MonoScene: Monocular 3D Semantic Scene Completion

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

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

Abstract

MonoScene proposes a 3D Semantic Scene Completion (SSC) framework, where the dense geometry and semantics of a scene are inferred from a single monocular RGB image. Different from the SSC literature, relying on 2.5 or 3D input, we solve the complex problem of 2D to 3D scene reconstruction while jointly inferring its semantics. Our framework relies on successive 2D and 3D UNets, bridged by a novel 2D-3D features projection inspired by optics, and introduces a 3D context relation prior to enforce spatio-semantic consistency. Along with architectural contributions, we introduce novel global scene and local frustums losses. Experiments show we outperform the literature on all metrics and datasets while hallucinating plausible scenery even beyond the camera field of view. Our code and trained models are available at https://github.com/cv-rits/MonoScene.

1. Introduction

Estimating 3D from an image is a problem that goes back to the roots of computer vision [54]. While we, humans, naturally understand a scene from a single image, reasoning all at once about geometry and semantics, this was shown remarkably complex by decades of research [57, 75, 80]. Subsequently, many algorithms use dedicated depth sensors such as Lidar [36, 50, 62] or depth cameras [2, 15, 19], easing the 3D estimation problem. These sensors are often more expensive, less compact and more intrusive than cameras which are widely spread and shipped in smartphones, drones, cars, etc. Thus, being able to estimate a 3D scene from an image would pave the way for new applications.

3D Semantic Scene Completion (SSC) addresses scene understanding as it seeks to jointly infer its geometry and semantics. While the task gained popularity recently [56], the existing methods still rely on depth data (i.e. occupancy grids, point cloud, depth maps, etc.) and are custom designed for either indoor or outdoor scenes.

Here, we present MonoScene which – unlike the literature – relies on a single RGB image to infer the dense 3D voxelized semantic scene working indifferently for indoor and outdoor scenes. To solve this challenging problem, we project 2D features along their line of sight, inspired by optics, bridging 2D and 3D networks while letting the 3D network self-discover relevant 2D features. The SSC literature mainly relies on cross-entropy loss which considers each voxel independently, lacking context awareness. We instead propose novel SSC losses that optimize the semantic distribution of group of voxels, both globally and in local frustums. Finally, to further boost context understanding, we design a 3D context layer to provide the network with a global receptive field and insights about the voxels semantic relations. We extensively tested MonoScene on indoor and outdoor, see Fig. 1, where it outperformed all comparable baselines and even some 3D input baselines. Our main contributions are summarized as follows.

• MonoScene: the first SSC method tackling both outdoor and indoor scenes from a single RGB image.
• A mechanism for 2D Features Line of Sight Projection bridging 2D and 3D networks (FLoSP, Sec. 3.1).
• A 3D Context Relation Prior (3D CRP, Sec. 3.2) layer that boosts context awareness in the network.
• New SSC losses to optimize scene-class affinity (Sec. 3.3.1) and local frustum proportions (Sec. 3.3.2).

2. Related works

3D from a single image. Despite early researches [31,57, 80], in the deep learning era the first works focused on single 3D object reconstruction with explicit [1, 11, 16, 23, 26,
We propose a 3D contextual component that leverages multi-contextual learning is shown to be beneficial in [71]. Self-attention in [24, 33] and global pooling in [72, 76]. Experiments with standard cross-entropy (L_{ce}), our Scene-Class Affinity loss (L_{scal}, Sec. 3.3.1) improves the global semantics (L_{sem}^{scal}) and geometry (L_{geo}^{scal}), and our Frustums Proportion loss (L_{fp}, Sec. 3.3.2) enforces class distribution in local frustums, providing supervision beyond occlusions.

