Enhanced Boundary Learning for Glass-like Object Segmentation Supplementary Material

Hao He[1,2*], Xiangtai Li[3*], Guangliang Cheng[4,6], Jianping Shi[4],
Yunhai Tong[3], Gaofeng Meng[1,2,5], Véronique Prinet[1], LuBin Weng[1],

[1] National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences

[2] School of Artificial Intelligence, University of Chinese Academy of Sciences, [3] Key Laboratory of Machine Perception (MOE), Peking University

[4] SenseTime Group Research [5] Centre for Artificial Intelligence and Robotics, HK Institute of Science & Innovation, CAS [6] Shanghai AI Lab

Email: hehao2019@ia.ac.cn, lxtpku@pku.edu.cn

Overview This supplementary file mainly contains two parts. In the first part, we provide more experimental analysis on RDM, PGM and cascade decoder as the supplementary of our experiment section in the main paper. Following [22], we also report more detailed results in Trans10k benchmark. Then we give the implementation details of Cityscapes [4], BDD [27], and COCO Stuff [1]. For the second part, we present more visualization results on three glass-like datasets [22, 25, 13].

1. More analysis experiments

In this section, we provide several supplementary experiments for main paper.

Visual Comparison on Boundary Results: Fig. 1 gives some boundary segmentation results on Trans10k test set. There are three different scenes, pictures of the first two rows contain different things object, the third and the fourth rows show a scene that the dividing line between two glass-like object is thin and hard to detection, the last two rows show a scene that the information inside the mirror is complex. In these three challenging scenes, boundary results of our method is much better than the boundary detection method: HED [23] and the transparent object segmentation method: Translab [22].

Effectiveness of RDM on Boundary Prediction: We show more effective evidence on RDM. We first demonstrate more visual examples on F_{edge} in Fig. 2. Compared to the main paper, we also visualize the predicted mask boundary comparison with the model trained with and without residual supervision. As shown in the Fig. 2, our differential design leads to thinner feature representation and better boundary results.

Experiments on Various Backbone: We perform experiments on various backbones including light-weight ResNet18 and heavy ResNet101 in Tab. 1 where we use DeeplaV3+ based EBLNet. On the ResNet18 backbone, our method results in a significant gain over the baseline by 4.9% mIoU. On the strong baseline with ResNet101, our method results in a 3.6% mIoU gain.

BackBone	mIoU ↑	mBER ↓	MAE ↓
ResNet18	82.8	8.01	0.088
+our module	87.7	5.18	0.064
ResNet101	87.5	7.78	0.064
+our module	91.1	3.79	0.043

Table 1: Comparison of DeeplabV3+ and our method using resnet18 and resnet101 as backbone network. Our method can achieve significant improvement on different backbones.

Settings	mIoU ↑	mBER ↓	MAE ↓
1(w/o cascade)	89.4	4.59	0.053
2	90.2	4.19	0.050
3	**90.7**	**3.97**	**0.047**
4	89.7	4.57	0.053

Table 2: Influence of the number of cascade stage.

Method	mIoU↑	mBER↓	mAE↓
Deeplabv3+(baseline)	85.4	6.73	0.075
PointRend [8]	88.2	4.86	0.060
Translab [22]	88.9	4.91	0.056
GSCNN [17]	88.2	5.10	0.059
EBLNet	**89.5**	**4.54**	**0.051**

Table 3: Comparison results with related methods on Trans10k validation set. All the models are trained in the same setting and tested with a single scale inference on the Trans10k validation set.

