MonoScene: Monocular 3D Semantic Scene Completion
– Supplementary Material –

Anh-Quan Cao
Inria
anh-quan.cao@inria.fr

Raoul de Charette
Inria
raoul.de-charette@inria.fr

We provide details about the main baselines and MonoScene in Sec. 1, and include additional qualitative and quantitative results in Sec. 2.

Results on image sequences are in the supplementary video: https://youtu.be/qh7La1tRJmE.

1. Architectures details

1.1. Baselines

AICNet [8]. We use the official implementation of AICNet1. For the RGB-inferred version, i.e. AICNetrgb, we infer depth with the pre-trained AdaBins [2] on NYUv2 [14] and SemanticKITTI [1] from the official repository2.

3DSketch [4]. We use 3DSketch official code3. For 3DSketchrgb, we again use AdaBins (cf. above) and convert depth to TSDF with ‘tsdf-fusion’4 from the 3DMatch Toolbox [17].

JS3C-Net [16]. We use the official code of JS3C-Net5. For JS3C-Netrgb, we generate the input point cloud by unprojecting the predicted depth (using AdaBins) to 3D using the camera intrinsics. The semantic point clouds, required to train JS3C-Net, are obtained by augmenting the unprojected point clouds with the 2D semantics obtained using the code6 of [19].

LMSCNet [12]. We use the official implementation of LMSCNet7. For LMSCNetrgb, the input occupancy grid is obtained by discretizing the unprojected point cloud.

1 https://github.com/waterljwant/SSC
2 https://github.com/shariqfarooq123/AdaBins
3 https://github.com/charlesCXX/TorchSSC
4 https://github.com/andyzeng/tsdf-fusion
5 https://github.com/yanx27/JS3C-Net
6 https://github.com/YeLyuUT/SSeg
7 https://github.com/cv-rits/LMSCNet

Figure 1. MonoScene 3D network. The 3D UNet uses 2 downscale layers with DDR blocks [9] and 2 upscale layers with deconv. The completion head uses ASPP and an optional deconv layer. Notations: DDR(dilation, downsample rate), Deconv(kernel size, dilation), ASPP(dilations).

1.2. MonoScene

Fig. 1 details our 3D UNet. Similar to 3DSketch [4], we adopt DDR [9] as the basic building block for large receptive field and low memory cost. The 3D encoder has 2 layers, each downscales by half and has 4 DDR blocks. The 3D decoder has two deconv layers, each doubles the scale. Similar to others [12] the completion head has an ASPP with dilations (1, 2, 3) to gather multi-scale features and an optional deconv to reach output size – used in SemanticKITTI only.

For training, MonoScene took 7 hours using 2 V100 32g GPUs (2 items per GPU) on NYUv2 [14] and 28 hours to train using 4 V100 32g GPUs (1 item per GPU) on SemanticKITTI [1].

2. Additional results

2.1. SemanticKITTI

Quantitative performance. We report performance on validation set in Tab. 1. Comparing against the test set performance from the main paper Tab. 1b, we notice MonoScene generalizes better than JS3C-Netrgb and AICNetrgb since the validation and test set gap is smaller (−0.42 vs −1.34 and −1.22). We also report the complete SemanticKITTI official benchmark (i.e. hidden test set) in Tab. 2 showing that while MonoScene uses only RGB, it still outperforms some of the 3D input SSC baselines.
Table 1. Performance on SemanticKITTI [1] (validation set). We report the performance on semantic scene completion (SSC - mIoU) and scene completion (SC - IoU) for RGB-inferred baselines and our method.

<table>
<thead>
<tr>
<th>Method</th>
<th>SSC Input</th>
<th>IOU</th>
<th>mIoU</th>
</tr>
</thead>
<tbody>
<tr>
<td>LMSCNet$^b$ [12]</td>
<td>$x^{rgb}$</td>
<td>28.64</td>
<td>10.31</td>
</tr>
<tr>
<td>3DSketch$^b$ [4]</td>
<td>$x^{rgb}$</td>
<td>33.30</td>
<td>14.81</td>
</tr>
<tr>
<td>AICNet$^b$ [5]</td>
<td>$x^{rgb}; x^{3D}$</td>
<td>29.59</td>
<td>14.71</td>
</tr>
<tr>
<td>*JS3C-Net$^b$ [16]</td>
<td>$x^{rgb}; x^{depth}$</td>
<td>38.98</td>
<td>24.65</td>
</tr>
</tbody>
</table>

MonoScene (ours) | $x^{rgb}$ | 37.12 | 28.71 |

* Uses pretrained semantic segmentation network.

Table 2. Complete SemanticKITTI official benchmark (hidden test set). Results are taken from [13]. Despite using only single RGB image as input, MonoScene still surpasses some of the SSC baselines with 3D input.

