Supplementary material for paper "CoReNet: Coherent 3D scene reconstruction from a single RGB image"

Stefan Popov spopov@google.com
Pablo Bauszat pablo.bauszat@gmail.com
Vittorio Ferrari vittoferrari@google.com
Google Research

1 Contents

The supplementary material provides:

- Qualitative results on synthetic scenes, with one (fig. 1), two (fig. 2), and three (fig. 3) objects in them.
- Qualitative results from the high resolution reconstruction experiments in sec. 4.1 in the main paper (fig. 4)
- Additional experiments to evaluate the generalization abilities of our multi-object models (sec. 2).
- Additional experiment to evaluate how well our models adapt to variation in camera parameters (sec. 3).
- Additional experiment on multi-object scenes. We train MeshRCNN [2] on ShapeNet-triplets and compare its performance to our models.
- Statistics for the ShapeNet-pairs and ShapeNet-triplets datasets from sec. 4.3 in the main paper (tab. 2).

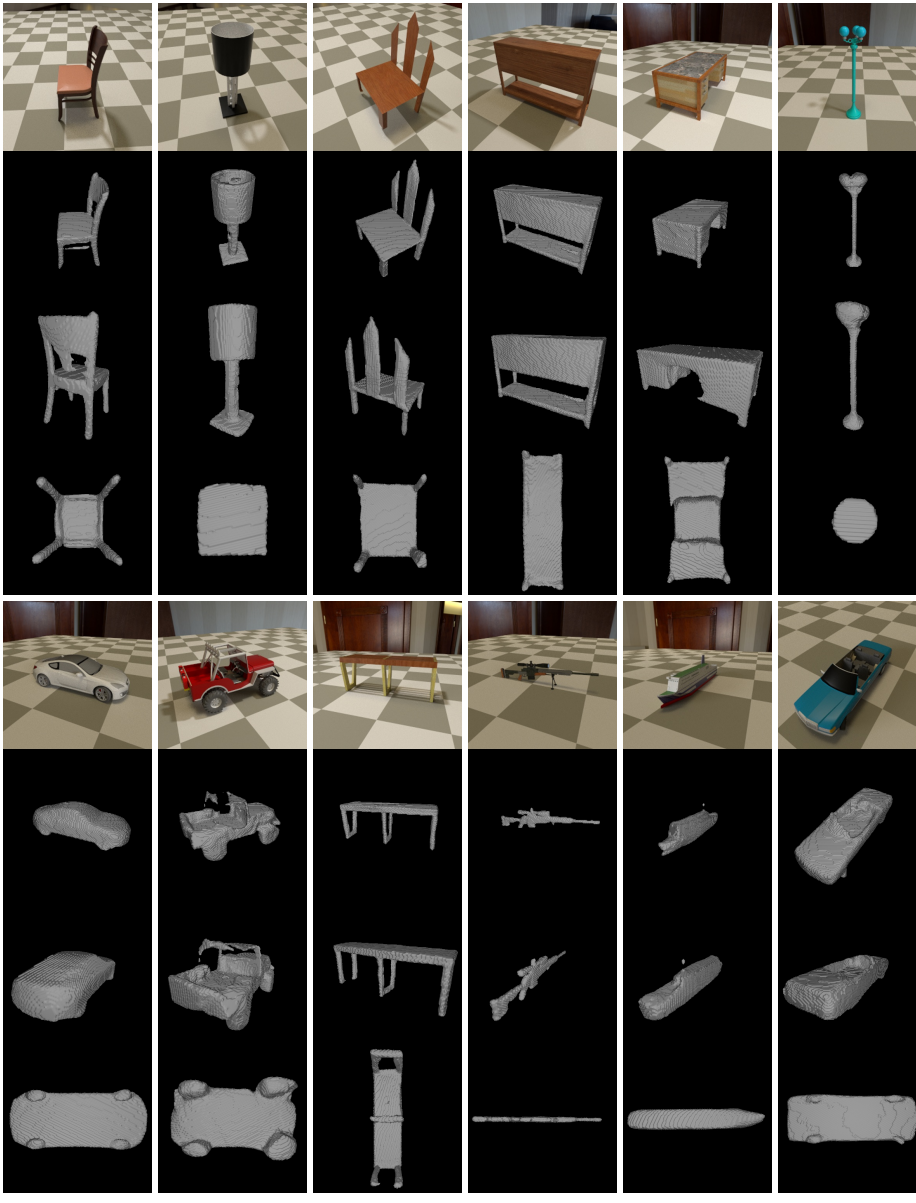

Fig. 1: Single object scenes, reconstructed with model h_7 from sec. 4.1 in the main paper. The first row for every scene is the input, followed by reconstruction from the same view point, a side view, and a view from below. The latter reveals that our model successfully reconstructs fine-grained details and thin structures of objects, even when they are only partially visible (e.g. chair legs, car tyres).

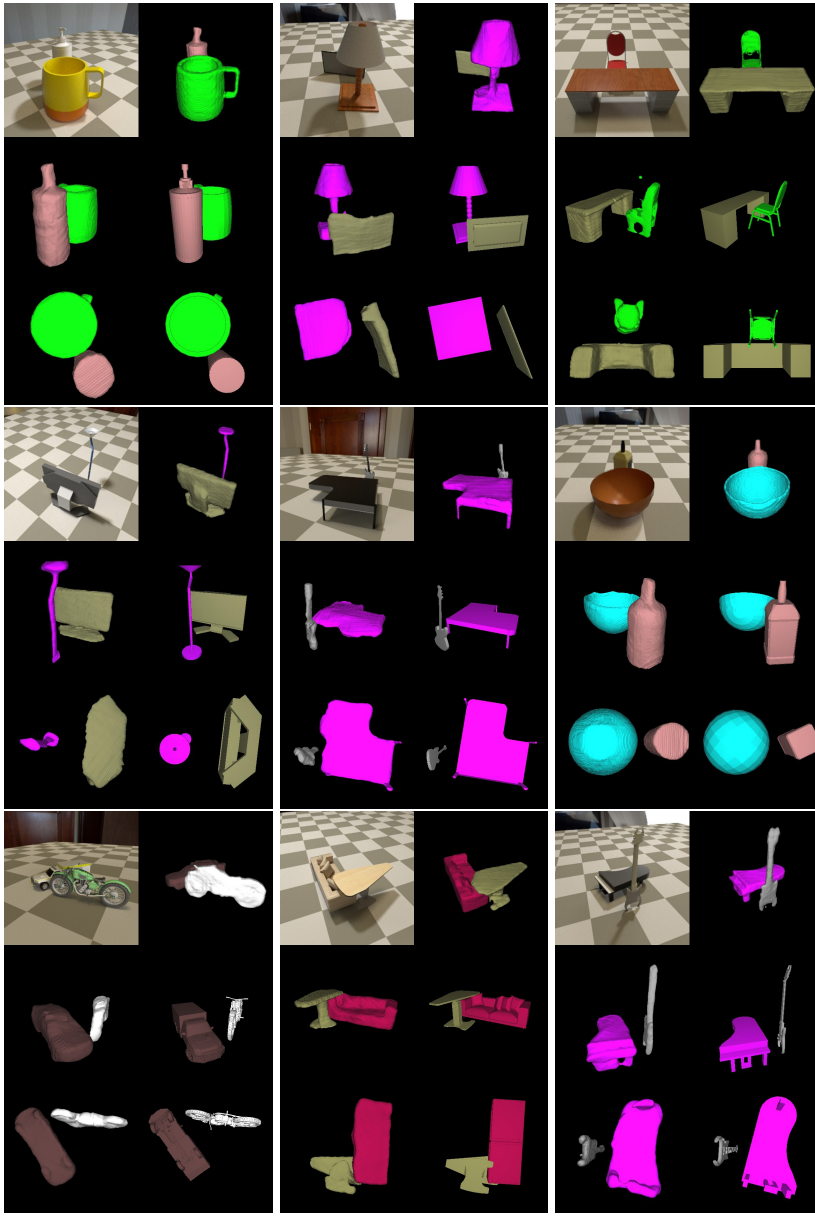

Fig. 2: Object pairs reconstructed by our model m_7 from sec. 4.3 in the main paper. The first row for every scene contains the input next to a reconstruction from the same viewpoint. The second shows reconstruction from another view next to ground-truth. The third – reconstruction and ground-truth from below. Our model hallucinates the occluded parts in these scenes and reconstructs all objects in their correct spatial arrangement, in a common coordinate frame.