*Equal Contribution. Corresponding to: LuBin Weng, Guangliang Cheng

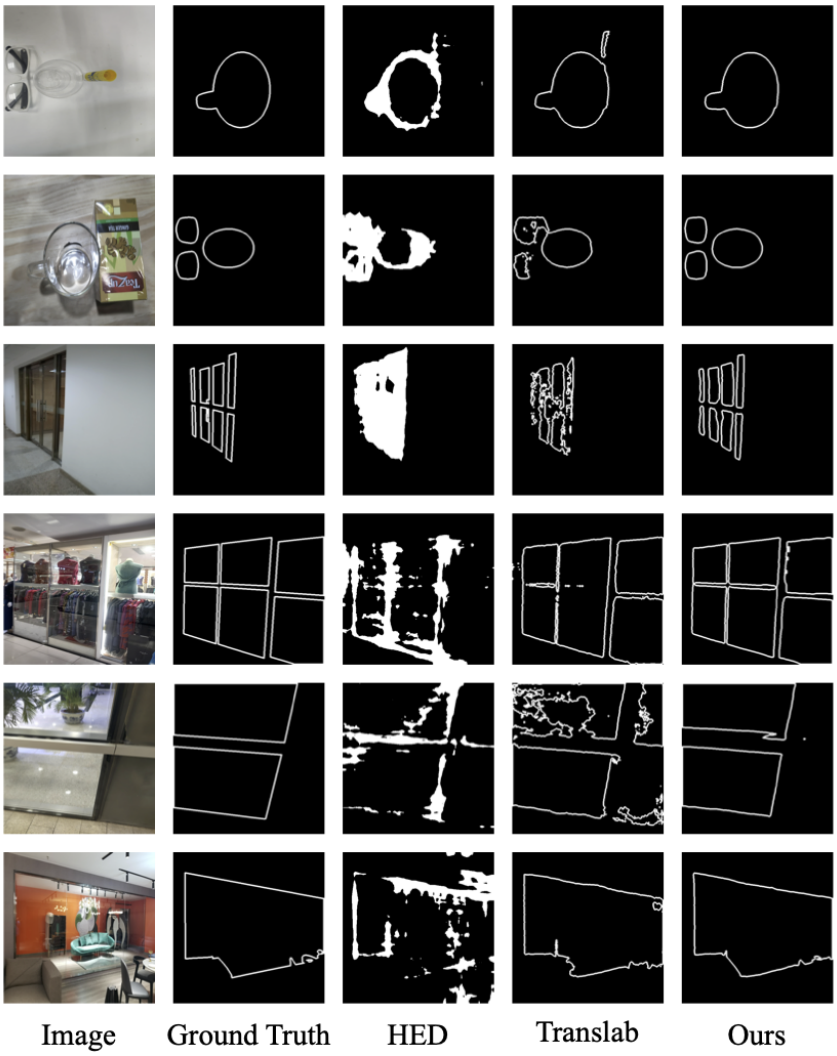

Figure 1: Visualization and comparison results of boundary on Trans10k test set.

Ablation on Number of Cascade Stage: We verify the number of cascade stages in Tab. 2. Appending more modules with more refined features leads to better results shown in the first 3 rows. After appending 4 modules, there exists a slight decrease. We argue that involving over-loaded low-level features weakens the semantic meaning and thus we set cascade number to 3 by default. To make the network architecture easier to understand, we give the network architecture using only one RDM and one PGM in Fig. 3.

Ablation on our PGM: We carry the effect of points number experiments in Fig. 4 where the Deeplabv3+ with 3 cascaded stages as base module. As shown in Fig. 4, the sampled point number is set to 96 according to the mIoU metric for the best trade-off between accuracy and computation. When sampling more points, the performance slightly drop. We argue that more points leads to redundant representation of boundaries and makes the learning of affinity unbalanced and difficult. Thus it degrades to the case as DANet [5] in Fig.4 (a) of the main paper.

Comparison with related methods: We select three representative works on segmentation boundary learning [17, 8, 22]. We use DeeplabV3+ with ResNet50 backbone as the baseline model for comparison. As shown in Tab. 3, our method achieves better results than those works on three metrics with **only one** RDM and **one** PGM involved.

Detailed results on Trans10k: We give the detailed results following the split of Trans10k benchmark [22] in Tab. 4 for further reference.

Implementation details on Cityscapes, BDD, and COCO Stuff: All networks are trained with the same setting, where stochastic gradient descent (SGD) with a batch size of 8 is used as the optimizer, with the momentum of 0.9, the weight decay of $5e-4$. The 'poly' learning rate policy is adopted to decay the initial learning rate by multi-

plying $(1 - \frac{\text{iter}}{\text{total\_iter}})^{0.9}$ during training. Data augmentation contains random horizontal flip, random resize with the scale range of $[0.5,\ 2.0]$, and random crop with size 800×800, 800×800, 512×512 for Cityscapes, BDD, and COCO Stuff datasets, respectively. We use ResNet-50 and ResNet-101 [6] as the backbones. Additionally, we re-implement the state-of-the-arts [3, 8] for fairness. We train 100 epochs, 100 epochs, and 60 epochs for Cityscapes, BDD, and COCO Stuff dataset, respectively and report results on validation datasets using single scale test. We port the PointRend [8] implementation from Detectron2 [21] codebase and then are trained under our settings.

2. More visualization results

In this section, we present more visual results on the datasets [13, 22, 25] in the main paper.

