The Visual Computer
https://doi.org/10.1007/s00371-018-1586-7

ORIGINAL ARTICLE

Deep 3D semantic scene extrapolation

Ali Abbasi[1] · Sinan Kalkan[1] · Yusuf Sahillioğlu[1]

Abstract

Scene extrapolation is a challenging variant of the scene completion problem, which pertains to predicting the missing part(s) of a scene. While the 3D scene completion algorithms in the literature try to fill the occluded part of a scene such as a chair behind a table, we focus on extrapolating the available half-scene information to a full one, a problem that, to our knowledge, has not been studied yet. Our approaches are based on convolutional neural networks (CNN). As input, we take the half of 3D voxelized scenes, then our models complete the other half of scenes as output. Our baseline CNN model consisting of convolutional and ReLU layers with multiple residual connections and Softmax classifier with voxel-wise cross-entropy loss function at the end. We train and evaluate our models on the synthetic 3D SUNCG dataset. We show that our trained networks can predict the other half of the scenes and complete the objects correctly with suitable lengths. With a discussion on the challenges, we propose scene extrapolation as a challenging test bed for future research in deep learning. We made our models available on https://github.com/aliabbasi/d3dsse.

Keywords 3D scenes · Extrapolation · Convolutional neural networks

1 Introduction

Thanks to the availability of consumer-level 3D data acquisition sensors such as Microsoft Kinect or Asus Xtion Live, it is now quite easy to build digital copies of the environments or the objects. Regardless of its rapid development, some problems will always remain in the raw output of such sensing technologies. The notorious example is the missing data artifact which emerges due to self-occlusion, occlusion by other objects, or insufficient coverage of the viewing frustum of the device. While the occlusion-related problems have been addressed extensively in the literature under the *shape completion* or *scene completion* problems, the coverage-related version is, to our knowledge, not studied yet. In this paper, we address this new problem that we cast as *scene extrapolation*. Specifically, given the first half of the scene, we extrapolate it to recover the second one. This is a valid scenario as the 3D sensor may not be able to capture the other half due to, for instance, physical limitations. This is also a more challenging problem than the completion task which is able to use the available surroundings of the occluded holes to be filled, an information that is unavailable to our extrapolation framework. Our problem is also fundamentally different from the family of *shape or scene synthesis* problems which aims to reconstruct the target based on textual or geometric descriptions such as captions or 3D point clouds (see Fig. 1).

In this paper, we study the problem of 3D scene extrapolation with convolutional neural networks (CNNs). The dataset we use [38] is voxelized 3D indoor scenes with object labeling at voxel level. Our proposed models take the first half of each 3D scene as input and complete (estimate) the other side of it. We use voxel-wise Softmax cross-entropy loss function in the network to train our models.

The main contributions of this paper can be described as follows. First of all, we introduce a novel problem different than other completion tasks, namely the extrapolation of a half-scene into a full one. Secondly, we propose a deep CNN model using both 3D and 2D inputs to estimate the unseen half of the scene given the visible part as its input.

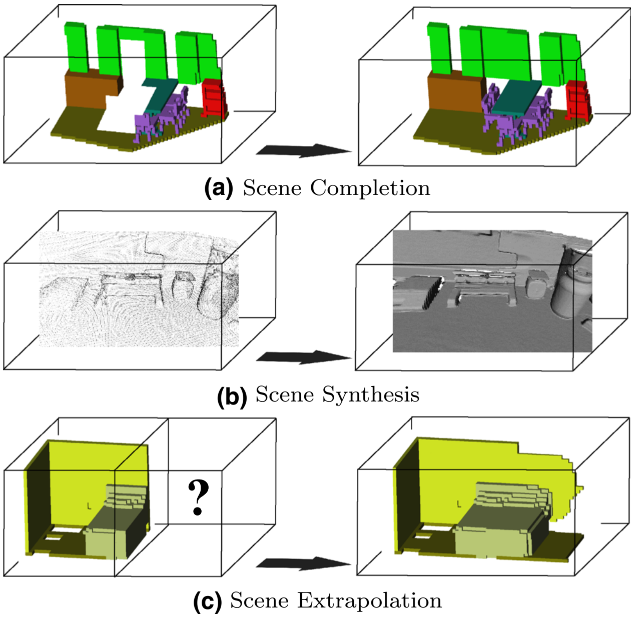
✉ Ali Abbasi
abbasi.ali.tab@gmail.com; ali.abbasi@ceng.metu.edu.tr

Sinan Kalkan
skalkan@ceng.metu.edu.tr

Yusuf Sahillioğlu
ys@ceng.metu.edu.tr

[1] Middle East Technical University, Ankara, Turkey

Published online: 17 August 2018



(a) Scene Completion

(b) Scene Synthesis

(c) Scene Extrapolation

Fig. 1 An illustration of the relevant problems. **a** Scene completion aims at estimating the missing hole in the middle of scene. **b** Scene synthesis constructs 3D scenes from point clouds. **c** Scene extrapolation estimates the unseen part of the environment from the seen scene (the problem addressed in this paper) [best viewed in color]

2 Related work

Missing data can be completed in various ways depending on the size of the misses. We broadly categorize these methods as rule-based and learning-based approaches.

