49.24/86.41

Table 4. Robustness evaluation results. To assess the robustness of the pretrained DGCNN [12] and PCT [4] models, we employ a range of augmentations during the evaluation process.

Method	**Fine-tuning**	PointWire$_{all}$ → PointVessel					PointWire$_{b}$ → PointVessel				
		mIoU↑	**mAcc↑**	**OA↑**	**v Iou/Acc↑**	**b IoU/Acc↑**	**mIoU↑**	**mAcc↑**	**OA↑**	**v IoU/Acc↑**	**b IoU/Acc↑**
DGCNN [12]	wo	52.02	66.70	72.21	66.70/82.46	37.35/50.94	45.94	66.08	63.67	52.36/59.17	39.51/73.00
	w	72.1	85.58	84.88	78.85/83.56	65.34/87.60	72.05	85.32	84.91	78.80/84.14	65.10/86.51
PCT [4]	wo	40.34	55.02	62.90	58.51/77.53	22.17/32.51	38.14	54.95	57.36	49.46/61.83	26.82/48.06
	w	49.88	70.41	67.15	55.64/61.07	44.13/79.75	44.44	66.49	61.81	48.39/53.08	40.51/79.89

Table 5. Fine-tuning results of DGCNN [12] and PCT [4]. Fine-tuning the pretrained models on the PointVessel dataset yields a substantial performance improvement compared to directly evaluating the models without additional training.