Effectiveness of PGM on boundary prediction: We provide more visualization results on PGM in Fig. 5 where our PGM refines the object boundaries and obtains more consistent results. For example, the bottom of glass in the first row(right) is refined into the consistent shape.

More visualization results on Trans10k: Fig. 6 provides more visualization results on Trans10k dataset. Compared with current state-of-the-art methods [3, 8, 22], our methods achieve better results.

More visualization results on GDD and MSD: Fig. 7 shows more visualization results on MSD (top) and GDD (bottom) dataset.

References

[1] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. Cocostuff: Thing and stuff classes in context. In *CVPR*, 2018. 1

[2] Ping Chao, Chao-Yang Kao, Yu-Shan Ruan, Chien-Hsiang Huang, and Youn-Long Lin. Hardnet: A low memory traffic network. In *ICCV*, pages 3552–3561, 2019. 7

[3] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018. 3, 7

[4] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016. 1

[5] Jun Fu, Jing Liu, Haijie Tian, Zhiwei Fang, and Hanqing Lu. Dual attention network for scene segmentation. In *CVPR*, 2019. 2

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 3

[7] Qiangguo Jin, Zhaopeng Meng, Tuan D Pham, Qi Chen, Leyi Wei, and Ran Su. Dunet: A deformable network for retinal vessel segmentation. *Knowledge-Based Systems*, 178:149–162, 2019. 7

[8] Alexander Kirillov, Yuxin Wu, Kaiming He, and Ross Girshick. PointRend: Image segmentation as rendering. In *CVPR*, 2020. 1, 2, 3

[9] Gen Li, Inyoung Yun, Jonghyun Kim, and Joongkyu Kim. Dabnet: Depth-wise asymmetric bottleneck for real-time semantic segmentation. *arXiv preprint arXiv:1907.11357*, 2019. 7

[10] Guosheng Lin, Anton Milan, Chunhua Shen, and Ian D. Reid. Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. In *CVPR*, 2017. 7

[11] Mengyu Liu and Hujun Yin. Feature pyramid encoding network for real-time semantic segmentation. *arXiv preprint arXiv:1909.08599*, 2019. 7

[12] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 7

[13] Haiyang Mei, Xin Yang, Yang Wang, Yuanyuan Liu, Shengfeng He, Qiang Zhang, Xiaopeng Wei, and Rynson W.H. Lau. Don't hit me! glass detection in real-world scenes. In *CVPR*, 2020. 1, 3, 9

[14] Rudra PK Poudel, Ujwal Bonde, Stephan Liwicki, and Christopher Zach. Contextnet: Exploring context and detail for semantic segmentation in real-time. *arXiv preprint arXiv:1805.04554*, 2018. 7

[15] Rudra PK Poudel, Stephan Liwicki, and Roberto Cipolla. Fast-scnn: Fast semantic segmentation network. *arXiv preprint arXiv:1902.04502*, 2019. 7

[16] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *MICCAI*, 2015. 7

[17] Towaki Takikawa, David Acuna, Varun Jampani, and Sanja Fidler. Gated-scnn: Gated shape cnns for semantic segmentation. *ICCV*, 2019. 1, 2

[18] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, et al. Deep high-resolution representation learning for visual recognition. *PAMI*, 2020. 7

[19] Yu Wang, Quan Zhou, Jia Liu, Jian Xiong, Guangwei Gao, Xiaofu Wu, and Longin Jan Latecki. Lednet: A lightweight encoder-decoder network for real-time semantic segmentation. In *ICIP*, pages 1860–1864. IEEE, 2019. 7

[20] Tianyi Wu, Sheng Tang, Rui Zhang, and Yongdong Zhang. Cgnet: A light-weight context guided network for semantic segmentation. *arXiv preprint arXiv:1811.08201*, 2018. 7

[21] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019. 3

[22] Enze Xie, Wenjia Wang, Wenhai Wang, Mingyu Ding, Chunhua Shen, and Ping Luo. Segmenting transparent objects in the wild. *ECCV*, 2020. 1, 2, 3, 7

[23] Saining Xie and Zhuowen Tu. Holistically-nested edge detection. In *ICCV*, 2015. 1

[24] Maoke Yang, Kun Yu, Chi Zhang, Zhiwei Li, and Kuiyuan Yang. Denseaspp for semantic segmentation in street scenes. In *CVPR*, 2018. 7

[25] Xin Yang, Haiyang Mei, Ke Xu, Xiaopeng Wei, Baocai Yin, and Rynson WH Lau. Where is my mirror? In *ICCV*, 2019. 1, 3, 9

