DOI: 10.1111/cgf.14662
Pacific Graphics 2022
N. Umetani, E. Vouga, and C. Wojtan
(Guest Editors)

*Volume 41 (2022), Number 7*

# Resolution-switchable 3D Semantic Scene Completion

Shoutong Luo[1] and Zhengxing Sun[†1] and Yunhan Sun [1] and Yi Wang [1]

[1]State Key Laboratory for Novel Software Technology, Nanjing University

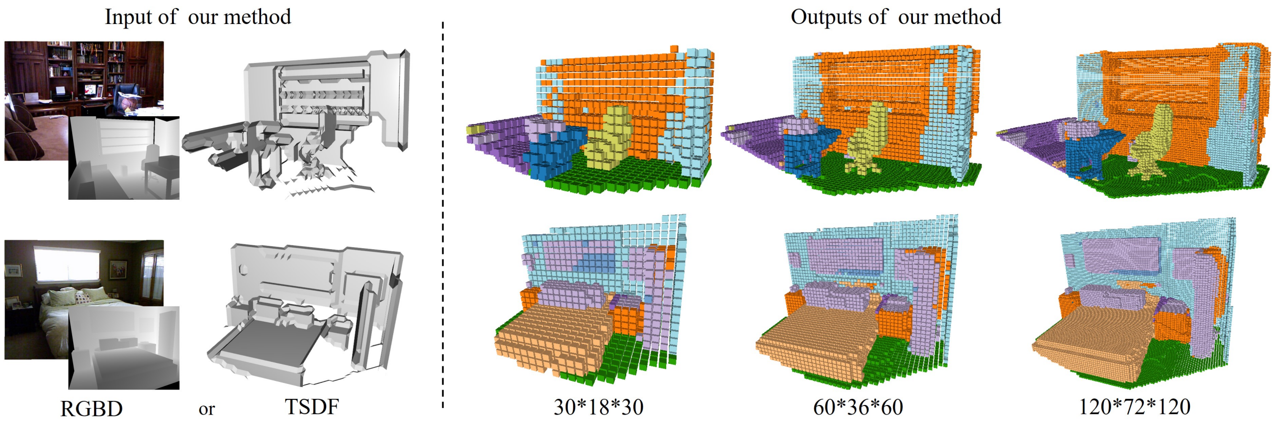


**Figure 1:** *Our method takes incomplete scene geometry (RGBD or TSDF) as input and outputs complete geometric and semantic results at multiple resolutions using only a single network. Our method doesn't need to be retrained for different resolutions. We are able to maintain consistent results over multiple resolutions.*

**Abstract**

*Semantic scene completion (SSC) aims to recover the complete geometric structure as well as the semantic segmentation results from partial observations. Previous works could only perform this task at a fixed resolution. To handle this problem, we propose a new method that can generate results at different resolutions without redesigning and retraining. The basic idea is to decouple the direct connection between resolution and network structure. To achieve this, we convert feature volume generated by SSC encoders into a resolution adaptive feature and decode this feature via point. We also design a resolution-adapted point sampling strategy for testing and a category-based point sampling strategy for training to further handle this problem. The encoder of our method can be replaced by existing SSC encoders. We can achieve better results at other resolutions while maintaining the same accuracy as the original resolution results. Code and data are available at https://github.com/lstcutong/ReS-SSC.*

**CCS Concepts**

• ***Computing methodologies*** → ***Volumetric models;***

## 1. Introduction

Given partial observation of scenes, 3D Semantic Scene Completion (SSC) aims to simultaneously infer the complete scene structure and perform semantic segmentation of the scene [SYZ*17]. This ability to understand and infer complete scene structure is very helpful for many realistic applications, including robotics, virtual reality and interior design. However, the difference in output between different application devices lies in the different resolutions required, (i.e. resolution-switchable). Therefore, how to dynamically and seamlessly switch between various resolutions remains a challenging problem in this area.

Previous work [SYZ*17, GT18, ZWZ*18, GCSG19] voxelized 3D scenes to perform SSC, typically using a series of 3D convo-

[†] *Corresponding author: szx@nju.edu.cn

lutions to process voxelized scenes. However, these voxel-based methods still suffer from the low resolution of results due to the curse of dimensionality [WY10]. Therefore, it is difficult for their methods to generate high-resolution results. In recent years, some methods have mitigated the curse of dimensionality by reducing the network parameters. It allows their methods to produce higher resolution as well as low resolution of results. For example, DDRNet [LLG*19] proposes a Dimensional Decomposition Residual(DDR) network that reduces the number of parameters required for 3D feature processing. CCPNet [ZLL*19] uses group convolution that further reduces the parameters. However, their methods can only generate fixed resolution rather than dynamically switchable.

We believe that the key to the inability of the above methods to do that lies in the high coupling between resolution and network structure. For example, voxel-based approaches [LLG*19, LHW*20, LLY*20] typically employ a series of stacked 3D convolutional networks to map the input to feature volume, which is then decoded by 3D channel convolution into voxel-by-voxel category labels. In this process, the resolution of the final result is determined by the feature volume, which is determined when the network design is completed. Therefore, to produce a different resolution of result, the network will inevitably have to be redesigned and retrained.

In fact, the feature of each voxel can be seen as global features of a 3D space occupying a certain volume, and the decoding process can be seen as global decoding of that space. This leads to a discretization of the whole process, resulting in a coupling between resolution and network structure. Inspired by neural implicit representation [PNM*20], we propose to decouple the direct connection between resolution and network structure. This allows us to generate results at multiple resolutions without retraining our network. Experiments show that our method can achieve similar accuracy to existing SSC methods at 60*36*60 resolution and surpass them at other resolutions. Ablation studies show the effectiveness of decoupling. Our contributions are as follows:

- We propose a resolution-switchable 3D semantic scene completion method. It decouples the direct connection between resolution and network structure with three key designs: i) resolution adaptive feature generation. ii) Resolution-adapted point sampling strategy for testing and iii) category-based training sample generation for training.
- We introduce a distance-based interpolation method that converts the feature volume generated by SSC encoders into a continuous feature representation (the RAF), it allows us to calculate the feature of any point in the 3D space. During the test, we adopt a dynamic point sampling strategy according to the final resolution of results, it allows us to achieve good results at different resolutions. Finally, during training, we sample training points category by category, it provides a better supervision signal for our network.
- The encoder of our method can be replaced with existing SSC encoders. In the future, if any better network is proposed, the encoder can be directly used as ours.

## 2. Related Work

### 2.1. Semantic Scene Completion

Scene semantic completion aims to simultaneously recover the complete geometric structure of scene observations as well as the semantic segmentation results. SSCNet [SYZ*17] first proposed this task by coupling geometric completion and semantic segmentation in an end-to-end network in order to make them mutually reinforcing and many subsequent works [GT18, ZWZ*18, LHZ*18, GCSG19, DdCKH19, CGG19, WTNT19,CHY19,LLY*20,LLY*19,CLQ*20,LHW*20] have followed this paradigm. The expensive cost of 3D CNN, however, limits the resolution of the final reconstruction results.