A set of rule-based approaches utilize an ‘as smooth as possible’ filling strategy for the completion of sufficiently small missing parts, or holes [26,42,51]. When the holes enlarge to a certain extent, context-based copy-paste techniques yield more accurate results than smooth filling techniques do. In this line of research, convenient patches from other areas of the same object [11,29,37] or from a database of similar objects [9,21,24] are copied to the missing parts. Although these methods generally arise in the context of 3D shape completion, appropriate modifications enable image [2,13,22] and video [16,40] completion as well. A common limitation to this set of works is the assumption on the existence of the missing part in the context. Besides, these algorithms may generate patches inconsistent with the original geometry as they may not be able to take full advantage of the global information. Although semi-automatic methods that operate under the guidance of a user [12,46] may improve the completion results, a more principled solution is based on supervised machine learning techniques, as we discuss in the sequel.

A different rule-based pipeline is based on the family of iterative closest point algorithms [3,4,30,45]. These algorithms are applicable when there are multiple sets of partial data available at arbitrary orientations, a case that typically emerges as a result of 3D scanning from different views [28]. By alternating between closest point correspondence and rigid transformation optimization, these methods unify all partial scans into one full scan, hence completing the geometry. Requiring access to other complementary partial data is a fundamental drawback on these works.

Scene completion, posed either on 2D image domain or 3D Euclidean domain, is more challenging than the shape completion case discussed thus far. The difficulty mainly stems from the fact that it is an inherently ill-posed problem as there are infinitely many geometric pieces that look accurate, e.g., a chair as well as a garbage bin is equally likely to be occluded by a table. A rule-based approach such as [52] should, therefore, be replaced with a machine learning-based scheme as the latter imitates the thinking mechanism of a human for whom the completion task is trivial, e.g., we immediately decide the highly occluded component based on our years of natural training. To this end, [38] recently proposed a 3D convolutional network that takes a single depth image as input and outputs volumetric occupancy and semantic labels for all voxels in the camera view frustum. Although this is the closest work to our paper, we deal with the voxels that are not in the view frustum, hence the even more challenging task of scene extrapolation instead of completion, or interpolation. Similar to [38], unobserved geometry is predicted by [6] using structured random forest learning that is able to handle the large dimensionality of the output space.

Recent deep generative models have shown promising success on completion of images at the expense of significant training times, e.g., in the order of months [17,19,25,47]. There also are promising 3D shape completion results based on deep neural nets [5,44], which again suffer from the tedious training stage.

In the shape synthesis problem, given a description of a shape or a scene to be synthesized, it is aimed to reconstruct the query in the form of an image [49,50], video [39], or surface embedded in 3D [7,15,18,36]. There are many sources for the descriptive information ranging from a verbal description of the scene to 3D point cloud of a shape. Synthesis can be performed by either generating the objects [41] or retrieving them from a database [8,20].

In deep learning, there are many generative models that can scale well at large spaces such as natural images. The prominent models include generative adversarial networks (GANs) [10], variational autoencoders [32], and pixel-level convolutional or recurrent networks [34]. In addition to modeling the extrapolation problem as a voxel-wise classification problem, we chose GANs among the generative models since (1) they are easier to construct and train, (2) they are computationally cheaper during training and inference, and (3) they yield better and sharper results.

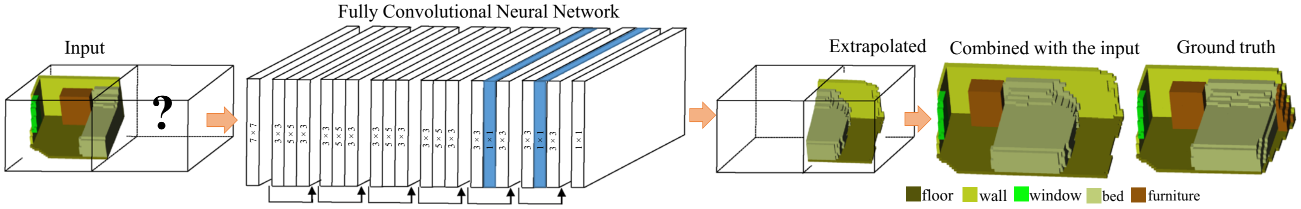


Fig. 2 The scene extrapolation problem addressed in the paper, and how we use deep learning to solve it. Our deep CNN model takes in a half of the 3D voxelized scene as input, and after applying some convolution operations, generates the full scene as output. The numbers inside the layers indicate the kernel sizes. Highlighted layers show dilation convolutions. The architecture consists of several residual connections, as shown by the arrows [best viewed in color]

As a summary, completion and synthesis of shape, scene, images or videos have been extensively studied in the literature. However, scene extrapolation is a new problem that we propose to solve in this paper using deep networks.

3 Scene extrapolation with deep models

We introduce two main models; the first is CNN-SE and the other one is Hybrid-SE, which use CNN-SE as a sub-network and extend it to use 2D top-view projection as another input. We also test the U-Net architecture [35] in scene extrapolation task.

3.1 A convolutional neural network for scene extrapolation (CNN-SE)

CNNs [23] are specialized neural networks with local connectivity and shared weights. With such constraints and inclusion of other operations such as pooling and non-linearity, CNNs essentially learn a hierarchical set of filters that extract the crucial information from the input for the problem at hand.