3D semantic scene completion (SSC). SSCNet [59] first defined the ‘SSC’ task where geometry and semantics are jointly inferred. The task gained attention lately, and is thoroughly reviewed in a survey [56]. Existing works all use geometrical inputs like depth [12, 25, 39–42, 45], occupancy grids [13, 25, 55, 69] or point cloud [53, 81]. Truncated Signed Distance Function (TSDF) were also proved informative [6, 9, 10, 12, 20, 21, 41, 59, 64, 77, 79]. Among others originalities, some SSC works use adversarial training to guide realism [10, 64], exploit multi-task [6, 38], or use lightweight networks [40, 55]. Of interest for us, while others have used RGB as input [6, 8, 9, 14, 20, 20, 25, 29, 39, 40, 42, 45, 81] it is always along other geometrical input (e.g. depth, TSDF, etc.). A remarkable point in [56] is that existing methods are designed for either indoor or outdoor, performing suboptimally in the other setting. The same survey highlights the poor diversity of losses for SSC. Instead, MonoScene solves voxel-wise SSC from a single RGB image \( x^{rgb} \), learning \( \hat{y} = f(x^{rgb}) \). This is significantly harder due to the complexity of recovering 3D from 2D. Our pipeline in Fig. 2 uses 2D and 3D UNets bridged by our Features Line of Sight Projection module (FLoSP, Sec. 3.1), lifting 2D features to plausible 3D locations, that boosts information flow and enables 2D-3D disentanglement. Inspired by [71], we capture long-range semantic context with our 3D Context Relation Prior component (3D CRP, Sec. 3.2) inserted between the 3D encoder and decoder. To guide the SSC training, we introduce new complementary losses. First, a Scene-Class Affinity Loss (Sec. 3.3.1) optimizes the intra-class and inter-class scene-wise metrics. Second, a Frustum Proportion Loss (Sec. 3.3.2) aligns the classes distribution in local frustums, which provides supervision beyond scene occlusions.

2D-3D backbones. We rely on consecutive 2D and 3D UNets with standard skip connections. The 2D UNet bases on a pre-trained EfficientNetB7 [61] taking as input the image \( x^{rgb} \). The 3D UNet is a custom shallow encoder-decoder with 2 layers. The SSC output \( \hat{y} \) is obtained by processing the 3D UNet output features with our completion head holding a 3D ASPP [7] block and a softmax layer.

3.1. Features Line of Sight Projection (FLoSP)
Lifting 2D to 3D is notoriously ill-posed due to the scale ambiguity of single view point [22]. We rather reason from optics and backproject multiscale 2D features to all possible 3D correspondences, that is along their optical ray, aggregated in a unique 3D representation. Our intuition here is that processing the latter with a 3D network will provide

\[ L_{rel} \]
guidance from the ensemble of 2D features. Our projection mechanism is akin to [52] but the latter projects each 2D map to a given 3D map – acting as 2D-3D skip connections. In contrast, our component bridges the 2D and 3D networks by lifting multiscale 2D features to a single 3D feature map. We argue this enables 2D-3D disentangled representations, providing the 3D network with the freedom to use high-level 2D features for fine-grained 3D disambiguation. Compared to [52], ablation in Sec. 4.3 shows our strategy is significantly better.

Our process is illustrated in Fig. 3. In practice, assuming known camera intrinsics, we project 3D voxels centroids (x^v) to 2D and sample corresponding features from the 2D decoder feature map F^2_{\text{3D}} of scale 1:s. Repeating the process at all scales S, the final 3D feature map F^3_{\text{3D}} writes

$$F^3_{\text{3D}} = \sum_{s \in S} \Phi_{\rho}(x^v(F^2_{\text{3D}})),$$

where \Phi_{\rho}(b) is the sampling of b at coordinates a, and \rho(\cdot) is the perspective projection. In practice, we backproject from scales S={1, 2, 4, 8}, and apply a 1x1 conv on 2D maps before sampling to allow summation. Voxels projected outside the image have their feature vector set to 0. The output map F^3_{\text{3D}} is used as 3D UNet input.

3.2. 3D Context Relation Prior (3D CRP)

Because SSC is highly dependent on the context [56], we inspire from CPNet [71] that demonstrates the benefit of binary context prior for 2D segmentation. Here, we propose a 3D Context Relation Prior (3D CRP) layer, inserted at the 3D UNet bottleneck, which learns n-way voxel-\leftrightarrow-voxel semantic scene-wise relation maps. This provides the network with a global receptive field, and increases spatio-semantic awareness due to the relations discovery mechanism.

Because SSC is a highly imbalanced task, learning binary (i.e. n=2) relations as in [71] is suboptimal.