To generate higher resolutions, the high cost of 3D convolution needs to be alleviated first. Existing methods can be divided into two categories to alleviate the problem. Some of these methods try to adapt to higher resolutions by reducing the number of network parameters, starting with the network structure. For example, EfficientNet [ZZY*18] introduces sparse convolution, which greatly reduces the computational cost by computing only the non-zero values of voxels. DDRNet [LLG*19] introduced a dimensional decomposition residual network for the 3D SSC task. They decompose a $k \times k \times k$ convolution into $1 \times 1 \times k, 1 \times k \times 1, k \times 1 \times 1$ convolutions, reducing the number of parameters needed for 3D feature processing and making the network more lightweight. CCPNet [ZLL*19] uses group convolutional networks and a reduced number of feature channels to reduce the number of network parameters further. Although these three methods have the ability to generate higher resolutions, their methods are designed for a fixed resolution. Without redesigning and retraining, their methods aren't able to generate results at another resolution. It is worth noting that CCPNet [ZLL*19] is resolution-independent because they use a U-shape network structure and their output can maintain the resolution of the original input, but unlike our method, their method requires retraining for different resolutions, but ours does not.

Some other methods attempt to use a different 3D representation like point cloud [ZLHQ21, TCWZ21] to alleviate the problem. [ZLHQ21] proposes a point-based SSC method, which calculates the position of a new point by predicting the position offset of an existing point and concatenates it with the original point to obtain the complete result. [TCWZ21] also proposes a point-based method, in which they first turn TSDF into a series of point sets, then classify the point sets by Pointnet++ [QYSG17], and finally turn the point results into voxel results. Our method can also be seen as a point-based method, the difference is that their method is limited by the capacity of the network and can only obtain a more sparse point cloud, while our method can theoretically predict the label of any point in the 3D space, i.e. an infinite number of points.

### 2.2. 3D Implicit Representation

Implicit representation models 3D structures by implicit functions. In Poisson reconstruction [KBH06], the concept of indicator function was proposed. The indicator function outputs 1 for a point if it is inside the object, and 0 otherwise. With the development of deep learning, researchers have tried to fit implicit functions using neural networks. [MON*19] proposed an OCCNet to predict

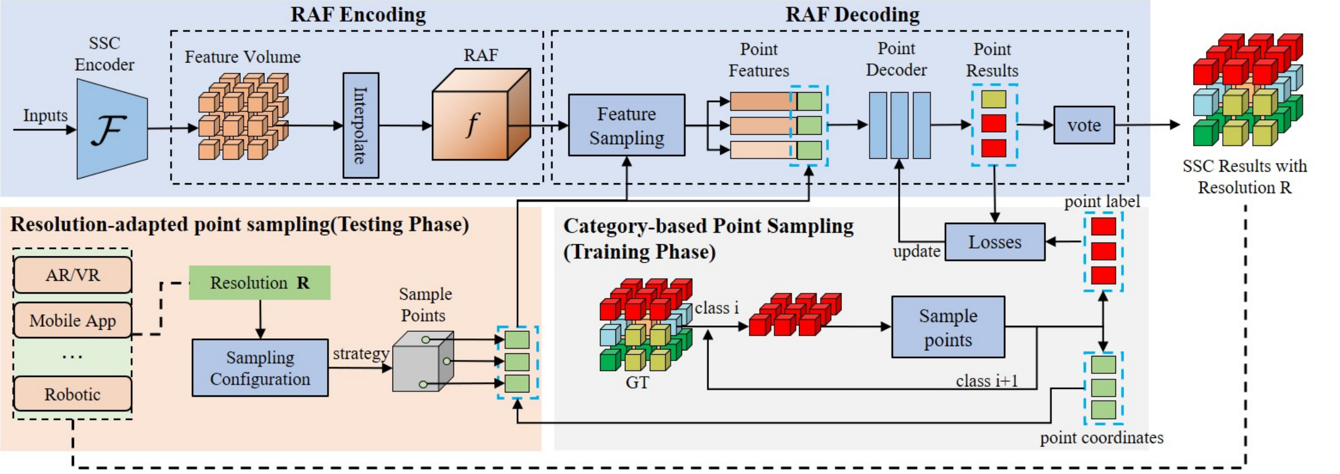


**Figure 2:** *Overview of our method. Our method uses existing SSC encoders to map RGBD or TSDF into a feature volume. Then, we decouple the direct connection between resolution and network structure with the following designs: 1) Resolution Adaptive Feature Encoding Module which takes feature volume as input and outputs a resolution adaptive feature(RAF). 2) RAF Decoding Module that decodes RAF via points. 3) Resolution-adapted point sampling module that adaptively use different sampling strategy to generate a voxel model according to the final resolution during the test. 4) Category-based point sampling module that generates training data for our network. Our approach is resolution-switchable and can be adapted to different applications and devices.*

the occupancy of each point in the space, and [Che19, PFS*19] proposes to predict the SDF value of each point. However, these methods usually represent the geometry as a single feature vector, which cannot be used in the scene-level reconstruction. Therefore, in recent years, researchers have focused on how to use implicit representation to perform scene-level reconstruction tasks. The basic idea of most of the work is to encode the scene locally and individually rather than globally for the whole scene [JSM*20, PNM*20, CLI*20]. In this regard, our work is more similar to COCCNet [PNM*20], where they propose a voxel form of feature encoding and use local features to decode the occupancy of each point in the space. Unlike their approach, first, their input is a complete 3D scene geometry, while our input 3D scene geometry is incomplete. Second, their network learns a binary division of the space (inside or outside), while our network learns a more fine-grained semantic division of the space (class).

## 3. Method

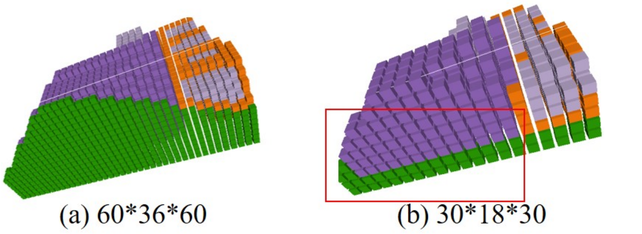
Our goal is to seamlessly and dynamically switch the resolution of results to meet the different requirements of different applications using a single network and without retraining. For example, for some mobile applications, storage and computational performance limitations make these applications only support coarser models. While for some gaming and VR applications, fine-grained geometric models are usually required. To achieve this, we propose our resolution-switchable semantic scene completion method by decoupling the direct relation between resolution and network structure. Figure 2 shows the overview of our method. The next sections will give comprehensive descriptions of every key design.