In our approach, we use deep fully CNN for the 3D semantic scene extrapolation task (Fig. 2). Fully CNN models consist of only convolution and do not use pooling or fully connected layers. Given the first half of 3D voxelized scene ($\mathbf{f}^{3D} \in \mathbb{V}^{w \times h \times d}$, where $\mathbb{V} = \{1, \ldots, 14\}$ is the set of object categories each voxel can take; w, h, d are, respectively, the width, the height and the depth of the scene), our task is to generate the other half of scene ($\mathbf{s}^{3D} \in \mathbb{V}^{w \times h \times d}$), which is semantically consistent with first half of scene ($\mathbf{f}^{3D}$) and conceptually realistic (e.g., objects have correct shapes, boundaries and sizes). Since our approaches work on voxelized data, we treat each 3D voxelized scene as '2D images' with multiple channels, where channels correspond to planes of the scene. We find that each voxel in a 3D scene is strongly dependent to adjacency voxels, or in the other words, each plane of voxels in a way similar to near planes. Therefore, we convert 3D voxelized scenes to 2D planes, like RGB images, instead we have 42 channels (42 layers of voxels—i.e., w is 42) for the input scene. Our models complete the missing channels from an input 3D scene.

We have experimented with many architectures and structures, including pooling layers, strided and deconvolution in bottleneck architectures, and dilated convolutions [48]. However, we found that a network with only convolution operations with stride 1 yielded better results. Therefore, we design our architecture as convolutional layers with stride 1 and same padding size, see Fig. 2.

Table 1 lists the details of the CNN-SE model. The fully CNN model takes input scene and first applies a 7 × 7 convolutions, then the architecture continues with 6 residual blocks [14]. Each residual block contains 3 layers of ReLU non-linearity (defined as $\text{relu}(x) = \max(0, x)$; see [33]) followed by convolution after each. It finishes with a 1×1 convolutional layers, and Softmax classifier with voxel-wise cross-entropy loss, L_{CE}, between the generated and the ground truth voxels as follows:

$$L_{\text{CE}} = \frac{1}{m} \sum_{i} H(\mathbf{t}_i^{3\text{D}}, \mathbf{s}_i^{3\text{D}}), \tag{1}$$

where m is the batch size; $\mathbf{s}_i^{3\text{D}}$ is the extrapolated scene for the ith input, and $t_i^{3\text{D}}$ is the corresponding ground truth. Cross-entropy (H) between two vectors (or serialized matrices) is defined as follows:

$$H(\mathbf{p}, \mathbf{q}) = -\sum_{i} p_i \log(q_i). \tag{2}$$

We also use a smoothness penalty term, L_S, to make the network produce smoother results by enforcing consistency between each generated plane $\mathbf{s}_{ij}^{p}$ (i.e., the generated jth voxel plane for the ith input instance) and its next plane in the ground truth, $\mathbf{t}_{i(j+1)}^{p}$:

$$L_S = \frac{1}{m} \sum_{i,j} H\left(\mathbf{t}_{i(j+1)}^{p}, \mathbf{s}_{ij}^{p}\right), \tag{3}$$

where m is again the batch size.

Table 1 The architecture details of our CNN model

	Layer	Activation	Batch norm.	Kernel	Stride	Dilation	Outputs
(a) CNN model architecture							
	Conv.	–	×	7 × 7	1	1	64
	Res1. ×4	–	–	–	–	–	64
	Res2. ×2	–	–	–	–	–	64
	Conv.	–	×	1 × 1	1	1	14 × 42
(b) Residual blocks internal architecture. Res2. use dilation 2 and 4, respectively, in two blocks							
Res1.	–	R	×	–	–	–	–
	Conv.	R	×	3 × 3	1	1	64
	Conv.	R	×	5 × 5	1	1	64
	Conv.	–	–	3 × 3	1	1	64
Res2.	–	R	×	–	–	–	–
	Conv.	R	×	3 × 3	1	1	64
	D.Conv.	R	×	1 × 1	1	2, 4	64
	Conv.	–	–	3 × 3	1	1	64

The layer type is shown in the 'Layer' column; 'Conv.' stands for convolution, and 'Res' for residual block, which is detailed in (b). 'R' shows the ReLU activation function in the second column. The last layer is a Softmax classifier