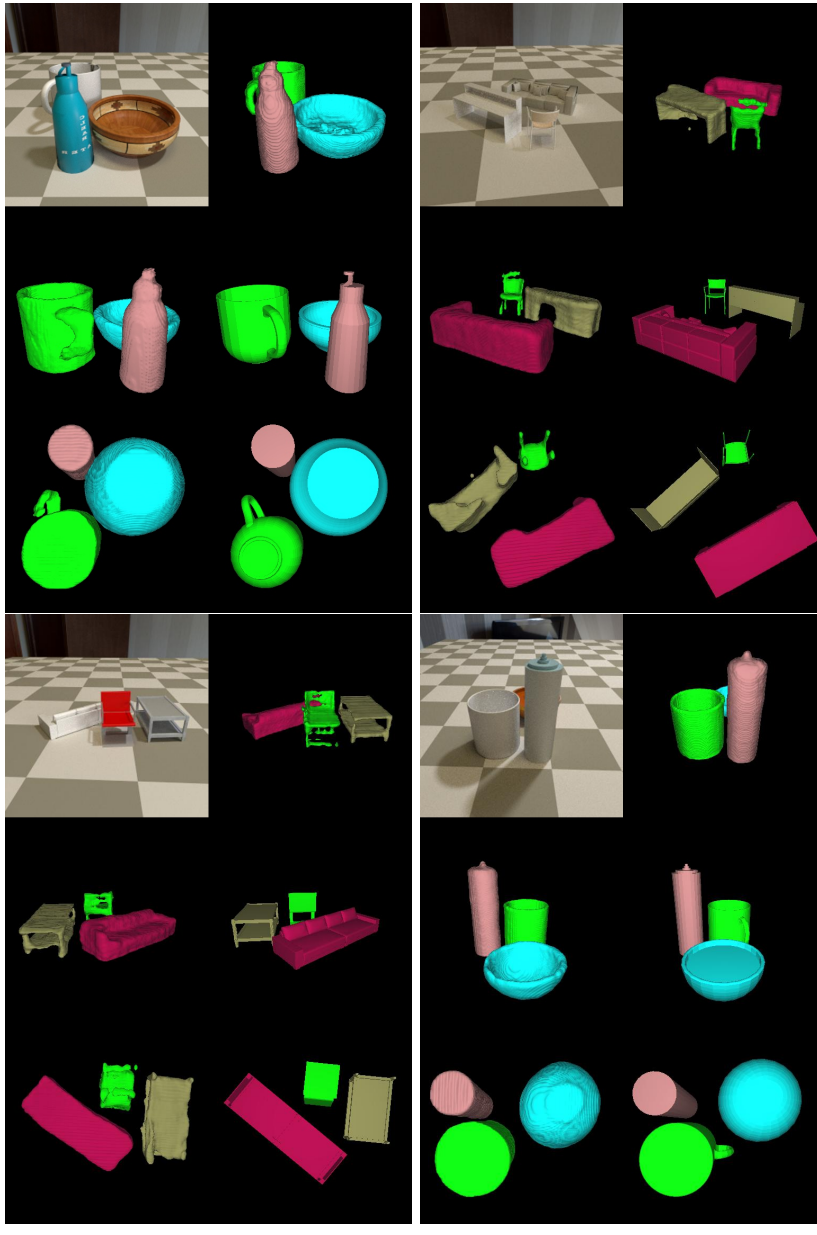

Fig. 3: Object triplets reconstructed by our model m_9 from sec. 4.3 in the main paper. The first row for every scene contains the input next to a reconstruction from the same view. The second shows reconstruction from another view next to ground-truth. The third – reconstruction and ground-truth from below. Our model hallucinates the occluded parts in these scenes and reconstructs all objects in their correct spatial arrangement, in a common coordinate frame.

