Semantic-shape Adaptive Feature Modulation for Semantic Image Synthesis (Supplementary Material)

Zhengyao Lv[1], Xiaoming Li[2], Zhenxing Niu[3], Bing Cao[4], Wangmeng Zuo[2,5](✉)
[1]Tomorrow Advancing Life [2]Harbin Institute of Technology
[3]Machine Intelligence Lab, Alibaba Group [4]Tianjin University [5]Peng Cheng Laboratory
{cszy98, hit.xmshr}@gmail.com wmzuo@hit.edu.cn

Algorithm 1 Calculation of the SPD

Input: Points set inside the instance object $\mathbf{P} \in \mathcal{R}^{m\times 2}$ and points set $\mathbf{C} \in \mathcal{R}^{n\times 2}$ on the contour of the instance. Additional input $rbins$ and $tbins$ represent the distance and angle intervals, respectively.
Output: The SPD map $\mathbf{G} \in \mathcal{R}^{h\times w\times 72}$ of the instance.

*# The distance r between points in **P** and **C**.*
1. xdis = **P[:, 0]**.reshape((-1, 1)) - **C[:, 0]**.reshape((1,-1))
   ydis = **P[:, 1]**.reshape((-1, 1)) - **C[:, 1]**.reshape((1,-1))
2. rarray = torch.sqrt(xdis ** 2 + ydis ** 2)
   rarray = rarray / (torch.max(rarray) / 2)

 # The number of points in different distance intervals.
3. for r in rbins:
       rq += (rarray <= r)

*# The angle t between points in **P** and **C**.*
4. tarray = torch.atan2(xgap, -ygap)
   tarray = tarray + 2 * math.pi * (tarray < 0)

 # The number of points in different angle intervals.
5. tq = (1 + torch.floor(tarray / (2 * math.pi / tbins)))
6. Count the number of points in each interval with rq and tq and get the final SPD map **G**.

1. Calculation of the SPD

The detailed calculation process of the shape-aware position descriptor (SPD) is shown in the pseudo codes Alg. 1. In this process, we use GPU to speed up the calculation.

2. Network Architecture

Our generator is mainly composed of the semantic-shape adaptive feature modualtion (SAFM) block. Table A shows the details of the SAFM block. The f_{t-1} denotes the features from the previous layer and the f_t denotes the features modulated by current SAFM block, respectively. Seg is the input semantic layouts and SPD is the shape-aware position descriptor maps. Conv and Depthwise Conv represent the convolution operation and depthwise convolution, in which convolution kernels are adaptively predicted from semantic layouts. Unfold extracts sliding local blocks from input for conditional convolution operation. Concat denotes the concatenation operation.

<table>
<tr><td>Input</td><td>f_{t-1}</td><td colspan="3">Seg</td><td colspan="2">SPD</td></tr>
<tr><td rowspan="6">SAFM</td><td rowspan="5">f_{t-1}</td><td rowspan="4">Seg</td><td colspan="2">Conv</td><td>Conv</td><td rowspan="4">SPD</td></tr>
<tr><td></td><td>Conv</td><td>Unfold</td></tr>
<tr><td>Conv</td><td colspan="2">Depthwise Conv</td></tr>
<tr><td colspan="3">Depthwise Conv</td></tr>
<tr><td colspan="5">Concat</td></tr>
<tr><td colspan="5">Conv</td></tr>
<tr><td colspan="3">Conv</td><td colspan="2">Conv</td></tr>
<tr><td colspan="4">Element-wise Multiply</td><td></td></tr>
<tr><td colspan="6">Element-wise Sum</td></tr>
<tr><td>output</td><td colspan="6">f_t</td></tr>
</table>

Table A. Details of the SAFM block.