### 3.1. Resolution Adaptive Feature Generation

In this subsection, we will first describe the general process of SSC. Then we will introduce our first key step in decoupling the resolution and network - converting the feature volume generated by SSC encoders into a **r**esolution **a**daptive **f**eature(RAF).

**SSC encoding** There are two types of input for semantic scene completion, one is 3D input such as TSDF, and the other is 2D input such as RGBD. Previous methods [SYZ*17, GCSG19, LLY*19] encode 3D inputs into 3D feature volumes using 3D convolution or 3D dilation convolution and decode them voxel-by-voxel. As for 2D inputs, a 2D to 3D feature projection process is usually involved before using 3D convolution [LLG*19, LHW*20, LLY*20]. This process can be summarized as $y = \mathcal{D}(\mathcal{F}(I))$. The input $I$ (TSDF, RGBD) is first mapped by the encoder $\mathcal{F}$ to the feature volume $V \in R^{h\times w\times d\times c}$ where $c$ represents the dimension of the feature. Then, $V$ is decoded into voxel-vise category label scores $y \in R^{h\times w\times d\times K}$, where $K$ represents the number of categories. $D$ usually consists of multiple 3D convolutions with a kernel size of $1\times 1\times 1$. Our method directly borrows the encoders of these methods to map RGBD or TSDF to feature volume $V$. To further improve the quality of the feature volume, we also concatenate the voxel-vise label score $y$ with the feature volume. We use the pseudo-code `torch.cat([V,y],dim=3)` to calculate the new feature volume $V$. The final shape of the tensor $V$ is $h \times w \times d \times (c+K)$. This new feature volume is used to generate our resolution adaptive feature.

**RAF Encoding** Feature volume mentioned above is resolution-dependent, it cannot be adapted to the needs of results with different resolutions. To solve this problem, we propose to use an interpolation function to assign a feature to each point in the space. Few interpolation functions can be chosen. For example, the most used trilinear interpolation, see Figure 4(a). Here, we introduce another interpolate function, i.e. distance-based interpolation to generate RAF. Let $\{x_i | i = 1,\ldots,8\}$ be the spatial 8-neighborhood points of point $x$, which are located on the 8 lattice points of the voxel. Let



(a) 60*36*60

(b) 30*18*30

**Figure 3:** *An illustration of 'preference'. When the resolution is reduced to a very low level (see b), the space occupied by a voxel may contain multiple geometries (in the red box, part of the floor (green) and part of the sofa (purple)). At this time, the semantic of a voxel often has some "preference". In the example, the algorithm tends to attribute part of the geometry to the sofa and part of the geometry to the floor. Note when the resolution is high, some voxels may also contain multiple geometries. But usually, these voxels occupy only a small part of the whole and are therefore negligible. So, in order to better determine the semantic label of each voxel at low resolution, we design a resolution-adapted sampling strategy. See Section 3.1 for more details.*

$\{V(x_i)|i=1,...,8\}$ be the features of the 8-neighborhood points, then the feature at any point $x$ in the space is

$$f(x)=\sum_{i=1}^{8}(1-\frac{d_i}{\sum_{i=1}^{8}d_i})V(x_i) \quad (1)$$

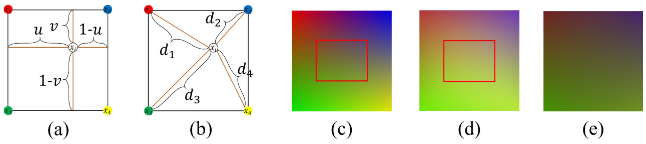
where $d_i$ denotes the Euclidean distance from point x to point $x_i$. The above equation shows that the feature of $x$ is jointly determined by the features of its 8 neighboring points, and more features are contributed by the points closer to $x$. In this equation, the feature weight $w_i$ for each lattice point $x_i$ is $1-d_i/\sum_{i=1}^{8}d_i$. Intuitively, we should normalize these 8 weights such that $\sum_{i=1}^{8}w_i=1$, just as the weights in the trilinear interpolation function. However, we find that the magnitude of features obtained by the interpolation with weight normalization is relatively small, which leads to a small drop in performance, see Figure 4(d),(e), and Tabel 4 for comparison.

Compared with the trilinear interpolation function, our distance-based interpolation function obtain features with larger magnitude in the center region of a voxel. We believe this helps decide each voxel's label during test time since we sample points at the center of the voxels. See Figure 4(c),(d), and Tabel 4 for comparison.

In conclusion, RAF can be seen as a continuous feature representation, which makes RAF itself have the adaptability to the results at different resolutions.

In practice, we do not precompute the feature of each point in the space and store them, because the number of points is infinite. However, Equation 1 still defines what the feature of each point in the space is. Therefore, in the implementation, we put the process of point feature calculation into the feature sampling of the RAF decoding.

**RAF Decoding** Although the RAF is a continuous representation, it can be seen as a collection of features consisting of an infinite number of points in the 3D space. Therefore, decoding the RAF only requires decoding each individual point in the space. The feature of each point is given by Equation 1.



(a) (b) (c) (d) (e)

**Figure 4:** *Possible choices of interpolation function and their results. (a) bilinear interpolation for 2D and trilinear for 3D. (b) our distance-based interpolation. (c) result of the bilinear interpolation function. (d) result of distance-based interpolation without weight normalization. (e) result of distance-based interpolation with weight normalization. In the center region, the features obtained by our interpolation method have a larger magnitude, which will help in the decoding process. Because for R2 and R3 resolution decoding, we simply sample points at the center of the voxel.*

Given a point $x$ in 3D space, we sample its feature $f(x)$ from RAF using Equation 1. Then, we introduce a point decoder $\chi_S$ that takes both as inputs and outputs the point label:

$$\chi_S(x,f(x))\rightarrow[0,1]^K \quad (2)$$

where $K$ is the number of classes. Class 1 represents the empty space, the other classes represent objects of different categories. We implement a small fully-connected network that comprises multiple ResNet blocks similar to [PNM*20]. Different from theirs, our output channel is $K$ and we use fewer layers than theirs since our training data is few.

### 3.2. Resolution-adapted point sampling

In this subsection, we present how to generate a voxel result for a given resolution during the test, which is the second key step in decoupling the resolution and network.

Given a target resolution $R$, this resolution can be configured from a variety of downstream applications such as VR, mobile and gaming apps, we aim to generate a discrete 3D model for these applications. It is intuitive to sample points for each voxel and then decide on each voxel's label by voting. However, how to sample points and how many points to sample still need to be discussed.

We propose two sampling strategies, one is uniform sampling and another is center sampling. The uniform sampling uniformly samples $N(N>1)$ points in each voxel while center sampling samples 1 point at the center of each voxel. The question is what is the difference between these two strategies and how do we choose between them?