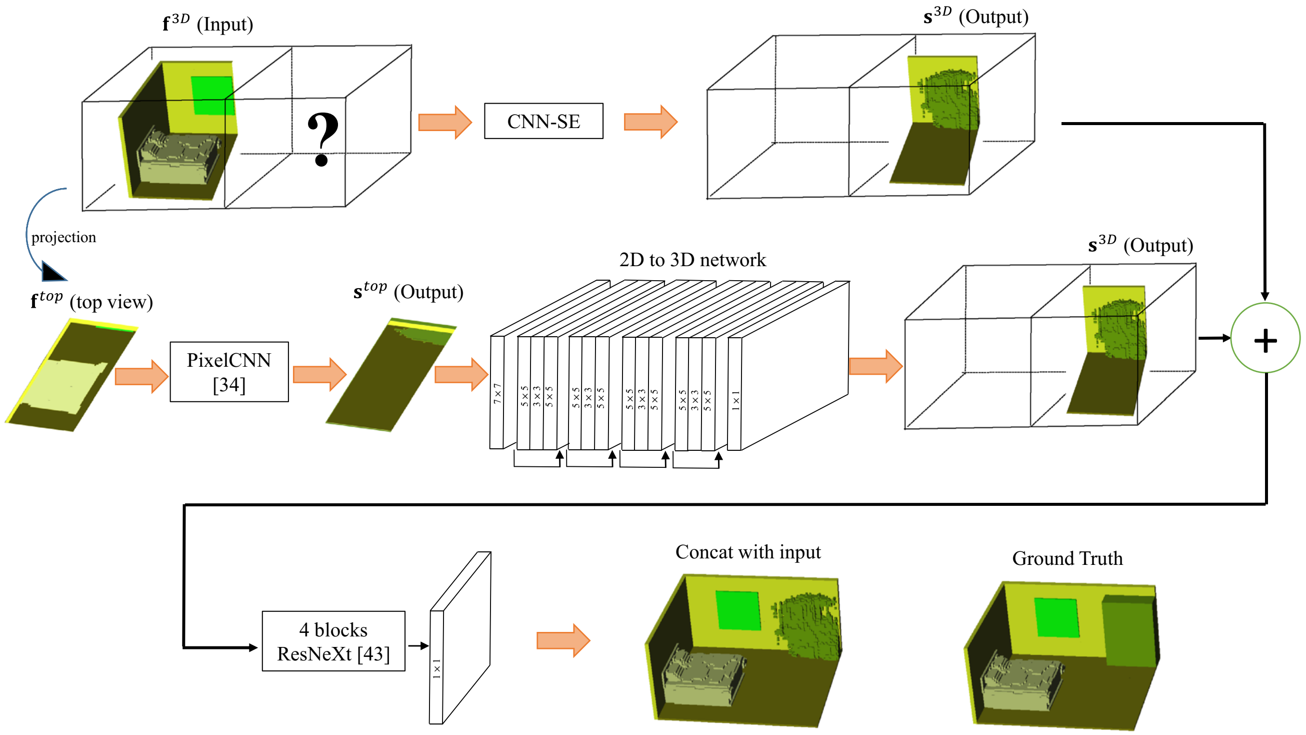


Fig. 3 The hybrid model used in our paper. We feed the half of 3D voxelized scene as input to the CNN-SE network—see Sect. 3.1. In parallel, the top-view projection of the first half (i.e., $\mathbf{f}^{\text{top}}$) is fed into PixelCNN [34] to generate the other side top view ($\mathbf{s}^{\text{top}}$). The 2D to 3D network, which consists of multiple residual blocks, maps its 2D input to 3D. Then this output is aggregated with CNN-SE network output and goes into a smoother network consists of 4 ResNeXt [43] blocks [best viewed in color]

Moreover, against overfitting, we add L_2 regularization on the weights:

$$L_2 = \frac{\lambda}{2} \sum_i w_i^2, \tag{4}$$

where λ is a constant, controlling the strength of regularization (λ is tuned to 0.001), and w_i is a weight.

In addition the focal loss L_f [27] is added to have faster convergence. Therefore, the total loss that we try to minimize with our CNN model is as follows:

$$L_{\text{total}} = L_{\text{CE}} + L_S + L_2 + L_f. \tag{5}$$

3.2 Scene extrapolation with hybrid model (hybrid-SE)

Figure 3 shows the overall view of our hybrid model. This model, in addition to taking the first half in 3D ($\mathbf{f}^{3\text{D}}$) as input, processes the 2D top-view projection in parallel. $\mathbf{f}^{3\text{D}}$ is fed to the CNN-SE model as explained in Sect. 3.1, in parallel, $\mathbf{f}^{\text{top}}$, the 2D top-view projection of $\mathbf{f}$ goes into an autoregressive generative model, namely PixelCNN [34], to generate the top view of the second part ($\mathbf{s}^{\text{top}}$). This top view should provide whereabouts and identities of the objects in the space to be extrapolated. Then a *2D to 3D* network takes this generated 2D top view as input and predicts the third dimension to make it 3D. The output from CNN-SE ($\mathbf{s}^{3\text{D}}$) and *2D to 3D* network ($\mathbf{s}^{\text{top}}$) are aggregated together and fed into a smoothing network consisting of 4 ResNeXt blocks [43]. In this model, we use the loss in Eq. 5 for CNN-SE network; for PixelCNN network, Eqs. 1, 4; for the focal loss for *2D to 3D* network, Eqs. 1, 2 and 4; and for the smoother network, Eqs. 1 and 2.

Our hybrid model predicts the top view of the unseen part. However, this 2D extrapolation is itself as challenging as 3D extrapolation. In order to demonstrate the full potential of having a good top-view estimation of the unseen part, we also solved the 3D extrapolation by feeding the known 2D top view of the unseen part.

4 Experiments and results

In this section, we train and evaluate our models on SUNCG [38] dataset. Our codes and models are available on https://github.com/aliabbasi/d3dsse.

4.1 Datasets

We used SUNCG [38] synthetic 3D scene dataset, for training and inference. This dataset contains 45,622 different scenes, about 400k rooms and 2644 unique objects. We constructed scenes by their available information with provided as JSON file for each scene. We parsed each scene JSON file and build a separate scene for each room. In the room extraction process we ignored scene types such as outdoor scenes, swimming pool. We used the binvox [31] voxelizer, which use space carving approach for both surface and interior voxels. To make the voxelizing process faster, we first voxelized each object individually then put them in their place in each room. During the object voxelization, we considered each 6cm as each voxel size, then voxelized objects with respect to their dimensions. The longest dimension in the object was divided by 6 to give the resolution of voxelizing for that object.