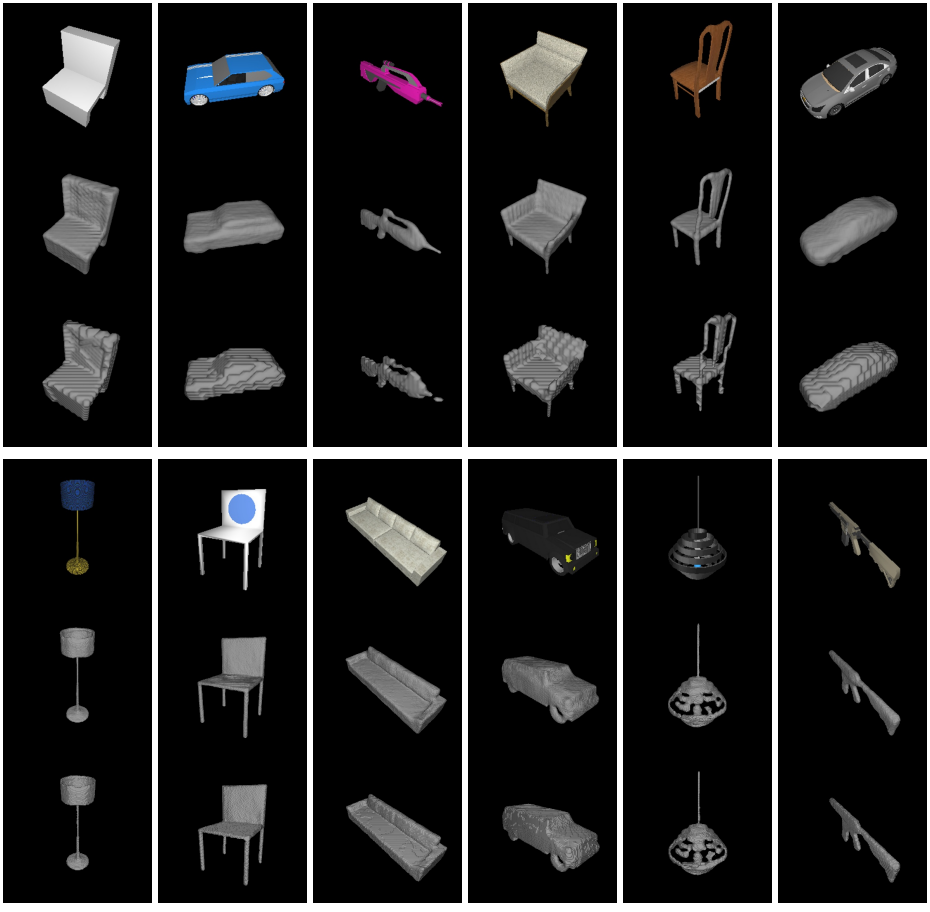

Fig. 4: Single object reconstructions at high resolutions from the models in sec. 4.1 in the main paper. In the top row, the model uses a 32^3 grid of points to represent the volume, while the reconstruction step computes a 128^3 discretization of the volume by calling the model 64 times (sec. 3.5 in paper). The bottom row shows images for a 128^3 grid with 256^3 reconstruction. For each scene, the top image shows the RGB input, followed by the high-resolution reconstruction, followed by a reconstruction at the native model resolution (*ie* 32^2 for the top row, 128^3 for the bottom).

2 Generalization of multi-object models

In this section, we evaluate how well our multi-class models m_7 and m_{11} from the main paper generalize in two scenarios: unseen class combinations and unseen number of objects.

For the first, we generate a new test set (ShapeNet-unseen-pairs) with the same procedure used in ShapeNet-pairs. We choose 7 class pairs that are not in ShapeNet-pairs and we generate 1000 scenes for each (bed-bottle, bowl-sofa, car-chair, display-table, guitar-mug, lamp-pillow, piano-motorcycle). mIoU of m_7 on ShapeNet-unseen-pairs is 24.6%, which is lower than m_7's mIoU on ShapeNet-pairs (43.1%). The global IoUs are however close – 47.7% on ShapeNet-unseen-pairs *vs* 52.7% on ShapeNet-pairs. This shows that m_7 can reconstruct the geometry of unseen class pairs rather well. It makes more mistakes in assigning the right classes however, likely learning to rely on the class pairs of ShapeNet-pairs for reconstruction.