We instead consider n=4 bilateral voxel-\leftrightarrow-voxel relations, grouped into free and occupied corresponding to ‘at least one voxel is free’ and ‘both voxels are occupied’, respectively. For each group, we encode whether the voxels semantic classes are similar or different, leading to the 4 non-overlapping relations: \(M=\{f, f_\text{d}, o, o_\text{d}\}\). Fig. 4a illustrates the relations in 2D (see caption for colors meaning).

As voxels relations are greedy with \(N^2\) relations for \(N\) voxels, we present the lighter supervoxel-\leftrightarrow-voxel relations.

Supervoxel-\leftrightarrow-Voxel relation. We define supervoxels as non-overlapping groups of \(s^3\) neighboring voxels each, and learn the smaller supervoxel-\leftrightarrow-voxel relation matrices of size \(N^2/s^2\). Considering a supervoxel \(V\) having voxels \(\{v_1, \ldots, v_{s^3}\}\) and a voxel \(v\), there are \(s^3\) pairwise relations \(v_1 \leftrightarrow v, \ldots, v_{s^3} \leftrightarrow v\). Instead of regressing the complex count of \(M\) relations in \(V\leftrightarrow v\), we predict which of the \(M\) relations exist, as depicted in Fig. 4b. This writes:

$$V \leftrightarrow v = \{v_1 \leftrightarrow v, \ldots, v_{s^3} \leftrightarrow v\}_{\neq},$$

where \{\} \neq returns distinct elements of a set.

3D Context Relation Prior Layer. Fig. 5 illustrates the architecture of our layer. It takes as input a 3D map of spatial dimension \(H\times W \times D\), on which is applied a series of ASPP convolutions [7] to gather a large receptive field, then split into \(n=|M|\) matrices of size \(HWD \times \frac{HWD}{s^2}\).

Each matrix \(A^m\) encodes a relation \(m \in M\), supervised by its ground truth \(A^m\). We then optimize a weighted multi-label binary cross entropy loss:

$$L_{\text{rel}} = - \sum_{m \in M} \sum_{i} (1-A_i^m)^{w_m} A_i^m \log \hat{A}_i^m,$$

where \(i\) loops through all elements of the relation matrix and \(w_m = \sum_i (1-A_i^m)^{w_m}\). The relation matrices are multiplied with reshaped supervoxels features to gather global context.

Alternatively, relations in \(A^m\) can be self-discovered (w/o \(M\)) by removing \(L_{\text{rel}}\), i.e. behaving as attention matrices.
3.3. Losses

We now introduce new losses pursuing distinct global (Sec. 3.3.1) or local (Sec. 3.3.2) optimization objectives.

3.3.1 Scene-Class Affinity Loss

We seek to explicitly let the network be aware of the global SSC performance. To do so, we build upon the 2D binary affinity loss in [71] and introduce a multi-class version directly optimizing the scene- and class-wise metrics.

Specifically, we optimize the class-wise derivable (P)recision, (R)ecall and (S)pecificity where $P_c$ and $R_c$ measure the performance of similar class $c$ voxels, and $S_c$ measures the performance of dissimilar voxels (i.e. not of class $c$). Considering $p_i$ the ground truth class of voxel $i$, and $\hat{p}_{i,c}$ its predicted probability to be of class $c$, we define:

$$P_c(\hat{p}, p) = \log \frac{\sum_i \hat{p}_{i,c}[p_i = c]}{\sum_i \hat{p}_{i,c}}$$

$$R_c(\hat{p}, p) = \log \frac{\sum_i \hat{p}_{i,c}[p_i = c]}{\sum_i [p_i = c]}$$

$$S_c(\hat{p}, p) = \log \frac{\sum_i (1 - \hat{p}_{i,c})(1 - [p_i = c])}{\sum_i (1 - [p_i = c])}$$

with $[]$ the Iverson bracket. For more generality, our loss $L_{scal}$ maximizes the above class-wise metrics with:

$$L_{scal}(\hat{p}, p) = -\frac{1}{C} \sum_{c=1}^{C} (P_c(\hat{p}, p) + R_c(\hat{p}, p) + S_c(\hat{p}, p)).$$

In practice, we optimize semantics $L_{scal} = L_{scal}(\hat{y}, y)$ and geometry $L_{geo} = L_{scal}(\hat{y}_{geo}, y_{geo})$, where $\{y, y_{geo}\}$ are semantic and geometric labels with respective predictions $\{\hat{y}, \hat{y}_{geo}\}$.