When the resolution is very low, the space occupied by a voxel can be very large, and center sampling may not reflect well the class distribution of that space. For example, a voxel may contain both parts of the table legs and part of the floor, and a fixed point at the center of the voxel may cause "preference", see Figure 3 for more details. When the resolution is high, the 3D space occupied by a voxel is small. At this time, a voxel can already be seen as a

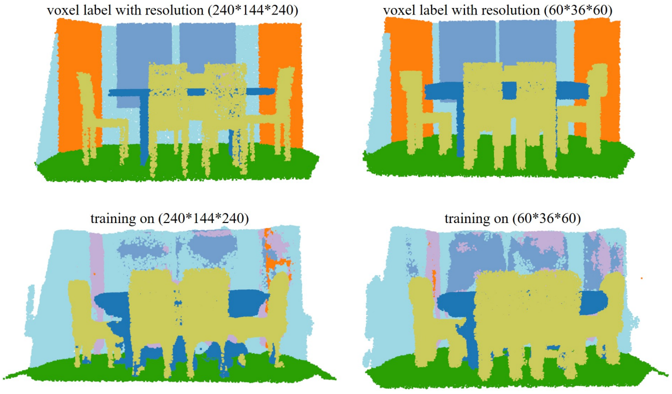

voxel label with resolution (240*144*240)
voxel label with resolution (60*36*60)
training on (240*144*240)
training on (60*36*60)

**Figure 5:** *We visualize the semantic space obtained with voxel-wise labels with different resolutions to illustrate the choice of our training data. The top row shows the semantic space that the GT label can provide. It is easy to see that the high-resolution voxel-wise label provides more detailed semantic boundaries. As a result, the semantic space trained from the high-resolution voxel-wise label (second row) is also more detailed. Since our method can be seen as a voxelization process for this semantic space, a more detailed semantic space is beneficial for the subsequent generation of voxel-like results with different resolutions.*

good approximation to the continuous space, center sampling and uniform sampling should make little difference. However, in experiments, we show that applying uniform sampling to high resolutions may cause a little drop in performance. Meanwhile, applying uniform sampling to high resolutions will greatly increase the computing cost since the number of points will be large. With the above considerations, we suggest that for low resolution, using the uniform sampling strategy and eventually deciding the label of each voxel by voting is a better choice. For high resolution, using the center sampling strategy and assigning the label of that point directly to that voxel is a better choice.

### 3.3. Label Generation and Training

In this subsection, we will discuss how to generate training data for our point decoder since we only have voxel-wise labels and how to train it.

**Category-based Point Sampling** Given a voxel-wise label $Y$, instead of using the sampling strategy that uniformly samples points in 3D space as mentioned in [MON*19, PNM*20], we propose a category-based label generation strategy. For each category, we sample points uniformly in the 3D space occupied by the voxels of all that categories and assign that category label to these points. We do this iteratively until all the categories are sampled. Then, the points of each category are randomly sampled such that the number of points of the empty category is twice the sum of the number of points of all non-empty categories, while the number of points of all non-empty categories is equal to each other. Compared with the sampling strategy used in [MON*19,PNM*20], the category-based sampling strategy obtains a more balanced point number over different categories and thus provides a better-supervised signal for our point decoder. In our experiments, we have a total number of 10240 training points per batch.

Since the resolution of our final results needs to be dynamically switched to the needs of the application, this means that our approach has to achieve good results at arbitrary resolutions. Although our continuous feature representation is theoretically capable of representing arbitrary resolution models, its performance is still limited by the resolution of the voxel-wise labels. The higher the resolution of the voxel-wise labels, the finer the semantic boundaries they can provide, and vice versa, the coarser they are. Finer semantic boundaries are beneficial for high-resolution results and are compatible with low-resolution results, so in our experiments, we use the labels with highest resolution available in the dataset for training. See Figure 5 for a visual explanation of this idea.

**Training** Our model contains two parts that need to be trained, one is the SSC encoder and the other is the point decoder. Our encoders can be replaced by existing SSC encoders. So, to fully exploit the capabilities of the encoder, we first pre-train SSC methods and later replace their decoder with our point decoder. Then we fix the encoder's parameters and only update the point decoder with the following two losses: one is semantic segmentation loss, which uses softmax cross-entropy loss to compute the difference between the predicted point labels and the true point labels:

$$\mathcal{L}_{sem} = -\sum_{c=1}^{K} w_c \hat{y}_{i,c} \log\left(\frac{e^{y_{i,c}}}{\sum_{c'}^{K} e^{y_{i,c'}}}\right) \tag{3}$$

where $\hat{y}_{i,c}$ are the binary ground truth vectors, i.e. $\hat{y}_{i,c} = 1$ if point $i$ is labeled by class $c$, $K$ is the number of classes, and $w_c$ is the loss weight. We set $w_c = 0.8$ for empty class and 1 for others. The other is a geometric loss to assist in the training of the point decoder. The geometric loss was first proposed in ForkNet [WTNT19]. The core idea is to separate geometric completion from semantic completion. This is because coupling geometric completion and semantic segmentation in one loss term makes optimization more difficult, while disentangling geometric completion separately can better assist in training the network so that the whole optimization process can converge faster. To achieve this, for a ground truth one-hot label vector $\hat{y}_i$, we create another all-zero two-dimensional vector $\hat{y}_i'$ and set the 1st dimension of $\hat{y}_i'$ to 1 if the 1st dimension of $\hat{y}_i$ is 1, otherwise the 1st dimension of $\hat{y}_i'$ is set to 0. For the point label vector $y_i$ predicted by the point decoder, we sum the values from the 2nd dimension to the K-th dimension of $y_i$ and obtain a new two-dimensional prediction vector $y_i'$. We additionally process $y_i'$ using softmax to ensure that the value of each dimension of $y_i'$ is between [0,1]. Then the geometric loss is obtained by calculating the binary cross-entropy loss of $\hat{y}_i'$ and $y_i'$:

$$\mathcal{L}_{geo} = -[w_1 \hat{y}_{i,1}' \log(y_{i,1}') + w_2(1-\hat{y}_{i,2}')\log(1-y_{i,2}'))] \tag{4}$$

We set $w_1 = 0.8$ and $w_2 = 1$. Finally, our overall loss is $\mathcal{L} = \mathcal{L}_{sem} + \lambda \mathcal{L}_{geo}$. In our experiments, we empirically set $\lambda = 2$.