[26] Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu, and Nong Sang. Bisenet: Bilateral segmentation network for real-time semantic segmentation. In *ECCV*, 2018. 7

[27] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *CVPR*, 2020. 1

[28] Yuhui Yuan and Jingdong Wang. Ocnet: Object context network for scene parsing. *arXiv preprint*, 2018. 7

[29] Hengshuang Zhao, Xiaojuan Qi, Xiaoyong Shen, Jianping Shi, and Jiaya Jia. Icnet for real-time semantic segmentation on high-resolution images. In *ECCV*, 2018. 7

[30] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *CVPR*, 2017. 7

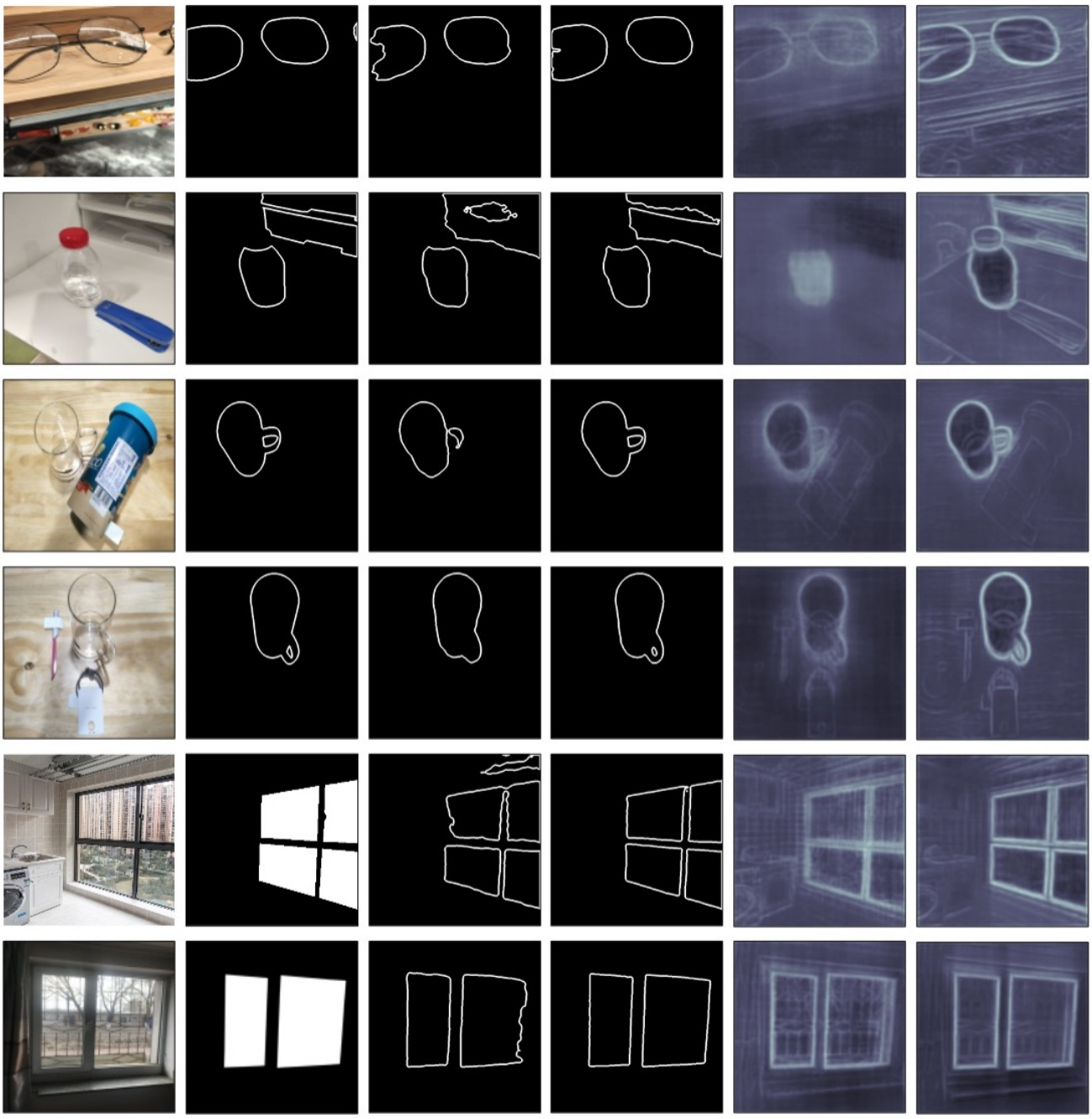

Figure 2: More visualization results on RDM with boundary comparison. Best view it on screen.