3.3.2 Frustum Proportion Loss

Disambiguation of occlusions is impossible from a single viewpoint and we observe that occluded voxels tend to be predicted as part of the object that shadows them. To mitigate this effect, we propose a Frustum Proportion Loss that explicitly optimizes the class distribution in a frustum.

As illustrated in Fig. 6, rather than optimizing the camera frustum distribution, we divide the input image into $\ell \times \ell$ local patches of equal size and apply our loss on each local frustum (defined as the union of the individual pixels frustum in the patch). Intuitively, aligning the frustums distributions provide additional cues to the network on the scene visible and occluded structure, giving a sense of what is likely to be occluded (e.g. cars are likely to occlude road).

Given a frustum $k$, we compute $P_k$ the ground truth class distribution of voxels in $k$, and $P_{k,c}$ the proportion of class $c$ in $k$. Let $\hat{P}_k$ and $\hat{P}_{k,c}$ be their soft predicted counterparts, obtained from summing per-class predicted probabilities. To enforce consistency, we compute $L_{fp}$ as the sum of local frustums Kullback-Leibler (KL) divergence:

$$L_{fp} = \sum_{k=1}^{\ell^2} D_{KL}(P_k||\hat{P}_k) = \sum_{k=1}^{\ell^2} \sum_{c \in C_k} P_k(c) \log \frac{P_k(c)}{\hat{P}_k(c)}.$$  \hspace{1cm} (8)

Note the use of $C_k$ instead of $C$. Indeed, frustums include small scene portions where some classes may be missing, making KL locally undefined. Instead, we compute the KL on $C_k$, the ground truth classes that exist in the frustum $k$.

3.4. Training strategy

MonoScene is trained end-to-end from scratch by optimizing our 4 losses and the standard cross-entropy ($L_{ce}$):

$$L_{total} = L_{ce} + L_{rel} + L_{scal} + L_{geo} + L_{fp}.$$  \hspace{1cm} (9)

Because real-world data comes with sparse ground truth $y$ due to occlusions, the losses are computed only where $y$ is defined [45, 56, 59]. Ground truths $y_{geo}$ and $A^m$, for $L_{geo}$ and $L_{rel}$, respectively, are simply obtained from $y$. We employ class weighting for $L_{ce}$ following [9, 55].
4. Experiments

We evaluate MonoScene on popular real-world SSC datasets being, indoor NYUv2 [58] and outdoor SemanticKITTI [3]. Because we first address 3D SSC from a 2D image, we detail our non-trivial adaptation of recent SSC baselines [9, 39, 40, 45] (Sec. 4.1) and then detail our performance (Sec. 4.2) and ablations (Sec. 4.3).

Datasets. NYUv2 [58] has 1449 Kinect captured indoor scenes, encoded as 240x144x240 voxel grids labeled with 13 classes (11 semantics, 1 free, 1 unknown). The input RGBD is 640x480. Similar to [9, 39, 40, 45] we use 795/654 train/test splits and always evaluate at cam2 of size 1226x370, left cropped to 1220x370. We use RGB image of cam2 of size 1226x370, left cropped to 1220x370. We use the official 3834/815 train/val splits and always evaluate at the scale 1:4.

SemanticKITTI holds outdoor Lidar scans voxelized as 256x256x32 grid of 0.2m voxels, labeled with 21 classes (19 semantics, 1 free, 1 unknown). We use RGB image of cam2 of size 1226x370, left cropped to 1220x370. We use the official 3834/815 train/val splits and evaluate on the test set at the scale 1:4.

Training setup. Unless otherwise mentioned, we use FLoSP at scales (1,2,4,8), 4 supervised relations for full scale (i.e. the official 3834/815 train/val splits and always evaluate at cam2 of size 1226x370, left cropped to 1220x370. We use a 256x256x32 grid of 0.2m voxels, labeled with 21 classes (11 semantics, 1 free, 1 unknown). The input RGBD is 640x480. Similar to [9, 39, 40, 45] we use 795/654 train/test splits and evaluate on the test set at the scale 1:4.