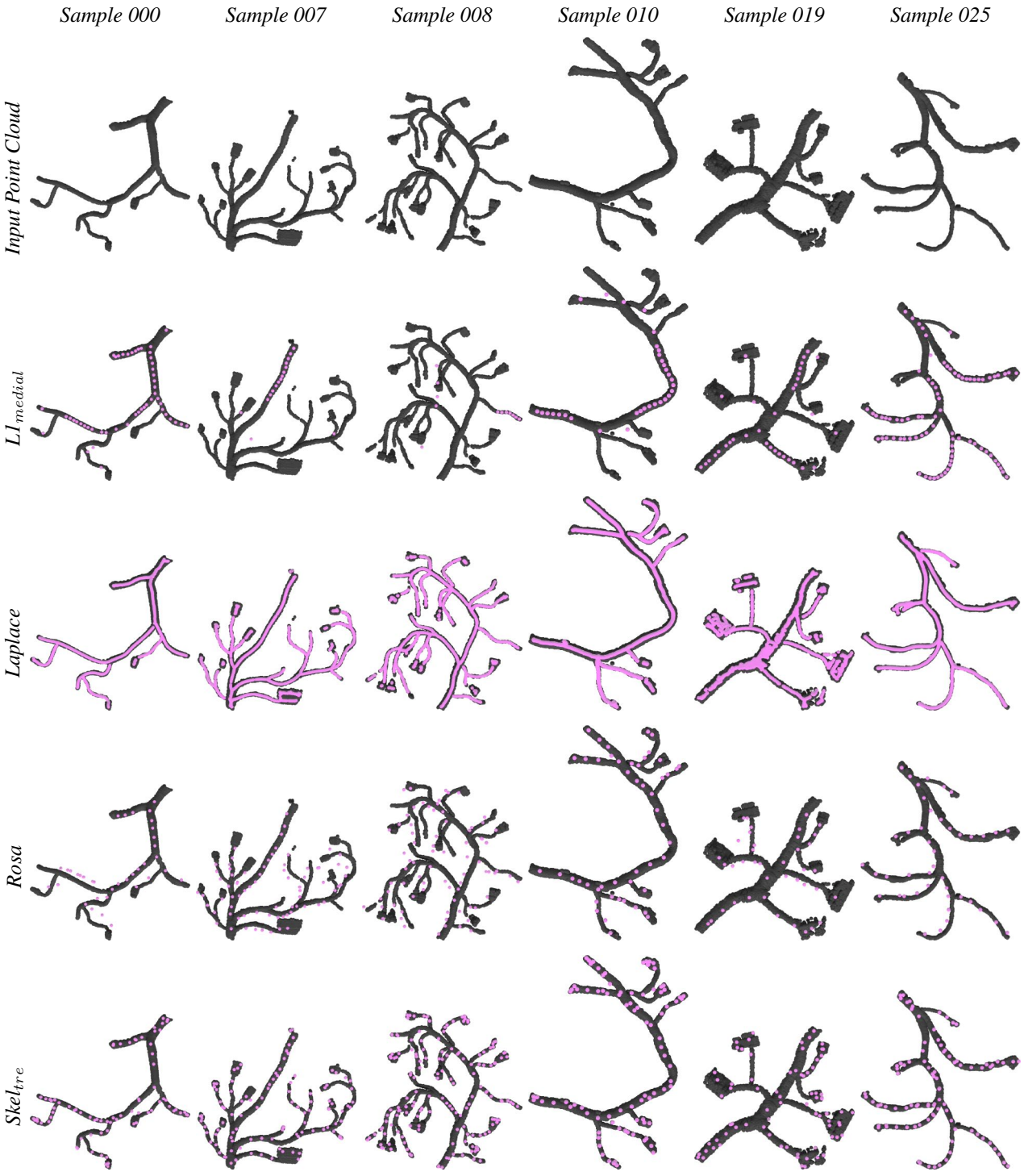

Figure 2. Qualitative comparison of the 3D skeleton algorithms. $L1_{medial}$: L1 medial skeleton method, *Laplace*: Laplacian-based contraction method, *Rosa*: Rotational symmetry axis method, $Skel_tre$: Skeletonisation of trees method.