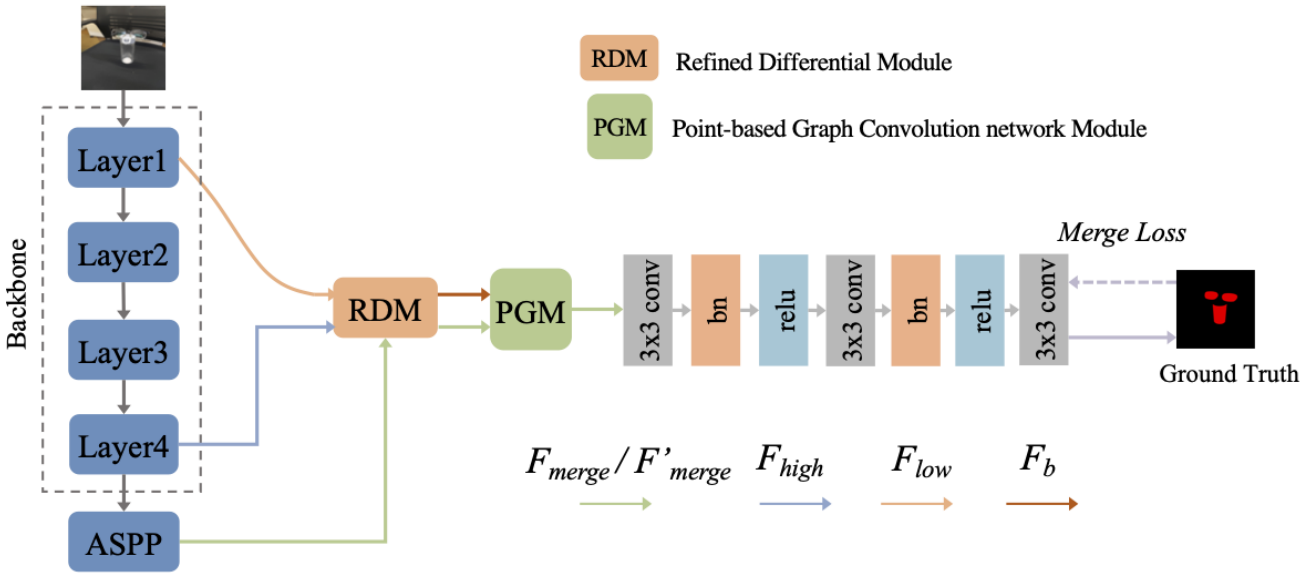

Figure 3: Network architecture without using cascade structure.

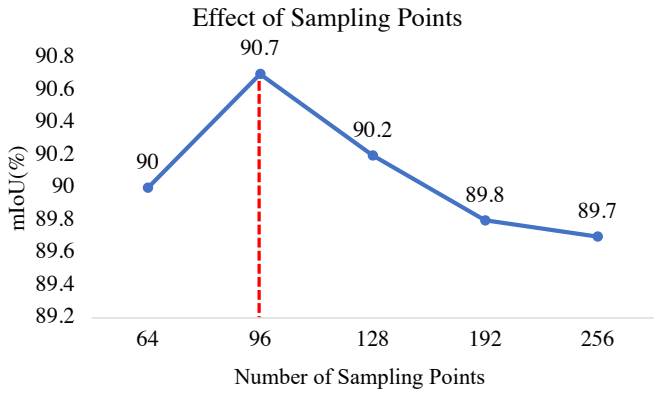

Figure 4: Influence of the number of sampled points in the PGM.