SemanticKITTI holds outdoor Lidar scans voxelized as 256x256x32 grid of 0.2m voxels, labeled with 21 classes (19 semantics, 1 free, 1 unknown). We use RGB image of cam2 of size 1226x370, left cropped to 1220x370. We use the official 3834/815 train/val splits and always evaluate at full scale (i.e. 1:1). Main results are from the hidden test set (online server), and ablations are from the validation set.

Metrics. Following common practices, we report the intersection over union (IoU) of occupied voxels, regardless of their semantic class, for the scene completion (SC) task and the mean IoU (mIoU) of all semantic classes for the SSC task. Note the strong interaction between IoU and the mean IoU (mIoU) of all semantic classes for the scene completion (SC) task.

4.1. Baselines

We consider 4 main SSC baselines among the best open-source ones available – selecting two indoor-designed methods, 3DSketch [9] and AICNet [39], and two outdoor-designed, LMSCNet [55] and JS3CNet [69]. We also locally compare against S3CNet [12], Local-DIFs [53], CoReNet [52]. We evaluate baselines in their 3D-input version and main baselines also in an RGB-inferred version.

RGB-inferred baselines. Unlike us, all baselines need a 3D input e.g. occupancy grid, point cloud or depth map, giving them an unfair geometric advantage. For fair comparison, we adapt main baselines to infer their 3D inputs directly from the 2D image (i.e. relying on the best found methods –, coined as ‘RGB-inferred’, denoted with a superscript, e.g. AICNetrgb). Note that baselines are unchanged. Infered 3D inputs are denoted with a hat, e.g. \( \hat{x}_{\text{depth}} \).

We use the pretrained AdaBin [4] to infer a depth map \( \hat{x}_{\text{depth}} \) serving as input for AICNetrgb. Using intrinsic calibration, we further converted depth to TSDF \( \hat{x}_{\text{TSDF}} \) with [74] for 3DSketchrgb input, and unproject depth to get a point cloud \( \hat{x}_{\text{pts}} \) directly used as input for JS3CNetrgb or discretized as occupancy grid \( \hat{x}_{\text{occ}} \) input for LMSCNetrgb. For training only, JS3CNetrgb also requires a semantic point cloud \( \hat{x}_{\text{sem pts}} \), obtained by augmenting \( \hat{x}_{\text{pts}} \) with 2D semantic labels from a pretrained network [82].

4.2. Performance

4.2.1 Evaluation

Tab. 1 reports performance of MonoScene and RGB-inferred baselines for NYUv2 (test set) and SemanticKITTI official benchmark (hidden test set). The low numbers for all methods advocate for task complexity.

On both datasets we outperform all methods by a significant mIoU margin of +4.03 on NYUv2 (Tab. 1a) and +2.11 on SemanticKITTI (Tab. 1b). Importantly, the IoU is improved on par (+3.87 and +0.16) which demonstrates our network captures the scene geometry while avoiding naively increasing the mIoU by lowering the IoU. On individual classes, MonoScene performs either best or second, excelling on large structural classes for both datasets (e.g. floor, wall; road, building). On SemanticKITTI we get outperformed mostly on small moving objects classes (car, motorcycle, person, etc.) which we ascribe to the aggregation of moving objects in the ground truth, highlighted in [53, 56]. This forces to predict the individual object’s motion which we argue is eased when using a 3D input.

Qualitative. We compare our SSC outputs in Fig. 7 showing the input image (leftmost column) and its corresponding camera frustum in ground truth (rightmost). Notice the noisy labels in NYUv2 having missing objects (e.g. windows, rows 2; ceiling, row 3), and in SemanticKITTI having sparse geometry (e.g. holes in buildings, rows 1–3).

On indoor scenes (NYUv2, Fig. 7a), all methods correctly capture global scene layouts though only MonoScene recovers thin elements as table legs and cushions (row 1), or the painting frame and properly sized TV (row 2). On complex cluttered outdoor scenes (SemanticKITTI, Fig. 7b), compared to baselines, MonoScene evidently cap-
4.2.2 Comparison against 2.5/3D-input baselines

For completeness, we also compare with some original baselines (i.e., using real 3D input) in Tab. 2. Despite the unfair setup since we use only RGB, MonoScene still outperforms the mIoU of some indoor baselines.

