Energy-Based Residual Latent Transport for Unsupervised Point Cloud Completion (Supplementary Material)

BMVC 2022 Submission # 48

1 Discussion on Impractical Latent-Space EBM Learning

In this section, we discuss why a direct end-to-end training method for latent-space energy-based model (EBM) training is impractical.

To preform gradient descent training, we need gradients for each learnable parameter. Suppose we deploy an EBM (E_θ) in the latent-space of an encoder-decoder architecture. We denote the encoder and decoder as $\mathcal{E}_\alpha$ and $\mathcal{D}_\beta$, respectively, where α and β are the corresponding learnable parameters. Recall that we use Langevin dynamics [4] to sample from the EBM, which is formulated:

$$z^{(t+1)} = z^{(t)} - \frac{\delta^2}{2}\frac{\partial}{\partial z}E_\theta(z^{(t-1)}) + \delta\varepsilon^{(t)}, \tag{1}$$

where t represents Langevin steps, δ is the step size, $\varepsilon^t \sim \mathcal{N}(0,\mathbf{I})$ is Gaussian random noise, and $z^{(t)} \in \mathbb{R}^n$. Subsequently, we can formulate the forward pass as:

$$y = \mathcal{D}_\beta \circ S_\theta^{(T)} \circ S_\theta^{(T-1)} \circ \cdots \circ S_\theta^{(1)} \circ \mathcal{E}_\alpha(x), \tag{2}$$

where $S_\theta^{(t)}$ denotes the t-th Langevin dynamics sampling iteration as shown by Eq. 1 and there are T steps in total. Suppose the loss function is $\mathcal{L}(y)$, the gradient for the loss function w.r.t β can be derived as $\frac{\partial \mathcal{L}(y)}{\partial \beta}$. However, the gradient for updating α involves unfolding the Langevin dynamics as shown by the equation below:

$$\frac{\partial \mathcal{L}(y)}{\partial \alpha} = \frac{\partial \mathcal{L}(y)}{\partial z^{(T)}}\frac{\partial z^{(T)}}{\partial z^{(T-1)}}\cdots\frac{\partial z^{(2)}}{\partial z^{(1)}}\frac{\partial z^{(1)}}{\partial \alpha} \tag{3}$$

Eq. 3 can be expensive to evaluate since it contains a second-order derivative term. This is shown by deriving a general gradient form of a Langevin dynamics step in Eq. 4:

$$\frac{\partial z^{(t)}}{\partial z^{(t-1)}} = \mathbf{I} - \frac{\delta^2}{2}\frac{\partial}{\partial z^2}E_\theta(z^{(t-1)}) \tag{4}$$

where $\mathbf{I}$ is an identity matrix and $\frac{\partial}{\partial z^2}E_\theta(z^{(t-1)})$ is the term that contains a second-order derivative. Since a second-order derivative requires quadratic time and space complexity for evaluation, it can be expensive or even infeasible to compute when the dimension of z^t is large, *e.g.* $n = 2^{10}$. In conclusion, directly back-propagating gradients is impractical for large-scale problems.

2

3 Training Algorithm

We provide pseudo-code to illustrate the training algorithm:

Algorithm 1: Training Algorithm

Input: partial point cloud domain $\mathcal{X}$, complete point cloud domain $\mathcal{Y}$
while *not converged* **do**
  $x \sim \mathcal{X}$
  $y \sim \mathcal{Y}$
  ▷ point cloud encoding
  $z_x \leftarrow \mathcal{E}_\alpha(x)$
  $z_y \leftarrow \mathcal{E}_\alpha(y)$
  ▷ latent transport
  $z^{(0)}_{x\to y} \leftarrow z_x$
  for *sample step t = 1 to T* **do**
    $\tilde{z}^t_{x\to y} \leftarrow \tilde{z}^{t-1}_{x\to y} - \frac{\delta^2}{2}\frac{\partial}{\partial z_{x\to y}} E_\theta(z_{x\to y}) + \delta\varepsilon^t, \quad \varepsilon^t \sim \mathcal{N}(0,\mathbf{I})$
  end
  $r_{xy} \leftarrow z_x - \tilde{z}^t_{x\to y}$
  $\tilde{z}^t_{x\to y} \leftarrow z_x + \Omega(r_{xy})$
  ▷ latent variable decoding
  $\tilde{x} \leftarrow \mathcal{D}_\beta(\tilde{z}_{x\to y})$
  $\tilde{y} \leftarrow \mathcal{D}_\beta(z_y)$
  ▷ update parameters
  $\Delta\alpha \leftarrow \nabla_\alpha \mathcal{L}_{\text{recon}} + \lambda_1 \nabla_\alpha \mathcal{L}_{\text{fidelity}} - \lambda_2 \nabla_\alpha \mathcal{D}_\gamma(\tilde{x})$
  $\Delta\beta \leftarrow \nabla_\beta \mathcal{L}_{\text{recon}} + \lambda_1 \nabla_\beta \mathcal{L}_{\text{fidelity}} - \lambda_2 \nabla_\beta \mathcal{D}_\gamma(\tilde{x})$
  $\Delta\gamma \leftarrow \nabla_\gamma \mathcal{L}_{\text{adv}}$
  $\Delta\theta \leftarrow \nabla_\theta \mathcal{L}_{\text{ebm}}$
  update parameters based on $\Delta\alpha, \Delta\beta, \Delta\gamma, \Delta\theta$ using Adam optimizer
end