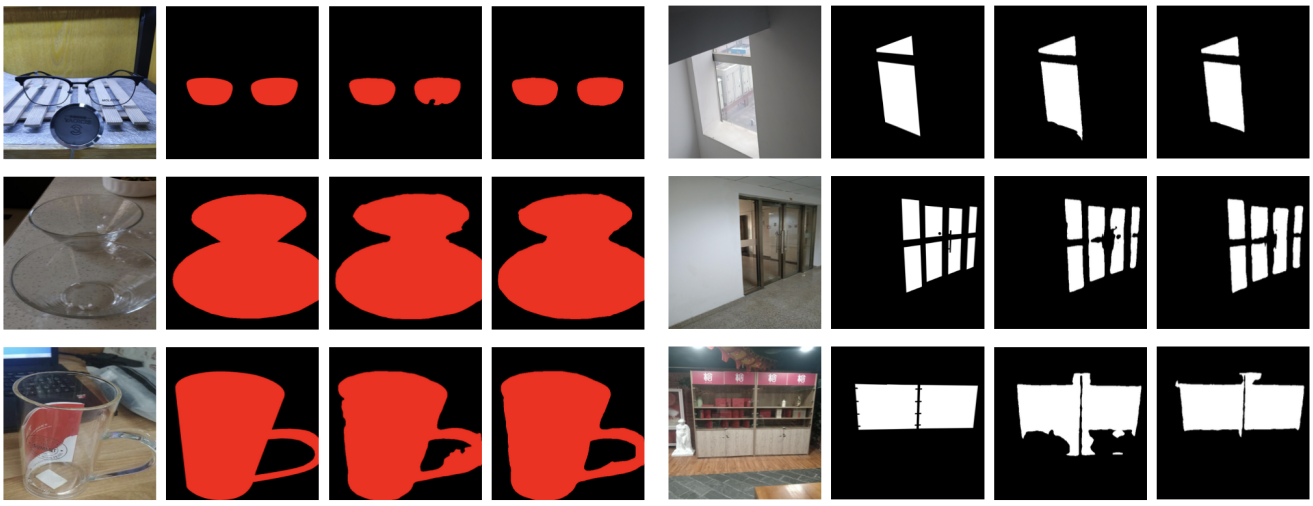

Figure 5: More visualization results on PGM. The boundary are refined by our efficient PGM module. The first four rows are thing classes while the last four rows are stuff classes in Trans10k [22]. Best view it on screen.

Method	MAE ↓			Acc ↑			mIoU ↑			mBER ↓		
	All	Easy	Hard	All	Easy	Hard	All	Easy	Hard	All	Easy	Hard
FPENet [11]	0.339	0.297	0.492	24.73	26.50	19.19	34.17	36.82	24.41	39.31	37.88	44.03
ContextNet [14]	0.217	0.171	0.386	62.09	67.14	46.34	56.46	61.73	37.71	22.36	18.77	34.44
FastSCNN [15]	0.206	0.161	0.373	64.20	69.42	48.01	59.18	64.63	40.27	22.27	18.74	34.22
Deeplabv3+MBv2 [3]	0.130	0.091	0.275	80.92	85.90	65.43	75.27	80.55	56.17	12.49	9.08	24.47
CGNet [20]	0.216	0.173	0.379	59.15	64.57	42.26	57.31	62.41	39.56	22.95	19.67	34.33
HRNet [18]	0.134	0.092	0.291	75.82	82.17	56.04	74.56	80.43	53.42	13.52	9.95	26.17
HardNet [2]	0.184	0.141	0.345	69.17	73.83	54.67	64.03	69.11	46.18	18.91	15.58	30.52
DABNet [9]	0.230	0.187	0.391	54.87	59.29	41.07	54.90	59.45	38.77	25.71	22.63	36.15
LEDNet [19]	0.168	0.124	0.331	75.70	80.62	60.37	67.54	73.04	48.38	15.15	11.83	26.58
ICNet [29]	0.244	0.200	0.408	52.65	58.31	35.01	50.65	55.48	33.44	24.63	21.71	35.24
BiSeNet [26]	0.140	0.102	0.282	77.92	82.79	62.72	73.93	78.74	56.37	13.96	10.83	24.85
DenseAspp [24]	0.114	0.078	0.247	81.22	86.25	66.55	78.11	83.11	60.38	12.19	8.85	23.71
DeepLabv3+R50 [3]	0.081	0.050	0.194	89.54	93.22	78.07	84.54	89.09	69.04	7.78	4.91	17.27
FCN [12]	0.108	0.073	0.239	83.79	88.55	68.93	79.67	84.53	62.51	10.33	7.36	20.47
OCNet [28]	0.122	0.087	0.253	80.85	85.63	65.96	76.85	81.53	59.75	12.65	9.43	23.69
RefineNet [10]	0.180	0.135	0.345	57.97	64.53	37.53	66.03	71.41	45.71	22.22	19.01	34.06
DeepLabv3+XP65 [3]	0.082	0.051	0.195	89.18	92.61	78.51	84.26	88.87	68.34	8.00	5.16	17.44
DUNet [7]	0.140	0.100	0.289	77.84	83.41	60.50	74.06	79.19	55.53	13.19	9.93	25.01
UNet [16]	0.234	0.191	0.398	51.07	55.44	37.44	53.98	58.60	37.08	26.37	23.40	36.80
PSPNet [30]	0.093	0.062	0.211	86.25	90.41	73.28	82.38	86.79	66.35	9.72	6.67	20.08
Translab [22]	0.063	0.036	0.166	92.69	95.77	83.04	87.63	92.23	72.10	5.46	3.12	13.30
EBLNet(OS16)	0.052	0.029	0.140	93.95	96.68	85.44	89.58	93.68	75.54	4.60	2.50	11.49
EBLNet(OS8)	**0.048**	**0.026**	**0.130**	**94.71**	**97.19**	**86.95**	**90.28**	**94.21**	**76.51**	**4.14**	**2.22**	**10.45**