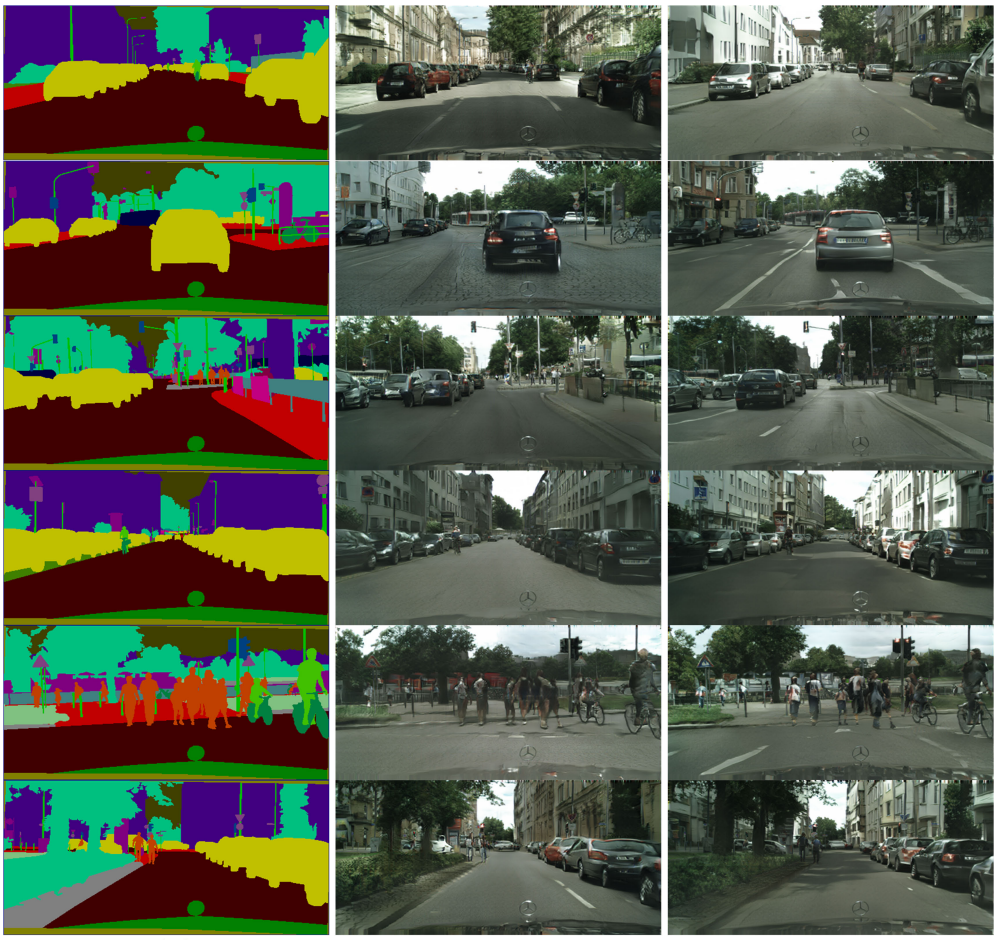
3. More Qualitative Results

We show more visual comparisons with the competing methods (*i.e.* SPADE [2], CC-FPSE [1], and OASIS [3]) on the Cityscapes, ADE20K and COCO-Stuff, as shown in Fig. B, Fig. C and Fig. D. Zoom in for more details. It can be seen that our method can generate more semantically aligned and photo-realistic images with rich details.

Additionally, we validate the effectiveness of SPD with OASIS [3] discriminator and training tricks. The generated results are shown in Fig. A, showing the benefit of SPD for realistic detail synthesis.

4. More Analysis on Instances Synthesis

To quantify the effect of our method on instance synthesis, we evaluate semantic segmentation and detection metrics for instance classes of synthesized results in the Cityscapes dataset. The per-class quantitative results of Cityscapes on the synthesized object instance are shown in Table B. One can see that our methods (*Ours w/o* L_{seg} and *Ours-Full*) greatly improve the performance upon the *Baseline*, performing favorably against the SoTAs in terms of the semantic segmentation (mIoU) and detection (AP) metrics.



Label OASIS OASIS+SPD

Figure A. Visual comparisons on the ADE20K and COCO-Stuff datasets.

Method	Person	Rider	Car	Truck	Bus	Train	Mcycle	Bcycle	All Objects
SPADE	14.0/62.27	16.4/38.67	26.6/88.68	18.0/64.95	28.5/70.16	7.9/41.44	5.4/28.59	9.2/58.86	15.7/56.70
LGGAN	14.8/64.47	18.3/45.99	27.4/90.17	18.8/73.29	34.1/79.05	11.8/52.73	9.1/39.08	11.1/61.38	18.2/63.27
OASIS	11.3/59.53	19.5/47.96	19.2/87.37	28.0/62.21	36.7/75.04	16.3/59.95	9.9/48.35	9.3/58.40	18.8/62.35
Baseline	15.2/65.31	17.5/43.54	28.1/89.97	19.2/68.02	34.2/72.05	18.0/45.14	6.7/38.58	10.8/62.88	18.7/60.69
Ours w/o L_{seg}	17.3/66.80	21.0/46.85	31.3/90.53	22.4/78.81	34.4/75.97	11.5/61.98	9.6/46.05	13.6/64.11	20.1/66.39
Ours-Full	17.7/68.04	20.5/49.90	31.1/91.01	24.9/76.58	38.9/78.53	14.1/47.38	9.9/45.28	13.1/65.34	21.3/65.26

Table B. Per-class quantitative comparison of the Detection (AP) / Semantic Segmentation (mIoU) metrics on object classes of Cityscapes. Red and blue indicate the best and the second best results, respectively. *Ours w/o* L_{seg} is a variant of *Ours-Full* without using L_{seg}.