For the second scenario, we evaluate m_7 and m_{11} on scenes with triplets of objects. None of the two models have seen scenes with three objects in them during training. Model m_7 was trained on of pairs, m_{11} – on single objects. We compare their performance to m_9, which has been trained on triplets.

Model m_7 achieves mIoU of 34.1% on the test set of ShapeNet-triplets, which is lower than the 43.9% of m_9 on the same test set. The global IoUs for the two models are on the other hand roughly equivalent – 45.2% for m_7 and 49.8% for m_9. This shows that m_7 generalizes well to the geometry of scenes with unseen number of objects. It makes however more mistakes in assigning the right class, likely relying on the learned class pairs from ShapeNet-pairs. This observation is also consistent with the unseen-classes scenario above.

Model m_{11} on the other hand underperforms on ShapeNet-triplets, both in reconstructing the overall geometry and assigning the right class. It achieves 5.5% mIoU and 24.1% global IoU. To understand why, we look at the train set of m_{11}. All object instances there are normalized and centered in the unit cube, which is in stark contrast to the pose distribution of object instances in ShapeNet-triplets. Model m_{11} has likely learned to rely on this fact during reconstruction. In contrast, model m_7 is trained on ShapeNet-pairs, which has a more varied distribution of object poses. This allows it to generalize to the unseen case of object triplets, as demonstrated above.

3 Handling camera parameters

model	grid resolution	skip conn.	mIoU
e_1	128×128×256	yes	58.2%
e_2	128×128×256	no	15.1%
e_3	128×128×384	yes	57.8%
e_4	128×128×384	no	8.6%
h_6	128×128×128	yes	58.1%

Table 1: Reconstructing with a variable camera and object position.

We evaluate how well our models adapt to variation in camera parameters, which we kept fixed in sec. 4.1 in the main paper. We move the camera randomly along the depth axis in a $1{\times}1{\times}l$ volume and place an object in front of it, always at the same distance. Irrespective of the camera position, the image will always be the same, since the object and the camera move together.

We ask our models to reconstruct the object at the *right* absolute location in 3D space. We train two pairs of models – e_1 and e_2 with a $128{\times}128{\times}256$ grid, occupying the cuboid $(0,0,0)-(1,1,2)$ in space, and e_3 and e_4 with $128{\times}128{\times}384$ grid, occupying $(0,0,0)-(1,1,3)$. Within each pair, we train one model with ray-traced skip connections and one without. Skip connections are the mechanism that makes our models aware of the camera parameters. In all cases we use high realism images and the focal loss. We compare to h_6 from sec. 4.1 in the main paper, which is trained with a 128^3 grid, fixed camera parameters, and otherwise equal settings.

Table 1 summarizes the results. As expected, the models (e_2, e_4) trained without skip connections cannot cope with the task. They need to reconstruct only based on the image, but the same image can lead to multiple reconstructions. The models trained with skip connections (e_1, e_3) performed on-par with the reference model. They received the camera matrix implicitly, encoded in the structure of the skip connections. This shows that our model uses the camera parameters and it can reconstruct the absolute location of the objects, which most previous work could not.

4 MeshRCNN on ShapeNet-triplets

In this section, we perform an exact comparison to MeshRCNN [2]. We use the open source implementation provided by the authors to train and test their model on our ShapeNet-triplet scenes, rendered with high realism. We then use our evaluation procedure on their output predictions. MeshRCNN achieves 3.7%