In our experiments, we removed categories that did not have sufficient number of instances, (or smaller than most occur objects in scene like beds, sofas, cabinets, windows), which reduced the number of object categories from 84 to 13 (plus one category, for emptiness). The removed objects include TVs, plants, toys, kitchen appliance, floor lamps, people, cats. We set the fix resolution i our models to $84 \times 42 \times 84$. We observed that this resolution was sufficient to represent the objects and the content of the scenes. We also removed scenes which has less than 12000 voxels in size, and leave about 211k scene, we used 200K as train the

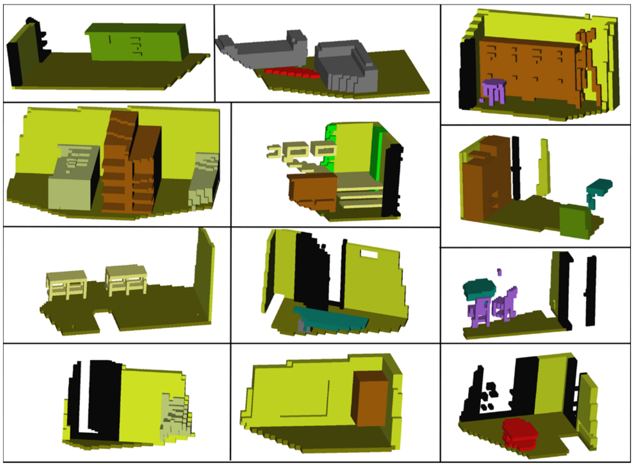


Fig. 4 Sample scenes from the synthetic SUNCG dataset [38], a challenging dataset containing diverse kinds of chairs, beds, sofas, shelves, tables, cabinets and coffee tables in different shapes and sizes [best viewed in color]

Table 2 The training details for our models

model	lr	bs	iter	optimizer
CNN-SE	5×10^{-5}	32	500,000	Adam
U-Net	5×10^{-5}	32	500,000	Adam
H-SE	1×10^{-5}	16	500,000	Adam
H-SE w gt	1×10^{-5}	16	500,000	Adam

The weights are initialize randomly with mean = 0 and std = 0.01. The training process for each model took about 8 days. 'w gt' indicates known 2D top view