## 4. Experiments

In this section, we first introduce the dataset used in our experiments and the evaluation metrics. Then, we perform comparison experiments with existing SSC methods at 3 different resolutions

**Table 1:** *Comparison experiments with AIC [LHW* 20] and DDR [LLG* 19] at three different resolutions on NYUCAD dataset.*

| | | SC | | | SSC | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | method | prec | recall | IoU | ceil. | floor | wall | win. | chair | bed | sofa | table | tvs | furn. | objs. | avg. |
| R1 | DDR | 85.3 | 83.5 | 72.8 | 36.9 | 70.0 | 39.2 | 5.88 | 31.5 | 54.9 | 42.6 | 30.7 | 14.2 | 33.2 | 20.1 | 34.5 |
| | Ours-DDR | 80.9 | 97.2 | 79.0 | 47.5 | 73.0 | 49.3 | 12.3 | 37.2 | 59.3 | 50.5 | 33.0 | 7.8 | 47.9 | 29.8 | 40.7 |
| | AIC | 83.4 | 85.9 | 73.2 | 43.5 | 70.1 | 39.9 | 8.81 | 35.1 | 55.7 | 46.4 | 33.8 | 11.2 | 36.7 | 23.2 | 36.8 |
| | Ours-AIC | 85.2 | 93.1 | 80.1 | 50.7 | 71.8 | 52.3 | 15.7 | 42.0 | 60.5 | 54.6 | 36.9 | 10.8 | 47.4 | 30.3 | 43.0 |
| R2 | DDR | 77.1 | 94.4 | 73.6 | 46.4 | 91.9 | 51.7 | 7.65 | 37.1 | 58.4 | 51.7 | 31.5 | 15.7 | 39.2 | 24.5 | 41.5 |
| | Ours-DDR | 81.6 | 91.8 | 75.9 | 45.1 | 86.4 | 53.7 | 11.5 | 38.2 | 60.5 | 54.5 | 32.3 | 7.8 | 48.5 | 30.4 | 42.6 |
| | AIC | 74.8 | 96.2 | 72.6 | 50.5 | 91.8 | 53.3 | 12.2 | 35.8 | 59.2 | 55.3 | 31.3 | 15.2 | 41.9 | 27.5 | 43.1 |
| | Ours-AIC | 82.7 | 90.4 | 75.9 | 50.9 | 86.2 | 54.4 | 14.2 | 41.6 | 60.2 | 59.3 | 34.5 | 11.0 | 47.4 | 31.4 | 44.7 |
| R3 | DDR | 61.5 | 94.5 | 59.2 | 34.5 | 47.9 | 37.4 | 6.66 | 29.2 | 56.7 | 49.6 | 25.5 | 14.3 | 37.9 | 22.0 | 32.9 |
| | Ours-DDR | 70.3 | 93.8 | 67.0 | 35.7 | 77.0 | 40.6 | 11.3 | 32.1 | 58.9 | 52.7 | 26.8 | 8.1 | 47.1 | 28.2 | 38.0 |
| | AIC | 59.8 | 96.3 | 58.3 | 35.1 | 47.8 | 38.2 | 10.8 | 27.1 | 57.6 | 52.6 | 24.4 | 14.2 | 40.1 | 24.1 | 33.8 |
| | Ours-AIC | 71.8 | 93.2 | 68.1 | 41.4 | 79.0 | 42.0 | 14.7 | 35.0 | 59.8 | 56.4 | 28.3 | 11.8 | 46.2 | 30.4 | 40.5 |

**Table 2:** *Comparison experiments with AIC [LHW* 20] and DDR [LLG* 19] at three different resolutions on NYU dataset.*

| | | SC | | | SSC | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | method | prec | recall | IoU | ceil. | floor | wall | win. | chair | bed | sofa | table | tvs | furn. | objs. | avg. |
| R1 | DDR | 68.1 | 95.8 | 66.1 | 21.0 | 61.7 | 33.1 | 11.8 | 19.1 | 51.1 | 40.5 | 11.8 | 9.5 | 34.5 | 16.8 | 28.3 |
| | Ours-DDR | 70.2 | 94.5 | 67.5 | 26.7 | 65.3 | 33.4 | 19.4 | 19.4 | 53.8 | 40.6 | 16.8 | 10.3 | 32.7 | 16.4 | 30.4 |
| | AIC | 69.4 | 92.5 | 65.6 | 28.8 | 68.5 | 34.8 | 5.8 | 22.9 | 52.3 | 43.5 | 17.5 | 7.6 | 31.5 | 12.7 | 29.6 |
| | Ours-AIC | 68.6 | 95.4 | 66.4 | 28.6 | 61.2 | 36.3 | 13.9 | 24.5 | 54.2 | 44.7 | 18.7 | 10.5 | 36.2 | 16.5 | 31.4 |
| R2 | DDR | 55.6 | 97.4 | 54.9 | 18.1 | 88.8 | 32.0 | 9.9 | 16.4 | 51.2 | 44.1 | 11.8 | 8.6 | 34.3 | 15.7 | 30.1 |
| | Ours-DDR | 64.1 | 87.6 | 58.6 | 23.2 | 81.3 | 30.3 | 18.9 | 17.5 | 54.7 | 45.4 | 14.4 | 12.1 | 34.2 | 16.2 | 31.6 |
| | AIC | 56.9 | 95.2 | 55.3 | 24.0 | 92.0 | 30.7 | 5.1 | 20.1 | 53.7 | 48.4 | 14.5 | 10.6 | 32.5 | 12.5 | 31.3 |
| | Ours-AIC | 62.5 | 90.1 | 58.4 | 25.3 | 89.7 | 33.6 | 11.7 | 21.5 | 54.7 | 48.2 | 15.7 | 10.5 | 36.0 | 15.5 | 32.9 |
| R3 | DDR | 43.8 | 97.5 | 43.3 | 14.4 | 44.0 | 21.7 | 15.3 | 13.9 | 52.7 | 43.1 | 11.6 | 11.6 | 32.9 | 14.5 | 25.1 |
| | Ours-DDR | 53.2 | 89.5 | 49.8 | 16.1 | 76.8 | 24.1 | 16.7 | 14.1 | 52.7 | 43.2 | 12.1 | 10.1 | 33.4 | 14.0 | 28.5 |
| | AIC | 44.6 | 95.1 | 43.6 | 15.0 | 44.8 | 21.8 | 4.6 | 15.7 | 52.2 | 45.9 | 10.9 | 9.3 | 31.2 | 11.2 | 23.9 |
| | Ours-AIC | 51.7 | 91.3 | 49.1 | 17.5 | 79.1 | 24.8 | 10.8 | 17.1 | 53.1 | 45.6 | 12.0 | 10.3 | 34.7 | 14.1 | 29.0 |

to verify the adaptability of our method to the resulting resolution. Finally, we validate each key design of our method through ablation experiments to show the effectiveness of decoupling.

### 4.1. Implementation Details

For the encoder, since SOTA methods [TCWZ21, ZLL*19] do not release their codes, we choose two open-source methods DDR [LLG*19] and AIC [LHW*20]. The resolution of the feature volume encoded by both of their encoders is $60 \times 36 \times 60$. For the results, we validate our method on 3 different resolution: $R1(30 \times 18 \times 30), R2(60 \times 36 \times 60), R3(120 \times 72 \times 120)$. For the sampling method and the number of points, we use the center point sampling strategy for $R2, R3$ and the uniform point sampling strategy for $R1$, with 16 points sampled in each voxel grid. For generating the training data for our point decoder, We use the voxel-wise label with the highest resolution ($240 \times 144 \times 240$) provided by the dataset.