Table 4: Comparison on easy and hard samples in Trans10k dataset. Results are reported with single scale inputs. The bold values in each column mean the best entries. OS means the output stride in backbone.

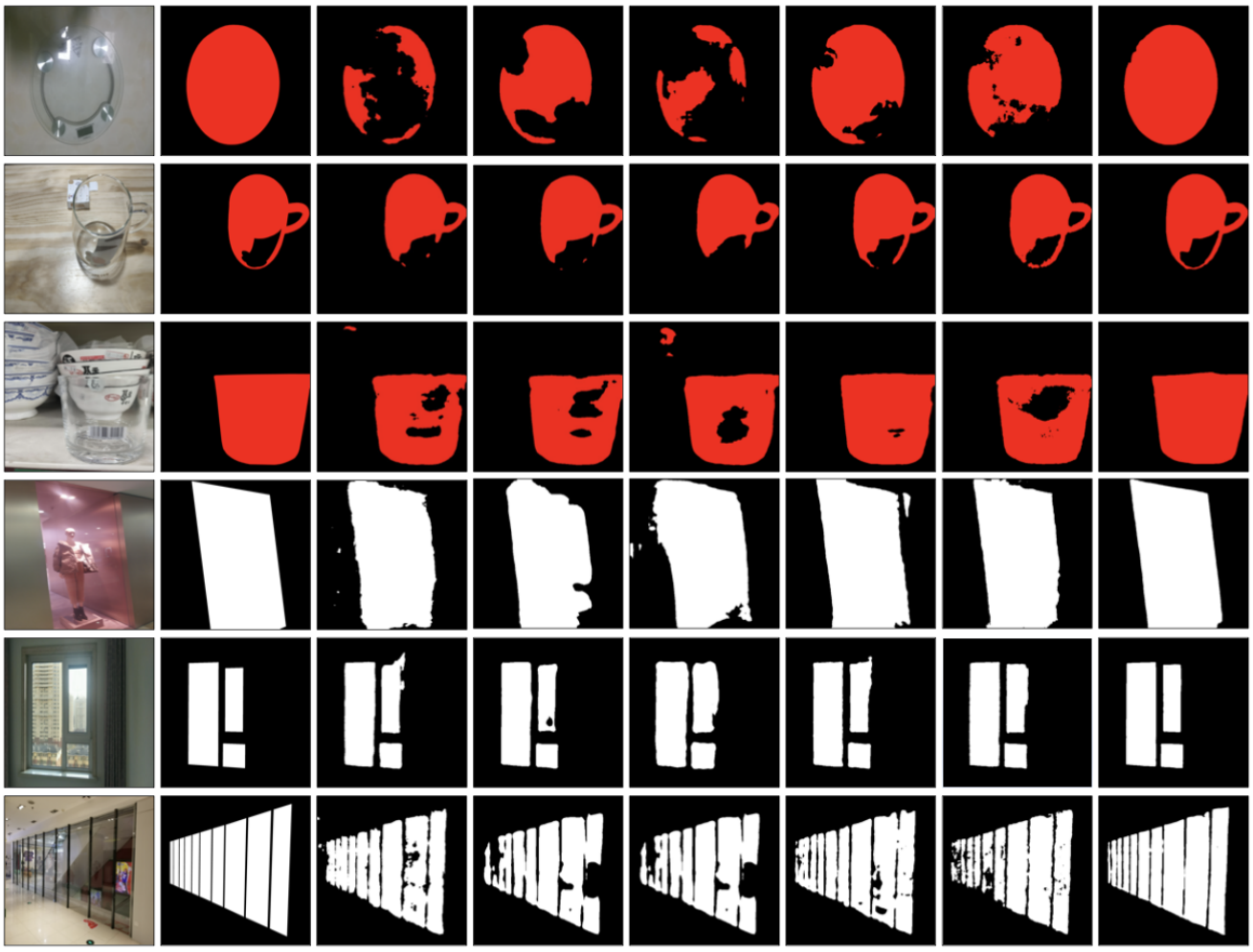

Figure 6: More visualization results on Trans10k datasets. Best view it on screen.