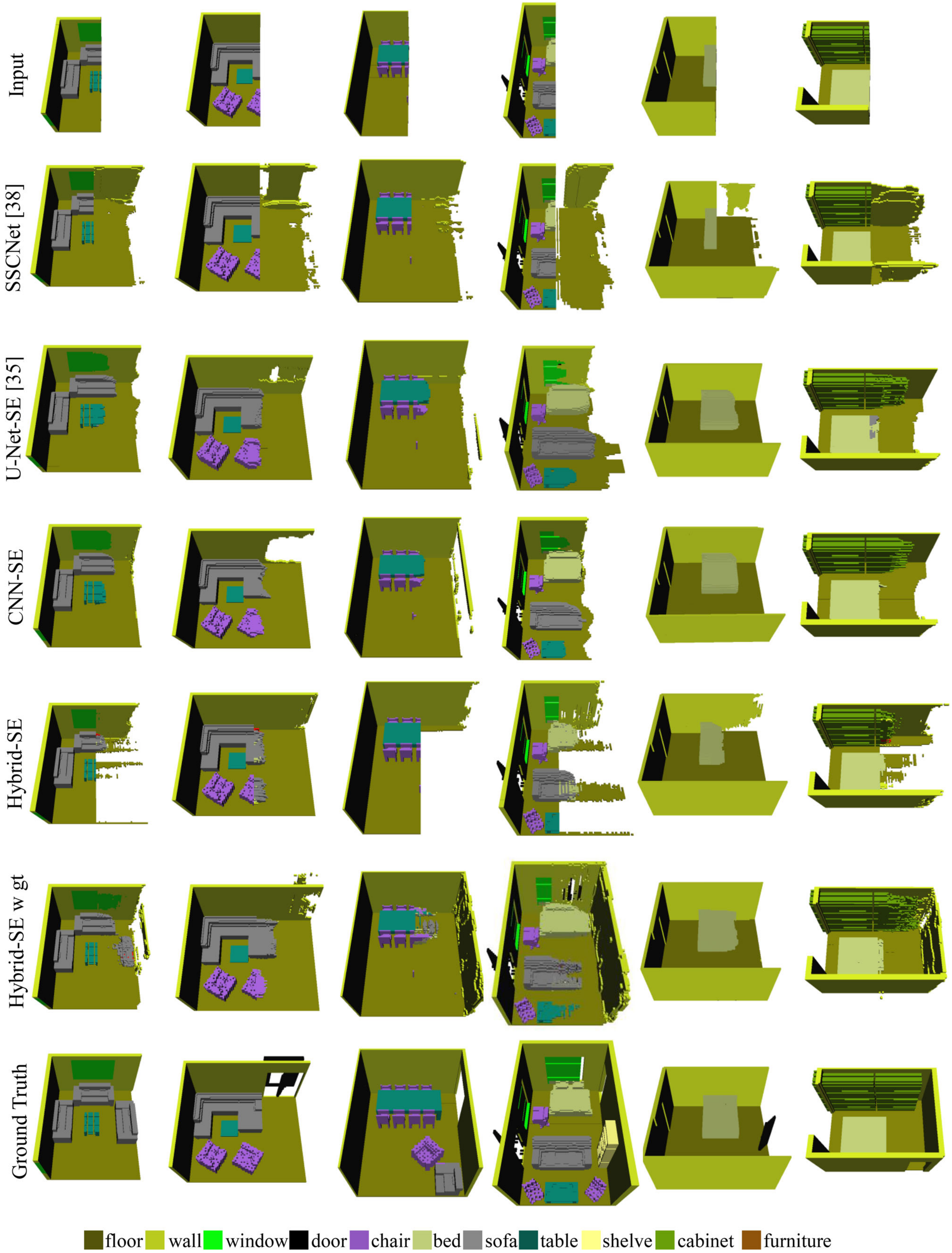


Fig. 5 Results on the SUNCG dataset [38] from our models. ‘w gt’ indicates known 2D top view [best viewed in color]

rest as test data. The data processing and building roughly took a week. Figure 4 shows some sample scenes from this challenging dataset.

4.2 Training details

We used Tensorflow [1] to implement our models. Table 2 summarizes the training details for each model individually. For all training processes, we used a single NVIDIA GeForce Titan X GPU.

4.3 Results

Figure 5 shows the results of our experiments. We use two metrics in order to evaluate the results. First, the accuracy metric, defined as the number of correctly generated voxels in the output divided by the total number of estimated voxels. In the first metric, we also consider empty voxels if they are estimated at the right place. The second metric is completeness, which is the number of correctly generated non-empty voxels in the output divided by the total number of non-empty voxels. Table 3 shows the quantitative results of our models, as well as per-category F1 scores.

4.4 Discussion

The architecture We have experimented with many architectures and structures for the generator, include pooling layers, bottleneck architectures with strided and deconvolution layers, yet, at the end, we concluded that a simple architecture of stride-1 convolution layers can generate better results. This suggests that it is better to keep the widths of the layers unreduced and allow the network to process information along the width of the scene at each convolution. This also allows the network to get a better estimation about the sizes of the objects since each convolution has access to the width of the scene. Moreover, worse performance on increased filter sizes suggests that highly local processing is better and bigger filter sizes lead to averaging of information across voxels.

Future work One can formulate the scene extrapolation task as a sequence modeling problem, such that the input sequence (the visible scene) is encoded by a sequence modeling framework, and the rest of the scene is decoded by another. Moreover, a context network can be trained in parallel together with the generation network such that the context network captures a high-level scene information and layout of the scene and modulates the generation process.

5 Conclusion

In this paper, we have proposed using convolutional networks for the scene extrapolation problem, which has not

Table 3 The accuracy, completeness, precision and recall measures of our models

Model	Accuracy	Completeness	Precision	Recall	Empty voxels	Floor	Wall	Window	Door	Chair	Bed	Sofa	Table	Cabinet	Shelving	Wardrobe	House stuff	Average
SSCNet [38]	89.1	27.3	16.4	22.9	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Our CNN-SE	93.7	40.8	45.7	**77.0**	**98.3**	**98.1**	82.5	57.8	26.8	21.6	80.6	69.7	12.5	61.6	38.0	64.9	5.0	55.2
Our U-Net	94.2	46.7	32.2	75.6	97.3	94.5	88.6	77.3	28.2	44.4	**82.2**	74.5	47.5	68.5	**49.3**	**85.9**	1.0	65.8
Our H-SE	94.4	49.2	58.2	46.2	97.2	90.4	83.3	44.0	13.0	18.0	61.5	51.3	8.0	45.7	21.0	58.3	8.0	46.1
Our H-SE w gt	**95.1**	**66.2**	**62.2**	62.6	97.4	95.5	**87.4**	**82.5**	**44.3**	**49.4**	81.1	**78.1**	**58.1**	**84.6**	43.7	71.0	**21.0**	**68.6**

The score under each object category shows F1 score. H-SE stands for Hybrid-SE model and 'w gt' means with ground truth of 2D projection top views
Bold text signifies the best scores for each column

been addressed in the literature before. We showed that the proposed models are able to extrapolate the scene toward the unseen part and extend successfully the ground, the walls and the partially visible objects.

However, we realized that the networks were unable to hallucinate objects in the extrapolated part of the scene if they did not have any part visible in the input scene. This is likely to be due to the highly challenging nature of the extrapolation problem, due to the huge inter-variability and intra-variability between the objects. To able to extrapolate at a level where new objects can be generated would likely require (1) a much larger dataset than we used and available in the literature, and (2) extraction and usage of higher-level semantic information from the input scene, as a modulator for the deep extrapolating networks.

Scene extrapolation is a challenging and highly-relevant problem for the computer graphics and deep learning societies, and our paper offers a new research problem and direction with which new and better learning and estimation methods can be developed.

Acknowledgements This work has been supported by the Scientific and Technological Research Council of Turkey (TÜBİTAK) under the Project EEEAG-215E255. We also thank NVIDIA for their donation of a Titan X graphics card.