4 Implementation of Residual Sampling

We provide PyTorch-style implementation of the proposed residual sampling method:

```
def residual_sampling(z0, energy_function, n_step, step_size):
    z = z0.clone().detach()
    z.requires_grad = True
    with torch.enable_grad():
        for _ in range(n_step):
            noise = step_size * torch.randn_like(z)
            grad = torch.autograd.grad(energy_function(z), z, only_inputs=True)[0]
            z = z - torch.square(step_size) * grad + noise
    z = z0 - z.detach()
    return z
```

5 Inference Cost

We compare inference cost between our method and other methods, *i.e.* Cycle4Completion [6], and ShapeInversion [7]. As shown in Table. 1, ours is much more efficient than Cycle4Completion and ShapeInversion while achieving the best performance in terms of average Chamfer Distance (CD) on the 3D-EPN [3] dataset. Note that our models can perform inference faster than Cycle4Completion by 2.9×10^{-3} per sample and it is over 2000 times faster than ShapeInversion. On the other hand, although our model contains 13M more parameters than Cycle4Completion, our method has a smaller inference time cost. This is because Cycle4Completion contains a time-consuming folding-based decoder [5] while we use an efficient attention-based decoder with a shallow MLP for decoding latent codes.

Table 1: Computational cost on our method compared with existing methods.

Method	#params	Time (s)	Avg. CD ($\times 10^4$)
C4C	21M	9.3×10^{-3}	14.3
ShapeInv.	41M	1.5×10^{1}	23.6
Ours	34M	6.4×10^{-3}	9.4

6 Additional ShapeNet samples

We provide additional ShapeNet completion results in this section. Our model, as well as two existing methods (Cycle4Completion [6] and ShapeInversion [7]), are trained on the 3D-EPN dataset [3]. For each of the eight categories in 3D-EPN dataset, 3 samples are visualized. As illustrated by Figure 1, Figure 2, and Figure 3, our method can produce high-fidelity completion with more details preserved.

7 Real-World samples

We provide qualitative results of our method compared with Cycle4Complete [6] and ShapeInversion [7]. The real-world scans comes from two datasets, *i.e.* ScanNet [2] and MatterPort3D [1].

8 Additional Uncertainty Estimation Samples

This section provides additional uncertainty estimation results of our method as shown in Figure 5 and Figure 6.