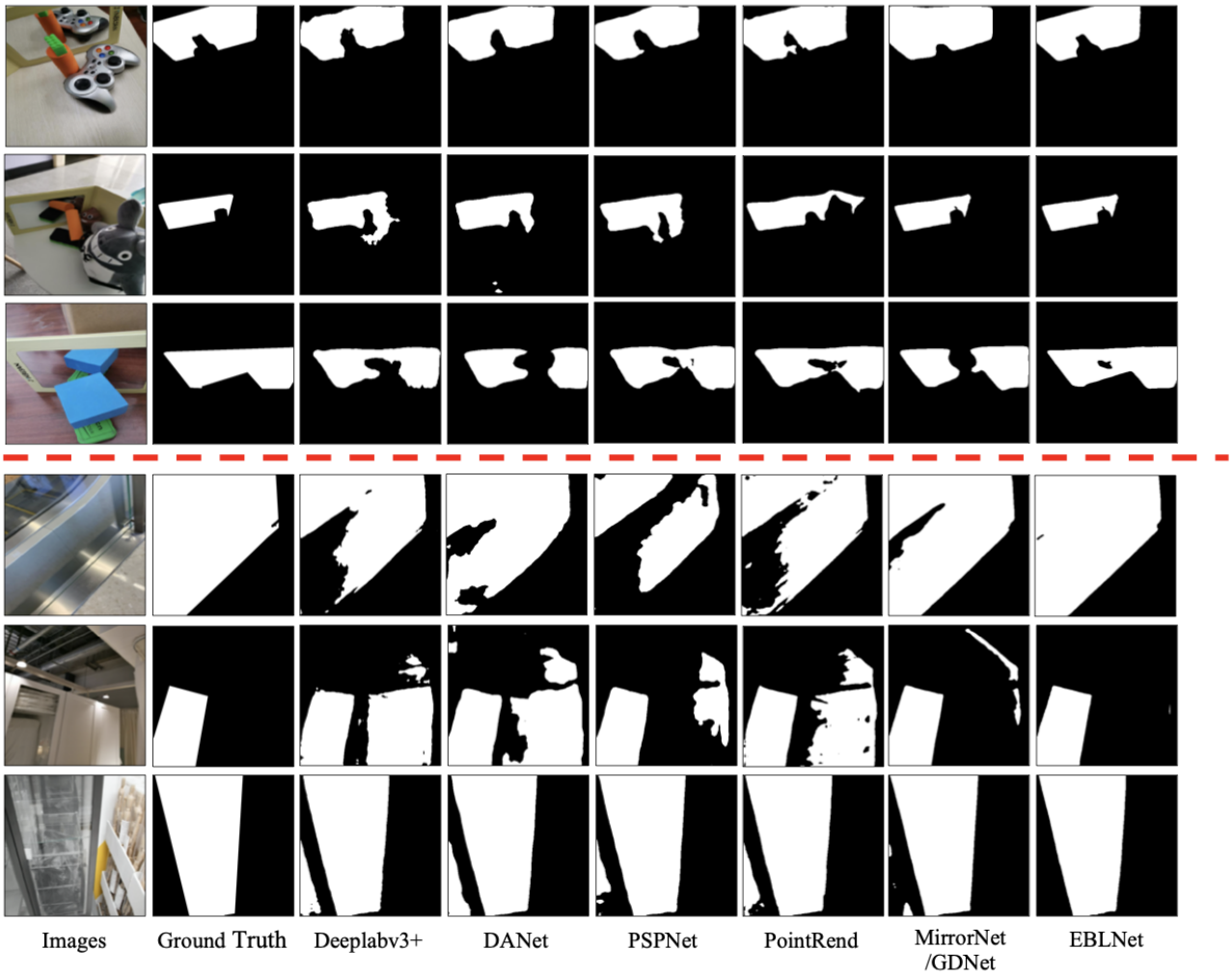

Figure 7: More visualization results on MSD and GDD datasets. The first three are on MSD [25] and the last three are on GDD [13]. The first three rows of the 7th column are MirrorNet results, the last three rows of the 7th column are GDNet results. Best view it on screen.