References

1. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X.: TensorFlow: large-scale machine learning on heterogeneous systems. Software available from tensorflow.org (2015)
2. Averbuch-Elor, H., Kopf, J., Hazan, T., Cohen-Or, D.: Co-segmentation for space-time co-located collections. Vis. Comput. **52**, 1–12 (2017)
3. Besl, P.J., McKay, N.D.: A method for registration of 3-D shapes. IEEE Trans. Pattern Anal, Mach. Intell. **14**(2), 239–256 (1992)
4. Bouaziz, S., Tagliasacchi, A., Pauly, M.: Sparse iterative closest point. In: Rushmeier, H., Deussen, O. (eds.) Computer Graphics Forum, vol. 32, pp. 113–123. Wiley Online Library, New York (2013)
5. Dai, A., Qi, C.R., Nießner, M.: Shape completion using 3D-encoder-predictor CNNS and shape synthesis. arXiv preprint arXiv:1612.00101 (2016)
6. 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, pp. 5431–5440 (2016)
7. Fisher, M., Ritchie, D., Savva, M., Funkhouser, T., Hanrahan, P.: Example-based synthesis of 3D object arrangements. ACM Trans. Graph. (TOG) **31**(6), 135 (2012)
8. Fisher, M., Savva, M., Li, Y., Hanrahan, P., Nießner, M.: Activity-centric scene synthesis for functional 3D scene modeling. ACM Trans. Graph. (TOG) **34**(6), 179 (2015)
9. Gal, R., Shamir, A., Hassner, T., Pauly, M., Cohen-Or, D.: Surface reconstruction using local shape priors. In: Symposium on Geometry Processing, EPFL-CONF-149318, pp. 253–262 (2007)
10. Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Proceedings of the 27th International Conference on Neural Information Processing Systems, Vol. 2, pp. 2672–2680. MIT Press (2014)
11. Harary, G., Tal, A., Grinspun, E.: Context-based coherent surface completion. ACM Trans. Graph. (TOG) **33**(1), 5 (2014)
12. Harary, G., Tal, A., Grinspun, E.: Feature-preserving surface completion using four points. In: Computer Graphics Forum, vol. 33, pp. 45–54. Wiley Online Library, New York (2014)
13. Hays, J., Efros, A.A.: Scene completion using millions of photographs. ACM Trans. Graph. (TOG) **26**, 4 (2007)
14. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
15. Hu, S.M., Chen, T., Xu, K., Cheng, M.M., Martin, R.R.: Internet visual media processing: a survey with graphics and vision applications. Vis. Comput. **29**(5), 393–405 (2013)
16. Huang, J.B., Kang, S.B., Ahuja, N., Kopf, J.: Temporally coherent completion of dynamic video. ACM Trans. Graph. (TOG) **35**(6), 196 (2016)
17. Iizuka, S., Simo-Serra, E., Ishikawa, H.: Globally and locally consistent image completion. ACM Trans. Graph. (TOG) **36**(4), 107 (2017)
18. Kazhdan, M., Hoppe, H.: Screened poisson surface reconstruction. ACM Trans. Graph. (TOG) **32**(3), 29 (2013)
19. Kelly, B., Matthews, T.P., Anastasio, M.A.: Deep learning-guided image reconstruction from incomplete data. arXiv preprint arXiv:1709.00584 (2017)
20. Kermani, Z.S., Liao, Z., Tan, P., Zhang, H.: Learning 3D scene synthesis from annotated RGB-D images. In: Computer Graphics Forum, vol. 35, pp. 197–206. Wiley Online Library, New York (2016)
21. Kraevoy, V., Sheffer, A.: Template-based mesh completion. In: Symposium on Geometry Processing, vol. 385, pp. 13–22 (2005)
22. Kwatra, V., Essa, I., Bobick, A., Kwatra, N.: Texture optimization for example-based synthesis. ACM Trans. Graph. (ToG) **24**(3), 795–802 (2005)
23. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
24. Li, D., Shao, T., Wu, H., Zhou, K.: Shape completion from a single RGBD image. IEEE Trans. Vis. Comput. Graph. **23**(7), 1809–1822 (2017)
25. Li, Y., Liu, S., Yang, J., Yang, M.H.: Generative face completion. arXiv preprint arXiv:1704.05838 (2017)
26. Liepa, P.: Filling holes in meshes. In: Eurographics/ACM SIGGRAPH Symposium on Geometry Processing, pp. 200–205. Eurographics Association (2003)
27. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. arXiv preprint arXiv:1708.02002 (2017)
28. Liu, H., Zhang, L., Huang, H.: Web-image driven best views of 3D shapes. Vis. Comput. **28**(3), 279–287 (2012)
29. Mavridis, P., Sipiran, I., Andreadis, A., Papaioannou, G.: Object completion using k-sparse optimization. In: Computer Graphics Forum, vol. 34, pp. 13–21. Wiley Online Library, New York (2015)
30. Mellado, N., Aiger, D., Mitra, N.J.: Super 4pcs fast global point-cloud registration via smart indexing. In: Computer Graphics Forum, vol. 33, pp. 205–215. Wiley Online Library, New York (2014)