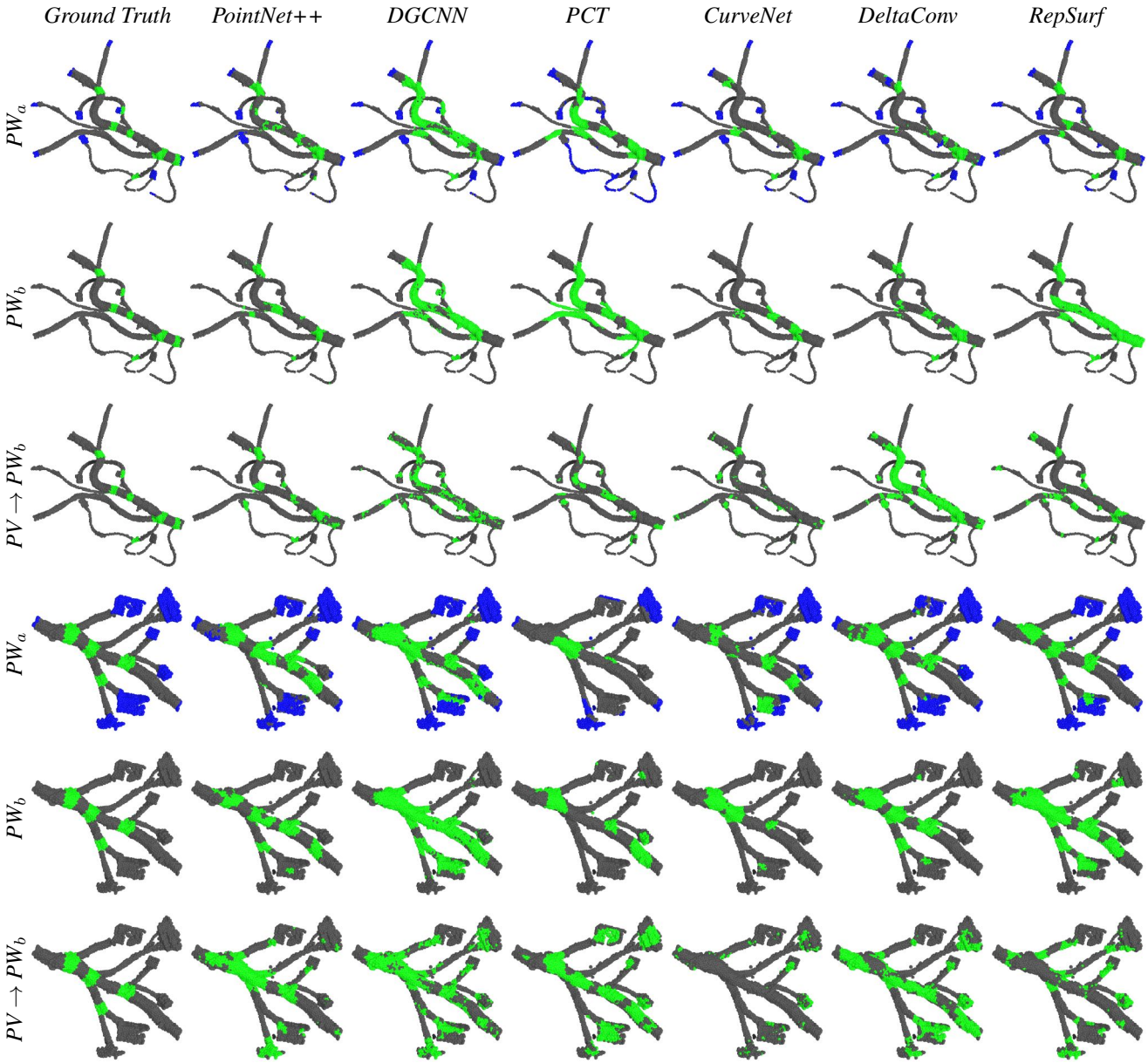

Figure 3. Qualitative results on PointWire all classes (PW_a), PointWire bifurcation (PW_b) and the transferability benchmark on the PointWire bifurcation ($PV \rightarrow PW_b$). For PW_a and PW_b, gray points are the wires, endpoints are blue, and green ones are bifurcations. The first three rows show the results from sample 034/0199, and the last three are from sample 038/0000.