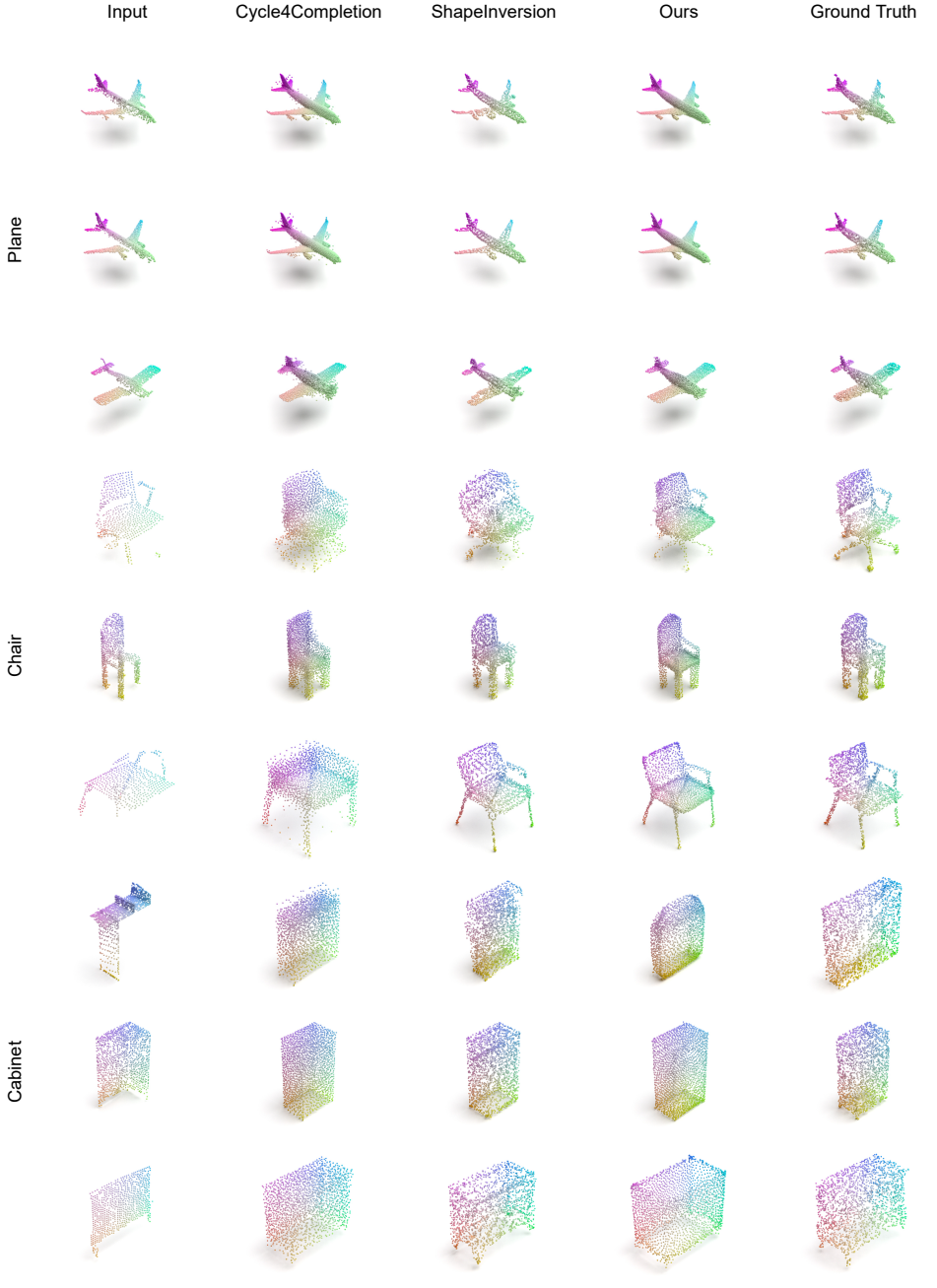

Figure 1: Additional Qualitative Result on the 3D-EPN dataset. From left to right by column: input incomplete point clouds, results from Cycle4Completion [6], ShapeInversion [7], ours, and ground truth.