31. Min, P.: Binvox. https://www.patrickmin.com/binvox/. Accessed 21 June 2018
32. Mnih, A., Gregor, K.: Neural variational inference and learning in belief networks. arXiv preprint arXiv:1402.0030 (2014)
33. Nair, V., Hinton, G.E.: Rectified linear units improve restricted Boltzmann machines. In: Proceedings of the 27th International Conference on Machine Learning (ICML-10), pp. 807–814 (2010)
34. Oord, A.V.D., Kalchbrenner, N., Kavukcuoglu, K.: Pixel recurrent neural networks. arXiv preprint arXiv:1601.06759 (2016)
35. Ronneberger, O., Fischer, P., Brox, T.: U-net: convolutional networks for biomedical image segmentation. In: International Conference on Medical Image Computing and Computer-Assisted Intervention, pp. 234–241. Springer, Berlin (2015)
36. Sahillioğlu, Y., Yemez, Y.: Coarse-to-fine surface reconstruction from silhouettes and range data using mesh deformation. Comput. Vis. Image Underst. **114**(3), 334–348 (2010)
37. Sharf, A., Alexa, M., Cohen-Or, D.: Context-based surface completion. ACM Trans. Graph. (TOG) **23**(3), 878–887 (2004)
38. Song, S., Yu, F., Zeng, A., Chang, A.X., Savva, M., Funkhouser, T.: Semantic scene completion from a single depth image. In: Conference on Computer Vision and Pattern Recognition (2017)
39. Suwajanakorn, S., Seitz, S.M., Kemelmacher-Shlizerman, I.: Synthesizing obama: learning lip sync from audio. ACM Trans. Graph. (TOG) **36**(4), 95 (2017)
40. Wexler, Y., Shechtman, E., Irani, M.: Space-time completion of video. IEEE Trans. Pattern Anal. Mach. Intell. **29**(3), 915 (2007)
41. Wu, J., Zhang, C., Xue, T., Freeman, B., Tenenbaum, J.: Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling. In: Advances in Neural Information Processing Systems, pp. 82–90 (2016)
42. Xia, C., Zhang, H.: A fast and automatic hole-filling method based on feature line recovery. Comput. Aided Des. Appl. **4**, 1–9 (2017)
43. Xie, S., Girshick, R., Dollár, P., Tu, Z., He, K.: Aggregated residual transformations for deep neural networks. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5987–5995. IEEE (2017)
44. Yang, B., Wen, H., Wang, S., Clark, R., Markham, A., Trigoni, N.: 3D object reconstruction from a single depth view with adversarial learning. arXiv preprint arXiv:1708.07969 (2017)
45. Yang, J., Li, H., Campbell, D., Jia, Y.: GO-ICP: a globally optimal solution to 3D ICP point-set registration. IEEE Trans. Pattern Anal. Mach. Intell. **38**(11), 2241–2254 (2016)
46. Yang, L., Yan, Q., Xiao, C.: Shape-controllable geometry completion for point cloud models. Vis. Comput. **33**(3), 385–398 (2017)
47. Yeh, R.A., Chen, C., Lim, T.Y., Schwing, A.G., Hasegawa-Johnson, M., Do, M.N.: Semantic image inpainting with deep generative models. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5485–5493 (2017)
48. Yu, F., Koltun, V.: Multi-scale context aggregation by dilated convolutions. Int. Confer. Learn. Represent. **896**, 2016 (2016)
49. Zhang, H., Xu, M., Zhuo, L., Havyarimana, V.: A novel optimization framework for salient object detection. Vis. Comput. **32**(1), 31–41 (2016)
50. Zhang, H., Xu, T., Li, H., Zhang, S., Huang, X., Wang, X., Metaxas, D.: Stackgan: text to photo-realistic image synthesis with stacked generative adversarial networks. arXiv preprint arXiv:1612.03242 (2016)
51. Zhao, W., Gao, S., Lin, H.: A robust hole-filling algorithm for triangular mesh. Vis. Comput. **23**(12), 987–997 (2007)
52. Zheng, B., Zhao, Y., Yu, J.C., Ikeuchi, K., Zhu, S.C.: Beyond point clouds: scene understanding by reasoning geometry and physics. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3127–3134 (2013)

Ali Abbasi Ali Abbasi is a M.S. student working with Assoc. Prof. Yusuf Sahillioğlu and Assoc. Prof. Sinan Kalkan in Computer Engineering at the Middle East Technical University. He received his B.S. in Software Engineering from Tabriz Azad University, Iran. His current research interests include 3D scene analysis, image and video processing.

Sinan Kalkan received his M.Sc. degree in Computer Engineering from Middle East Technical University, Turkey, in 2003, and his Ph.D. degree in Informatics from the University of Gottingen, Germany, in 2008. After working as a postdoctoral researcher at the University of Gottingen and at Middle East Technical University, he is an assistant professor at Middle East Technical University since 2010. Sinan Kalkan's research interests include biologically motivated computer vision and image processing and developmental robotics.

Yusuf Sahillioğlu received the B.S. degree from Bilkent University in 2004, the first M.S. degree from Koç University in 2006, and the second from University of Florida in 2008, all in computer science. He completed his Ph.D. at Koç University in 2012 and worked as a postdoctoral researcher at University of Pennsylvania until he started as an assistant professor at the computer engineering department of Middle East Technical University in 2014. He is currently an associate professor at the same university. His research interests include digital geometry processing, computer graphics and vision.