Overall, we implement our model using PyTorch. We first pretrain the encoder according to [LLG*19, LHW*20]. Then, we fix the encoder parameters and train our point decoder with a learning rate of $10^{-3}$ using the SGD optimizer with a momentum of 0.9 and weight decay of $10^{-4}$. Our batch size is set to 8 and training stops when the loss no longer decreases within 5 epochs.

**Datasets.** We evaluate our method on two SSC datasets. One is the NYU-Depth-V2 [SHKF12] dataset. This dataset contains 1449 depth scenes. The second dataset is the NYUCAD [FMAJB16] dataset. This dataset uses synthetic depth maps, which provide more accurate depth values compared to the NYU dataset. Thus, it avoids the misalignment problem caused by sensors. Both datasets provide voxel-wise semantic labels.

**Evaluation metrics.** For semantic scene completion, we measure the intersection over union (IoU) between predicted voxel labels and ground-truth labels for all object categories. The overall performance is also obtained by computing the average IoU for all categories. For scene completion, all voxels will be categorized as empty or occupied. If a voxel belongs to any semantic category, it is counted as an occupied voxel. In addition to IoU, precision and recall are also reported. Note that IoU for semantic scene completion is often considered a more important metric for SSC tasks.

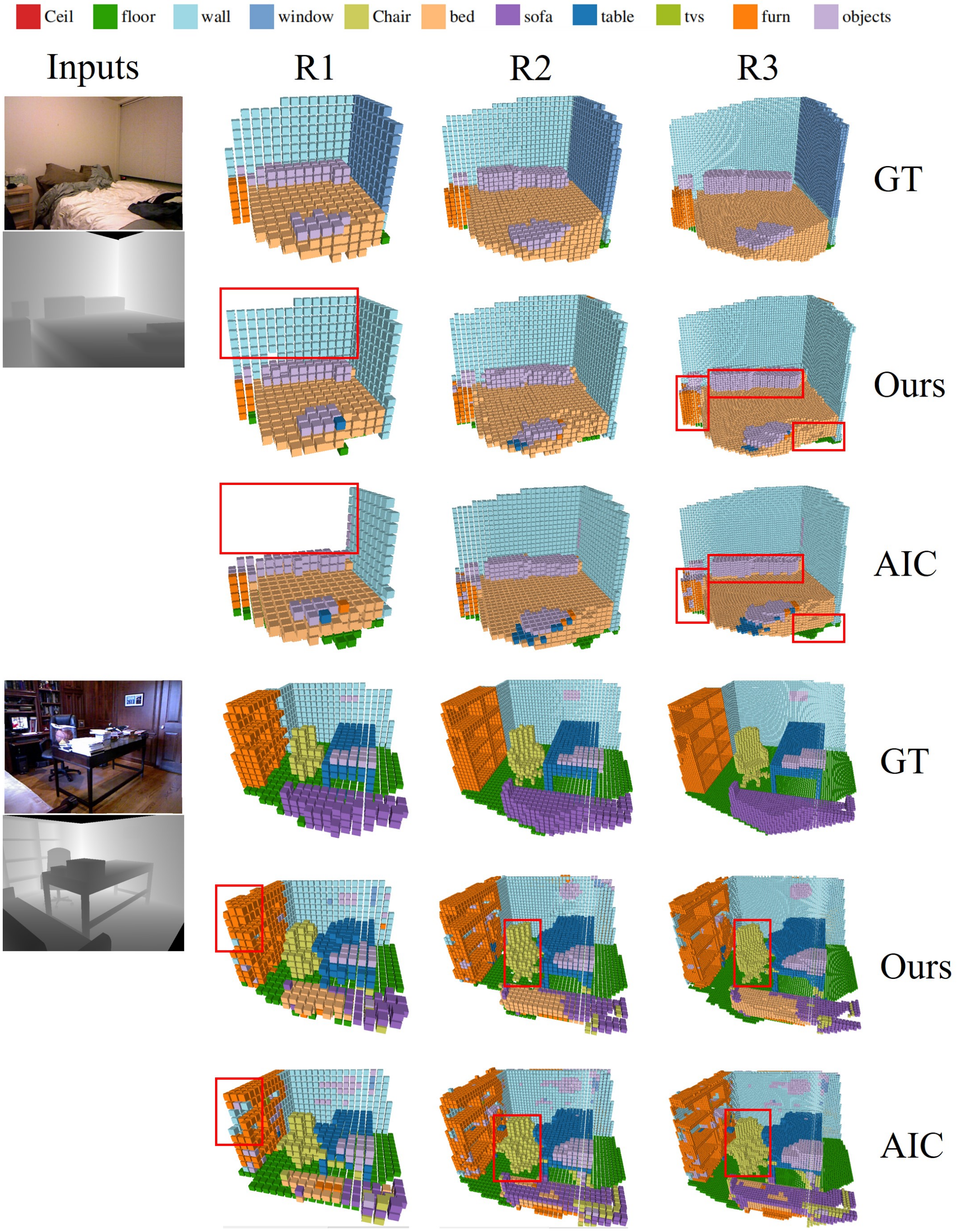


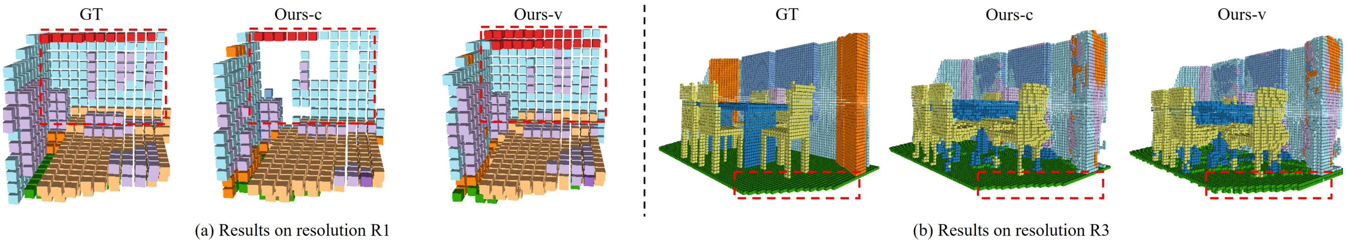
**Figure 6:** *Comparison results of our method with AIC [LHW*20] at three resolutions.*

### 4.2. Resolution adaptability

To verify the adaptability of our method to different resolution results, we perform comparison experiments with existing methods at three resolutions, R1, R2, and R3. Since DDR [LLG*19] and AIC [LHW*20] can only output the results for R2 resolution, for this reason, we directly sampled their results by trilinear sampling [TCA*17] to obtain results at R1 and R3 resolution.

Table 1 shows the comparison results on NYUCAD dataset. Our method outperforms both DDR [LLG*19] and AIC [LHW*20] by 6% in terms of $R1$ and $R3$ resolution. Meanwhile, even though their method is designed for $R2$ resolution, our method still achieves nearly 1% over theirs.