Table 2. 2.5/3D input baselines. Despite a single RGB, MonoScene still outperforms the mIoU of some indoor baselines.

Table 3. Architecture ablation. Our components boost performance on NYUv2 [58] (test set) and SemanticKITTI [3] (val. set).

Table 1. Performance on (a) NYUv2 [58] and (b) SemanticKITTI [3]. We report the performance on semantic scene completion (SSC - mIoU) and scene completion (SC - IoU) for RGB-inferred baselines and our method. Despite the various indoor and outdoor setups, we significantly outperform other RGB-inferred baselines, in both mIoU and IoU.
Figure 7. Outputs on (a) NYUv2 [58] and (b) SemanticKITTI [3]. In both, the input is shown left and the camera viewing frustum is shown in the ground truth (rightmost) with darker colors being parts of scenes unseen by the image in (b). MonoScene better captures the scene layout on both datasets. On indoor scene (a), it reconstructs thin objects like table legs (row 1), painting and TV (row 2), while in outdoor (b), it better estimates occluded geometry e.g. car (row 1–3) and better hallucinates the scenery beyond the field of view (row 1–4).

4.3. Ablation studies

We ablate our MonoScene framework on both NYUv2 (test set) and SemanticKITTI (validation set), and report the average of 3 runs to account for training variability.

Architectural components. Tab. 3 shows that all components contribute to the best results. For ‘w/o FLoSP’, we instead interpolate and convolve the 2D decoder features to the required 3D UNet input size. Specifically, FLoSP (Sec. 3.1) is shown to be the most crucial as it improves remarkably both semantics ([+12.83, +6.72] mIoU) and geometry ([+14.11, +9.56] IoU). 3D CRP (Sec. 3.2) contributes equally to IoU (in [+0.77, +1.12]) and mIoU (in [+0.54, +1.33]). Both SCAL losses (Sec. 3.3.1) contribute differently as expected, since \( L_{sem}^{scal} \) helps semantics (+1.61 mIoU in both), while \( L_{geo}^{scal} \) boosts geometry (+1.55, +2.20 IoU). In NYUv2 only, \( L_{sem}^{scal} \) harms IoU (-0.31) but improves the same metric on SemanticKITTI (+0.34). Finally, the frustums proportion loss (Sec. 3.3.2) boosts both metrics on both datasets by at least +0.38 and up to +0.61.

Effect of features projection. We now study in-depth the effect of FLoSP (Sec. 3.1) at the core our RGB-based task. In Tab. 4a, we use our FLoSP projecting only 2D features from given 2D scales by changing \( S \) in Eq. (1). More 2D...
Table 4. Study of FLoSP and 3D CRP. (a) Projecting from different 2D scales (S) in FLoSP (Sec. 3.1) show more scales is better. (b) In our 3D CRP (Sec. 3.2) using more relations (n) and supervision (L_{rel}) lead to higher metrics. Results are on NYUv2.

<table>
<thead>
<tr>
<th>2D scales (S)</th>
<th>IoU ↑</th>
<th>mIoU ↑</th>
</tr>
</thead>
<tbody>
<tr>
<td>1, 2, 4, 8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1, 2, 4</td>
<td>42.51 ±0.15</td>
<td>26.94 ±0.10</td>
</tr>
<tr>
<td>1, 2</td>
<td>42.08 ±0.19</td>
<td>26.28 ±0.24</td>
</tr>
<tr>
<td>1</td>
<td>41.56 ±0.16</td>
<td>25.66 ±0.21</td>
</tr>
<tr>
<td>2</td>
<td>41.51 ±0.15</td>
<td>25.61 ±0.43</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>n</th>
<th>L_{rel}</th>
<th>IoU ↑</th>
<th>mIoU ↑</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>✓</td>
<td>42.51 ±0.15</td>
<td>26.94 ±0.10</td>
</tr>
<tr>
<td></td>
<td>x</td>
<td>42.24 ±0.15</td>
<td>26.55 ±0.20</td>
</tr>
<tr>
<td>2</td>
<td>✓</td>
<td>42.09 ±0.15</td>
<td>26.63 ±0.09</td>
</tr>
<tr>
<td></td>
<td>x</td>
<td>42.15 ±0.10</td>
<td>26.47 ±0.06</td>
</tr>
</tbody>
</table>