References

[1] Xihui Liu, Guojun Yin, Jing Shao, Xiaogang Wang, and Hongsheng Li. Learning to predict layout-to-image conditional convolutions for semantic image synthesis. *arXiv preprint arXiv:1910.06809*, 2019. 1

[2] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2337–2346, 2019. 1

[3] Vadim Sushko, Edgar Schönfeld, Dan Zhang, Juergen Gall, Bernt Schiele, and Anna Khoreva. You only need adversarial supervision for semantic image synthesis. *arXiv preprint arXiv:2012.04781*, 2020. 1

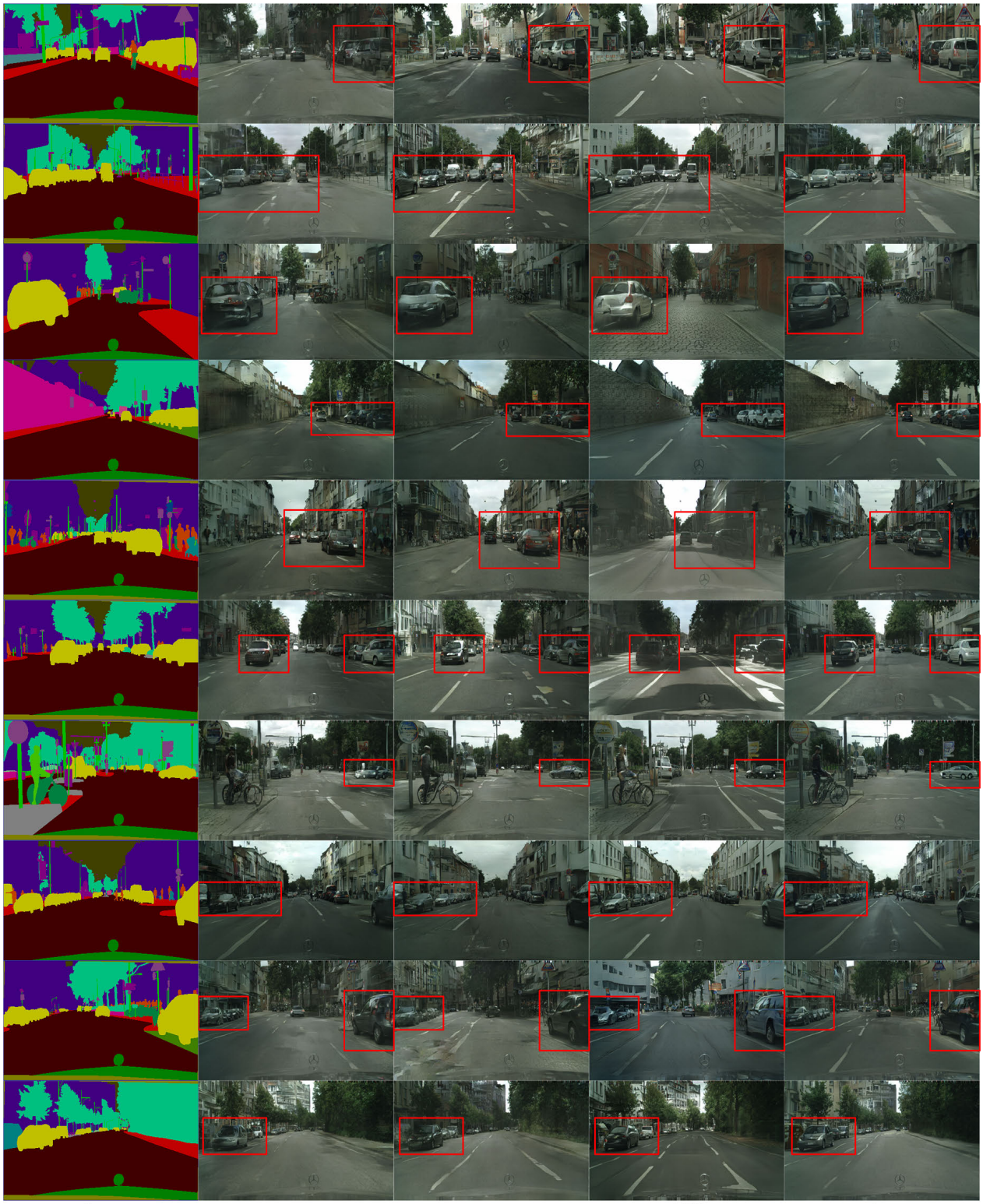


Figure B. Visual comparisons on the Cityscapes dataset.

Figure C. Visual comparisons on the ADE20K and COCO-Stuff datasets.

Figure D. Visual comparisons on the ADE20K and COCO-Stuff datasets.