Figure 6 shows the visualization results. We give the comparison results with AIC [LHW*20]. As seen in Figure 6, at R1 resolu-



(a) Results on resolution R1

(b) Results on resolution R3

**Figure 7:** *The necessity of dynamically adjusting the sampling strategy for different resolution results during the test.*

tion, direct trilinear downsampling of the results may cause loss of structure, while our method maintains the structure well. Also, at R3 resolution, our method shows much better details.

Comparison results on NYU dataset are reported in Table 2. Note the results of DDR and AIC differ from the original results in their paper on the NYU and NYUCAD datasets because they only open-sourced the code but not the model. Since our results need to be calculated based on their encoders, so we retrain their networks and retest the metrics using their code.

The above results show that we can maintain the accuracy of DDR and AIC at the original resolution while achieving better results at other resolutions, which validates the adaptability of our method to different resolutions.

### 4.3. Effectiveness of decoupling

In this subsection, we will verify the effectiveness of decoupling. Specifically, there are three key designs in our method to ensure adequate decoupling of resolution and network: i) Resolution adaptive feature encoding. ii) resolution-adapted point sampling strategy and iii) category-based label generation strategy. In the following, we conduct ablation experiments for these three strategies to verify the effectiveness of decoupling.

For the setup, we use AIC as our encoder and leave the other settings unchanged unless otherwise mentioned. Since the NYU and NYUCAD datasets are relatively similar, we only report ablation results on the NYUCAD dataset.

**Effectiveness of RAF.** Instead of using RAF as $f$, we directly use discrete feature volume as $f$ and retrain our point decoder. The altered $f(x)$ is calculated as follows: for a point $x$ in 3D space, we calculate the index value of the voxel to which x belongs and use the feature of that voxel as f(x). We denote this model as 'Ours-d'. The comparison results are given in Table 3. It shows that using only original feature volume cannot achieve good results at all resolutions while RAF can.

We also report the performance of different interpolation functions. The results are reported in Table 4. The results in the table illustrate that our interpolation method will be slightly better than trilinear interpolation. At the same time, the performance decreases a little if the weights are normalized, which indicates that the magnitude of the features will result in a better performance with appropriate amplification.

**Effectiveness of resolution-adapted point sampling.** Instead of using a dynamic sampling strategy for different resolutions, we use center sampling (Ours-c) or uniform sampling (Ours-v) strategy only. The comparison results are given in Table 3. It shows that using center sampling only does not yield good results at lower resolutions. Using uniform sampling only will reduce the accuracy of higher resolution slightly. Also, at higher resolutions, the speed of inference is much slower than that of center sampling because of the large number of points used for uniform sampling. Figure 7 gives the visualization results to explain why we need to dynamically adjust the sampling strategy for different resolution results during the testing phase. The necessity of using uniform sampling at low resolution is given in Figure 7(a). It can be seen that if center sampling is used, the final result shows some structure loss, while using uniform sampling maintains the structure well. (b) gives the results of different sampling strategies on high resolution. Although the difference in accuracy between uniform sampling and center sampling is not significant on high resolution, the use of uniform sampling causes some unwanted structural noise in local areas such as the floor, which is the reason for the 1% decrease in the performance of using even sampling on high resolution. This is one of the reasons why, in addition to efficiency, we use center sampling on high resolution.

**Effectiveness of category-based label generation.** Instead of sampling category by category, we uniformly sample points in the whole 3D space just like [MON*19, PNM*20] did and determine the label of the points based on the label value of the voxel in which the point is located. We retrain our model and denote this model as "Ours-r" and the results are reported in Table 3. The results show that sampling only uniformly over the entire space does not train our network well, however, category-wise sampling does. The reason may be a balanced point number of each class can provide a better supervision signal, making the network focus on the semantic boundary of each category and thus learn better spatial semantic partitioning.

## 5. Conclusion and Limitation.

In this paper, we propose a resolution-switchable semantic scene completion method. By decoupling the direct connection between resolution and network structure, we can generate results at multiple resolutions without redesigning and retraining. To achieve this, we design three key mechanisms. First, by converting feature volume into a resolution adaptive feature we can achieve better results at multiple resolutions. Second, the resolution-adapted point sampling strategy ensures that the lower resolution results are not

**Table 3:** Ablation studies on our 3 key designs.

| | R1 | | | | R2 | | | | R3 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| method | prec | recall | IoU | mIoU | prec | recall | IoU | mIoU | prec | recall | IoU | mIoU |
| Ours-d | 79.7 | 96.8 | 77.6 | 39.3 | 76.0 | 92.3 | 71.4 | 38.5 | 62.4 | 93.4 | 59.5 | 32.0 |
| Ours-c | 90.9 | 77.3 | 71.6 | 37.9 | 82.7 | 90.4 | 75.9 | 44.7 | 71.8 | 93.2 | 68.1 | 40.5 |
| Ours-v | 85.2 | 93.1 | 80.1 | 43.0 | 80.4 | 90.6 | 74.1 | 43.5 | 69.9 | 93.5 | 66.5 | 39.4 |
| Ours-r | 90.0 | 82.3 | 76.0 | 37.4 | 92.3 | 54.6 | 52.2 | 26.7 | 87.1 | 67.5 | 61.3 | 32.2 |
| Ours | 85.2 | 93.1 | 80.1 | 43.0 | 82.7 | 90.4 | 75.9 | 44.7 | 71.8 | 93.2 | 68.1 | 40.5 |

**Table 4:** Effect of different interpolation functions on the final results. (WN for weight normalization)

| | R1 | | | | R2 | | | | R3 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| method | prec | recall | IoU | mIoU | prec | recall | IoU | mIoU | prec | recall | IoU | mIoU |
| Ours-dist(with WN) | 83.5 | 90.9 | 77.1 | 41.5 | 78.5 | 92.2 | 73.5 | 44.0 | 66.6 | 94.4 | 63.9 | 38.6 |
| Ours-trilinear | 87.3 | 81.8 | 73.1 | 42.2 | 80.0 | 91.3 | 74.2 | 44.5 | 64.2 | 92.4 | 60.7 | 37.5 |
| Ours-dist(without WN) | 85.2 | 93.1 | 80.1 | 43.0 | 82.7 | 90.4 | 75.9 | 44.7 | 71.8 | 93.2 | 68.1 | 40.5 |

degraded and that the high-resolution results will not show too much noise. Ultimately, category-based training point generation can better learn the division of spatial semantics, thus making our approach further adaptable at different resolutions. We conducted experiments on two SSC datasets to verify the adaptability of our method to the results and the effectiveness of decoupling.

Our method still has the following limitations: first, our method is currently not an end-to-end method. In the future, it is worth exploring how to train our method end-to-end and maintain the current results. We believe that the key to solving this problem may lie in how to maintain the quality of the feature volume so that it is as good as after pre-training. Second, our RAF is generated by a linear interpolation method. This causes the whole RAF to be too smooth and lacks sharper "feature boundaries", which affects the final results. changing it to a nonlinear generation method could potentially improve the results. We believe these two aspects can be further improved in the subsequent work.