<table>
<thead>
<tr>
<th>Method</th>
<th>Input</th>
<th>IOU</th>
<th>mIoU</th>
</tr>
</thead>
<tbody>
<tr>
<td>3D</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SSCNet [15]</td>
<td>$x^{3D}$</td>
<td>29.8</td>
<td>9.5</td>
</tr>
<tr>
<td>TS3D [7]</td>
<td>$x^{3D} + x^{rgb}$</td>
<td>29.8</td>
<td>9.5</td>
</tr>
<tr>
<td>TS3D+Net [1]</td>
<td>$x^{3D} + x^{rgb}$</td>
<td>25.0</td>
<td>10.2</td>
</tr>
<tr>
<td>ESSCNet [18]</td>
<td>$x^{occ}$</td>
<td>41.8</td>
<td>17.5</td>
</tr>
<tr>
<td>LMSCNet [12]</td>
<td>$x^{occ}$</td>
<td>56.7</td>
<td>17.6</td>
</tr>
<tr>
<td>TS3D+Net+SATNet [1]</td>
<td>$x^{occ}$</td>
<td>50.6</td>
<td>17.7</td>
</tr>
<tr>
<td>Local-DIFs [11]</td>
<td>$x^{occ}$</td>
<td>57.7</td>
<td>22.7</td>
</tr>
<tr>
<td>JS3C-Net [16]</td>
<td>$x^{occ}$</td>
<td>56.6</td>
<td>23.8</td>
</tr>
<tr>
<td>S3CNet [5]</td>
<td>$x^{occ}$</td>
<td>45.6</td>
<td>29.5</td>
</tr>
</tbody>
</table>

| 2D           |        |       |       |
| MonoScene    | $x^{rgb}$ | 34.2  | 11.1  |

Table 2. Qualitative performance. In Fig. 2 we also include additional qualitative results. Compared to all baselines, MonoScene captures better landscape and objects (e.g., cars, trees, roads, the gap is smaller with KITTI-360 having similar important role. As MonoScene is trained on the mid-size German city of Karlsruhe, with residential scenes and narrow roads, the gap is smaller with KITTI-360 having similar scenes. The results on nuScenes and Cityscapes suffer both from the camera setup changes and the large metropolitan scenes (i.e. Stuttgart - Cityscapes; Singapore, Boston - nuScenes) having wider streets.

2.3. Generalization

Fig. 4 illustrates the predictions of MonoScene, trained on SemanticKITTI, on datasets with different camera setups. We can see the increase in distortion as the camera setups depart from the ones used during training. Furthermore, the domain gap (i.e., city, country, etc.) also plays an important role. As MonoScene is trained on the mid-size German city of Karlsruhe, with residential scenes and narrow roads, the gap is smaller with KITTI-360 having similar scenes. The results on nuScenes and Cityscapes suffer both from the camera setup changes and the large metropolitan scenes (i.e. Stuttgart - Cityscapes; Singapore, Boston - nuScenes) having wider streets.

Table 3. SemanticKITTI performance (validation set) on in-/out-FOV and the Whole Scene. We report the performance on the scenery inside (in-FOV), outside (out-FOV) camera FOV, and considering all voxels (Whole Scene). MonoScene is best in most cases, with in-FOV performance logically higher.

<table>
<thead>
<tr>
<th>Method</th>
<th>in-FOV</th>
<th>out-FOV</th>
<th>Whole Scene</th>
</tr>
</thead>
<tbody>
<tr>
<td>LMSCNet$^b$ [12]</td>
<td>37.62</td>
<td>25.36</td>
<td>34.41</td>
</tr>
<tr>
<td>3DSketch$^b$ [4]</td>
<td>32.24</td>
<td>26.50</td>
<td>33.30</td>
</tr>
<tr>
<td>AICNet$^b$ [5]</td>
<td>35.69</td>
<td>25.79</td>
<td>29.59</td>
</tr>
<tr>
<td>*JS3C-Net$^b$ [16]</td>
<td>42.22</td>
<td>28.27</td>
<td>38.98</td>
</tr>
</tbody>
</table>

MonoScene(ours) | 39.13  | 31.60   | 37.12       |

* Uses pretrained semantic segmentation network.
Figure 2. **Results on SemanticKITTI** [1] (validation set). The input is shown left. Darker voxels represent the scenery outside the viewing frustum (i.e. unseen by the image).
Figure 3. Results on NYUv2 [14] (test set). The input is shown leftmost and the camera viewing frustum is shown in the ground truth (rightmost).
Figure 4. **Domain gap and Camera effects.** Outputs of MonoScene when trained on SemanticKITTI having horizontal FOV of 82°, and tested on datasets with decreasing (left) or increasing (right) FOV. SemanticKITTI and KITTI-360 are recorded in mid-size German city of Karlsruhe while nuScenes and Cityscapes are from large metropolitan areas (e.g. Stuttgart - Cityscapes; Singapore, Boston - nuScenes) whose streets are much wider, denser and have different landscapes.

**References**


[4] Xiaokang Chen, Kwan-Yee Lin, Chen Qian, Gang Zeng, and Hongsheng Li. 3d sketch-aware semantic scene completion via semi-supervised structure prior. In CVPR, 2020. 1, 2, 4

[5] Ran Cheng, Christopher Agia, Yuan Ren, XinHai Li, and


[9] Jie Li, Yu Liu, Dong Gong, Qinfeng Shi, Xia Yuan, Chunxia Zhao, and Ian Reid. Rgbd based dimensional decomposition residual network for 3d semantic scene completion. In CVPR, 2019. 1


[16] Xu Yan, Jiantao Gao, Jie Li, Ruimao Zhang, Zhen Li, Rui Huang, and Shuguang Cui. Sparse single sweep lidar point cloud segmentation via learning contextual shape priors from scene completion. In AAAI, 2021. 1, 2, 3