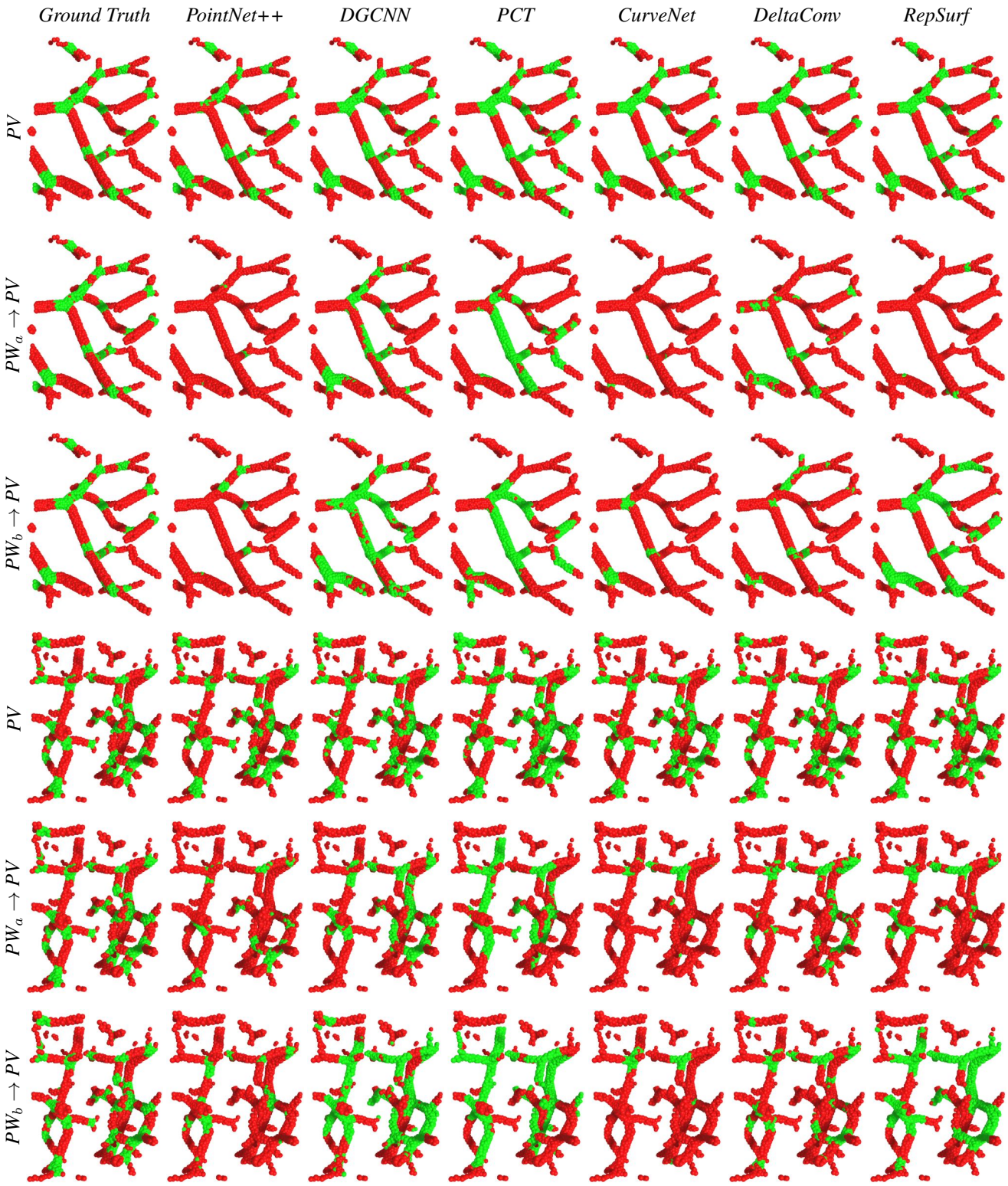

Figure 4. Qualitative results on PointVessel and transferability benchmark on PointVessel. PointWire all classes to PointVessel ($PW_a \rightarrow PV$), PointWire bifurcation to PointVessel ($PW_b \rightarrow PV$). We show vessels and bifurcations in red and green. The results from 0125/0077 and 0130/0033 are shown in the first three and last three rows, respectively.

References

[1] Alexander Bucksch, Roderik Lindenbergh, and Massimo Menenti. Skeltre: Robust skeleton extraction from imperfect point clouds. *The Visual Computer*, 26:1283–1300, 2010.

[2] Junjie Cao, Andrea Tagliasacchi, Matt Olson, Hao Zhang, and Zhixun Su. Point cloud skeletons via laplacian-based contraction. In *Proc. of IEEE Conf. on Shape Modeling and Applications*, 2015.

[3] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

[4] Meng-Hao Guo, Jun-Xiong Cai, Zheng-Ning Liu, Tai-Jiang Mu, Ralph R Martin, and Shi-Min Hu. Pct: Point cloud transformer. *Computational Visual Media*, 7:187–199, 2021.

[5] H. Huang, S. Wu, D. Cohen-Or, M. Gong, H. Zhang, G. Li, and B.Chen. L1-medial skeleton of point cloud. *ACM Transactions on Graphics*, 32:65:1–65:8, 2013.

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

[7] Haoxi Ran, Jun Liu, and Chengjie Wang. Surface representation for point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18942–18952, 2022.

[8] Matthias Schneider, Sven Hirsch, Bruno Weber, Gábor Székely, and Bjoern H Menze. Tgif: Topological gap infill for vascular networks: A generative physiologicalmodeling approach. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2014: 17th International Conference, Boston, MA, USA, September 14-18, 2014, Proceedings, Part II 17*, pages 89–96. Springer, 2014.

[9] Matthias Schneider, Johannes Reichold, Bruno Weber, Gábor Székely, and Sven Hirsch. Tissue metabolism driven arterial tree generation. *Medical image analysis*, 16(7):1397–1414, 2012.

[10] Andrea Tagliasacchi, Hao Zhang, and Daniel Cohen-Or. Curve skeleton extraction from incomplete point cloud. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 2009.

[11] Giles Tetteh, Velizar Efremov, Nils D. Forkert, Matthias Schneider, Jan Kirschke, Bruno Weber, Claus Zimmer, Marie Piraud, and Björn H. Menze. Deepvesselnet: Vessel segmentation, centerline prediction, and bifurcation detection in 3-d angiographic volumes. *Frontiers in Neuroscience*, 14, 2020.

[12] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5):1–12, 2019.

[13] Ruben Wiersma, Ahmad Nasikun, Elmar Eisemann, and Klaus Hildebrandt. Deltaconv: anisotropic operators for geometric deep learning on point clouds. *ACM Transactions on Graphics (TOG)*, 41(4):1–10, 2022.

[14] Tiange Xiang, Chaoyi Zhang, Yang Song, Jianhui Yu, and Weidong Cai. Walk in the cloud: Learning curves for point clouds shape analysis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 915–924, 2021.