**Acknowledgement.** The paper is supported by: The National Natural Science Foundation of China No. 42075139,42077232; The Science and technology program of Jiangsu Province No. BE2020082 and BE2022063; The Innovation Fund of State Key Laboratory for Novel Software Technology No. ZZKT2022A18.

## References

[CGG19] Chen Y.-T., Garbade M., Gall J.: 3d semantic scene completion from a single depth image using adversarial training. In *2019 IEEE International Conference on Image Processing (ICIP)* (2019), IEEE, pp. 1835–1839. 2

[Che19] Chen Z.: *IM-NET: Learning implicit fields for generative shape modeling*. PhD thesis, Applied Sciences: School of Computing Science, 2019. 3

[CHY19] Chen R., Huang Z., Yu Y.: Am 2 fnet: Attention-based multiscale & multi-modality fused network. In *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)* (2019), IEEE, pp. 1192–1197. 2

[CLI*20] Chabra R., Lenssen J. E., Ilg E., Schmidt T., Straub J., Lovegrove S., Newcombe R.: Deep local shapes: Learning local sdf priors for detailed 3d reconstruction. In *European Conference on Computer Vision* (2020), Springer, pp. 608–625. 3

[CLQ*20] Chen X., Lin K.-Y., Qian C., Zeng G., Li H.: 3d sketch-aware semantic scene completion via semi-supervised structure prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2020), pp. 4193–4202. 2

[DdCKH19] Dourado A., de Campos T. E., Kim H., Hilton A.: Edgenet: Semantic scene completion from rgb-d images. *arXiv preprint arXiv:1908.02893 1* (2019). 2

[FMAJB16] Firman M., Mac Aodha O., Julier S., Brostow G. J.: Structured prediction of unobserved voxels from a single depth image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 5431–5440. 6

[GCSG19] Garbade M., Chen Y.-T., Sawatzky J., Gall J.: Two stream 3d semantic scene completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops* (2019), pp. 0–0. 1, 2, 3

[GT18] Guo Y.-X., Tong X.: View-volume network for semantic scene completion from a single depth image. *arXiv preprint arXiv:1806.05361* (2018). 1, 2

[JSM*20] Jiang C., Sud A., Makadia A., Huang J., Niessner M., Funkhouser T., et al.: Local implicit grid representations for 3d scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2020), pp. 6001–6010. 3

[KBH06] Kazhdan M., Bolitho M., Hoppe H.: Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing* (2006), vol. 7. 2

[LHW*20] Li J., Han K., Wang P., Liu Y., Yuan X.: Anisotropic convolutional networks for 3d semantic scene completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2020), pp. 3351–3359. 2, 3, 6, 7

[LHZ*18] Liu S., Hu Y., Zeng Y., Tang Q., Jin B., Han Y., Li X.: See and think: Disentangling semantic scene completion. *Advances in Neural Information Processing Systems 31* (2018). 2

[LLG*19] Li J., Liu Y., Gong D., Shi Q., Yuan X., Zhao C., Reid I.: Rgbd based dimensional decomposition residual network for 3d semantic scene completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2019), pp. 7693–7702. 2, 3, 6, 7

[LLY*19] Li J., Liu Y., Yuan X., Zhao C., Siegwart R., Reid I., Cadena C.: Depth based semantic scene completion with position importance aware loss. *IEEE Robotics and Automation Letters 5*, 1 (2019), 219–226. 2, 3

[LLY*20] Liu Y., Li J., Yan Q., Yuan X., Zhao C., Reid I., Ca-

DENA C.: 3d gated recurrent fusion for semantic scene completion. *arXiv preprint arXiv:2002.07269* (2020). 2, 3

[MON*19] MESCHEDER L., OECHSLE M., NIEMEYER M., NOWOZIN S., GEIGER A.: Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2019), pp. 4460–4470. 2, 5, 8

[PFS*19] PARK J. J., FLORENCE P., STRAUB J., NEWCOMBE R., LOVEGROVE S.: Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2019), pp. 165–174. 3

[PNM*20] PENG S., NIEMEYER M., MESCHEDER L., POLLEFEYS M., GEIGER A.: Convolutional occupancy networks. In *European Conference on Computer Vision* (2020), Springer, pp. 523–540. 2, 3, 4, 5, 8

[QYSG17] QI C. R., YI L., SU H., GUIBAS L. J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems 30* (2017). 2

[SHKF12] SILBERMAN N., HOIEM D., KOHLI P., FERGUS R.: Indoor segmentation and support inference from rgbd images. In *European conference on computer vision* (2012), Springer, pp. 746–760. 6

[SYZ*17] SONG S., YU F., ZENG A., CHANG A. X., SAVVA M., FUNKHOUSER T.: Semantic scene completion from a single depth image. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2017), pp. 1746–1754. 1, 2, 3

[TCA*17] TCHAPMI L., CHOY C., ARMENI I., GWAK J., SAVARESE S.: Segcloud: Semantic segmentation of 3d point clouds. In *2017 international conference on 3D vision (3DV)* (2017), IEEE, pp. 537–547. 7

[TCWZ21] TANG J., CHEN X., WANG J., ZENG G.: Not all voxels are equal: Semantic scene completion from the point-voxel perspective. *arXiv preprint arXiv:2112.12925* (2021). 2, 6

[WTNT19] WANG Y., TAN D. J., NAVAB N., TOMBARI F.: Forknet: Multi-branch volumetric semantic completion from a single depth image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2019), pp. 8608–8617. 2, 5

[WY10] WANG X., YANG R.: Learning 3d shape from a single facial image via non-linear manifold embedding and alignment. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (2010), IEEE, pp. 414–421. 2

[ZLHQ21] ZHANG S., LI S., HAO A., QIN H.: Point cloud semantic scene completion from rgb-d images. In *Proceedings of the AAAI Conference on Artificial Intelligence* (2021), vol. 35, pp. 3385–3393. 2

[ZLL*19] ZHANG P., LIU W., LEI Y., LU H., YANG X.: Cascaded context pyramid for full-resolution 3d semantic scene completion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2019), pp. 7801–7810. 2, 6

[ZWZ*18] ZHANG L., WANG L., ZHANG X., SHEN P., BENNAMOUN M., ZHU G., SHAH S. A. A., SONG J.: Semantic scene completion with dense crf from a single depth image. *Neurocomputing 318* (2018), 182–195. 1, 2

[ZZY*18] ZHANG J., ZHAO H., YAO A., CHEN Y., ZHANG L., LIAO H.: Efficient semantic scene completion network with spatial group convolution. In *Proceedings of the European Conference on Computer Vision (ECCV)* (2018), pp. 733–749. 2