Table 5. Frustums Proportion loss ablation. Varying the number of local frustums (ℓ×ℓ) in our loss shows more frustums (i.e., smaller) lead to finer guidance and better results on both datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>NYUv2</th>
<th>SemanticKITTI</th>
</tr>
</thead>
<tbody>
<tr>
<td>ℓ×ℓ</td>
<td>IoU ↑</td>
<td>mIoU ↑</td>
</tr>
<tr>
<td>8 × 8</td>
<td>42.51 ±0.15</td>
<td>26.94 ±0.10</td>
</tr>
<tr>
<td>4 × 4</td>
<td>42.52 ±0.12</td>
<td>26.85 ±0.19</td>
</tr>
<tr>
<td>2 × 2</td>
<td>42.41 ±0.13</td>
<td>26.85 ±0.22</td>
</tr>
<tr>
<td>1 × 1</td>
<td>42.39 ±0.18</td>
<td>26.52 ±0.33</td>
</tr>
<tr>
<td>w/o L_{fr}</td>
<td>41.90 ±0.26</td>
<td>26.37 ±0.16</td>
</tr>
</tbody>
</table>

5. Discussion

MonoScene tackles monocular SSC originally using successive 2D-3D UNets, bridged by a new features projection, with increased contextual awareness and new losses.

Limitations. Despite good results, our framework still struggles to infer fine-grained geometry, or to separate semantically-similar classes, e.g., car/truck or chair/sofa. It also performs poorly on small objects partly due to their scarcity (<0.3% in Sem.KITTI [3]). Due to the single viewpoint, occlusion artefacts such as distortions are visible along the line of sight in outdoor scenes. Additionally, as we exploit 2D-3D projection with the FLoSP module (Sec. 3.1), we evaluate the effect of inferring from datasets having various camera setups, showing in Fig. 9 that results — though consistent — have increasingly greater distortion when departing from the camera settings of the training set.

Broader impact, Ethics. Jointly understanding the 3D geometry and semantics from image paves ways for better mixed reality, photo editing or mobile robotics applications. But the inevitable errors in the scene understanding could have fatal issues (e.g., autonomous driving) and such algorithms should always be seconded by other means.

Acknowledgment This work used HPC resources from GENCI-IDRIS (Grant 2021-AD011012808). It was done in the SAMBA collaborative project, co-funded by BpiFrance in the Investissement d’Avenir Program.
References

[9] 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. 2, 4, 5, 6, 7
[10] Yueh-Tung Chen, Martin Garbade, and Juergen Gall. 3d semantic scene completion from a single depth image using adversarial training. In *ICIP*, 2019. 2
[14] Ian Cherabier, Johannes L. Schönberger, M. Oswald, M. Pollefeys, and Andreas Geiger. Learning priors for semantic 3D reconstruction. In *ECCV*, 2018. 2
[17] 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. 8
[34] Hamid Izadinia, Qi Shan, and Steven M. Seitz. Im2cad. In *CVPR*, 2017. 2


[38] Jie Li, Laiyan Ding, and Rui Huang. Imenet: Joint 3d semantic scene completion and 2d semantic segmentation through iterative mutual enhancement. In IJCAI, 2021. 2


[40] 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. 2, 4, 5

[41] Jie Li, Yu Liu, Xia Yuan, Chunxia Zhao, Roland Siegwart, Ian Reid, and Cesar Cadena. Depth based semantic scene completion with position importance aware loss. RA-L, 2020. 2

[42] Siqi Li, Changqing Zou, Yipeng Li, Xibin Zhao, and Yue Zhao. Attention-based multi-modal fusion network for semantic scene completion. In AAAI, 2020. 2


[56] Luis Roldão, Raoul De Charette, and Anne Verroust-Blondet. 3D Semantic Scene Completion: a Survey. IJCV, 2021. 1, 2, 3, 4, 5, 6

[57] Ashutosh Saxena, Min Sun, and Andrew Y Ng. Make3d: Learning 3d scene structure from a single still image. T-PAMI, 2008. 1


[68] Qianguang Xu, Weiyue Wang, Duygu Ceylan, Radomir Belongie, and Bharath Hariharan. Pointflow: 3d point cloud