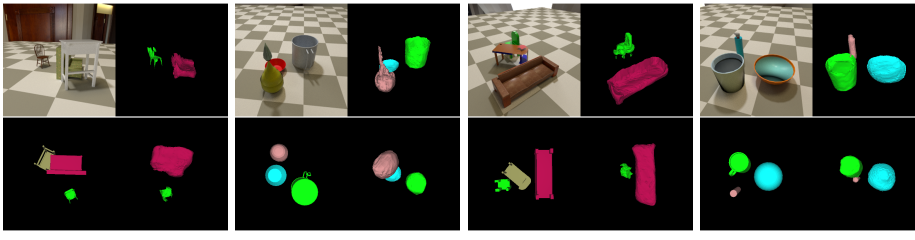

Fig. 5: Scenes reconstructed by MeshRCNN [2]. The top row in each scene shows the input next to a reconstruction from the same view. The bottom row shows the ground-truth next to a reconstruction, both viewed from below. All reconstructions look plausible from the input view, but show pose errors when viewed from below.

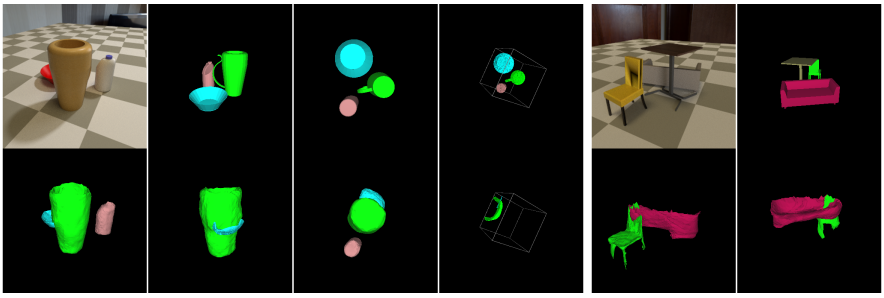

Fig. 6: MeshRCNN reconstruction errors. In the first row of both scenes, the reconstruction looks plausible when viewed from the input camera. In all other rows, we show ground-truth above reconstruction for different view points. In the left scene, Mesh-RCNN fails to hallucinate the occluded parts of the bowl (row 2), predicts objects in wrong positions relative to each other (row 3), and predicts most of the objects outside the volume of scene (row 4). In the right scene, it predicts objects that intersect each other.

mIoU. This is approximately 12 times lower than the performance of model m_9 from sec. 4.3 in the main paper (45.8%[1]).

MeshRCNN underperforms severely on our data. To understand why, we visualized 20 reconstructions (some are shown in fig. 5). In all of them, MeshRCNN produced a reconstruction that is consistent with the input image, when viewed from the input camera. When viewed from other angles however (fig. 6), we noticed that (1) MeshRCNN predicts objects at the wrong depth from the camera most of the time. This leads to a wrong absolute position and the predicted volume often has very low or zero IoU with the ground-truth. This is also acknowledged as a limitation in MeshRCNN [2, Appendix E] and the experiments with quantitative results in MeshRCNN rely on ground-truth depth at test time to correct for it. (2) The relative positions of the objects with respect to each other were often wrong. (3) MeshRCNN sometimes does not resolve occlusions. (4) Reconstructed objects sometimes intersect with each other.

References

1. Chang, A.X., Funkhouser, T.A., Guibas, L.J., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L., Yu, F.: Shapenet: An information-rich 3d model repository. CoRR **abs/1512.03012** (2015), http://arxiv.org/abs/1512.03012 9
2. Georgia Gkioxari, Jitendra Malik, J.J.: Mesh r-cnn. ICCV (2019) 1, 7, 8

[1] We trained MeshRCNN on only 5 of the 6 ShapeNet-triplet classes, due to a mistake in our configuration. Thus, we report averages on only these 5 classes both for m_9 and MeshRCNN. This is also the reason why we report a different mIoU for m_9 here compared to the main paper.

dataset	train	val	test
Number of scenes			
ShapeNet-pairs	320000	45600	91200
ShapeNet-triplets	80000	11400	22800
Fraction of ShapeNet meshes used			
ShapeNet-pairs	98.2%	99.1%	99.3%
ShapeNet-triplets	97.9%	98.1%	97.9%

Table 2: Statistics for ShapeNet-pairs and ShapeNet-triplets. The fraction of ShapeNet [1] meshes used to build the two datasets is measured only on the respective classes.