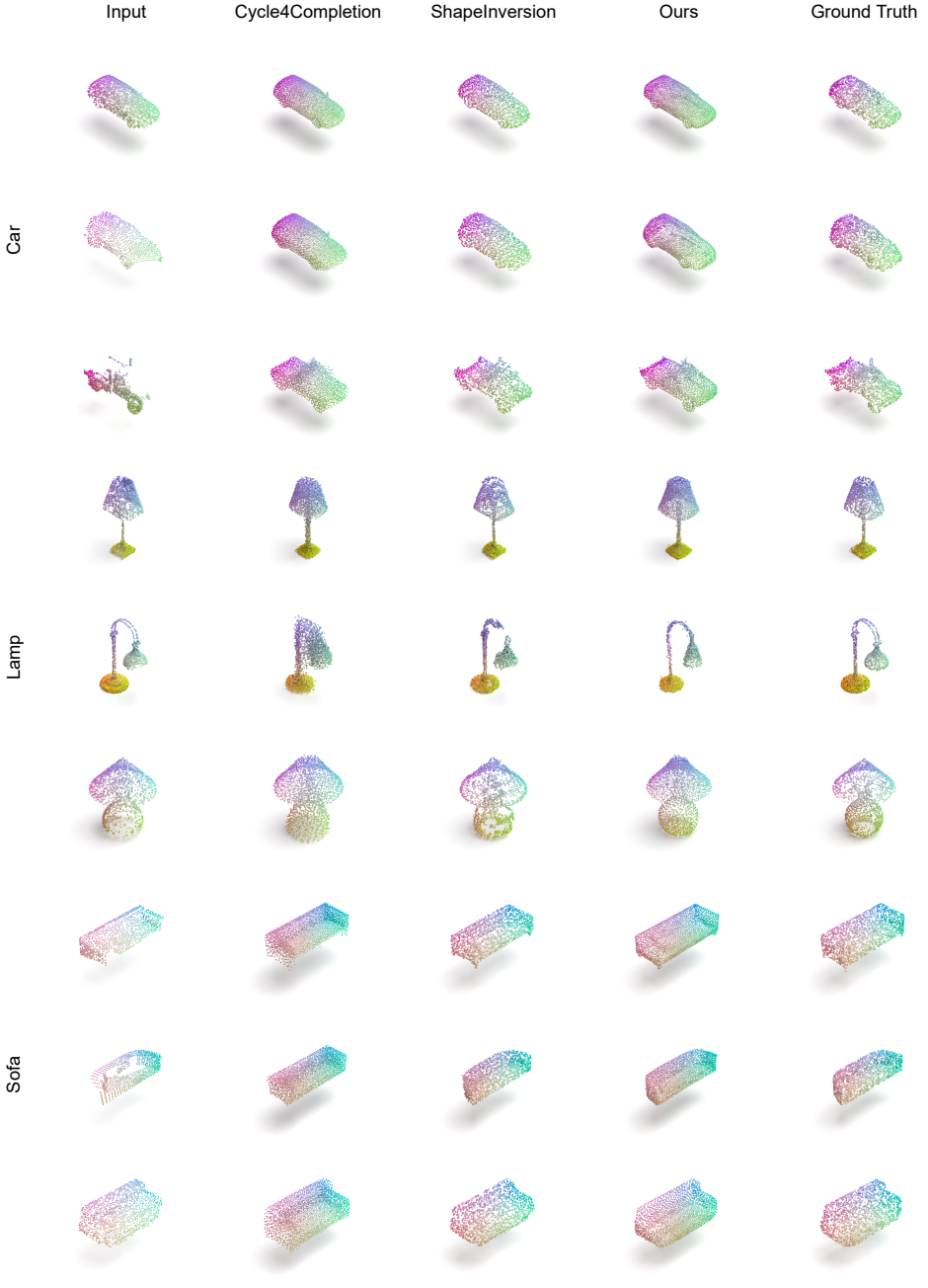

Figure 2: Additional Qualitative Results on the 3D-EPN dataset (continued 1). From left to right by column: input incomplete point clouds, results from Cycle4Completion, ShapeInversion, ours, and ground truth.

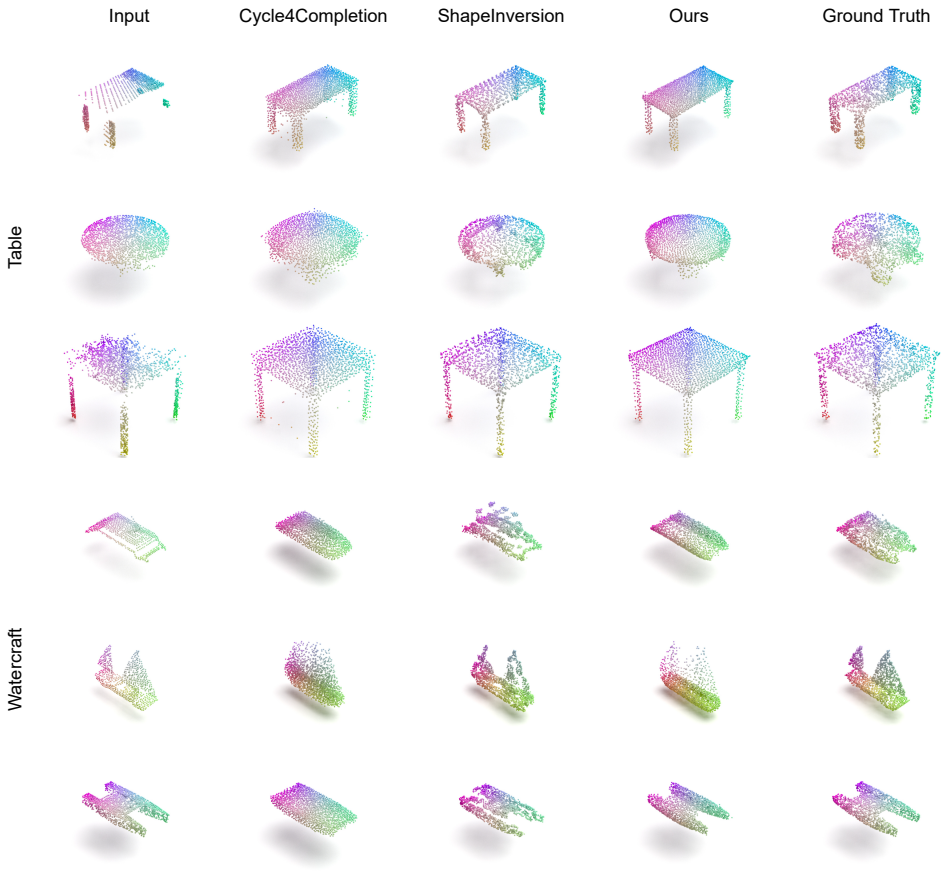

Figure 3: Additional Qualitative Results on the 3D-EPN dataset (continued 2). From left to right by column: input incomplete point clouds, results from Cycle4Completion, ShapeInversion, ours, and ground truth.

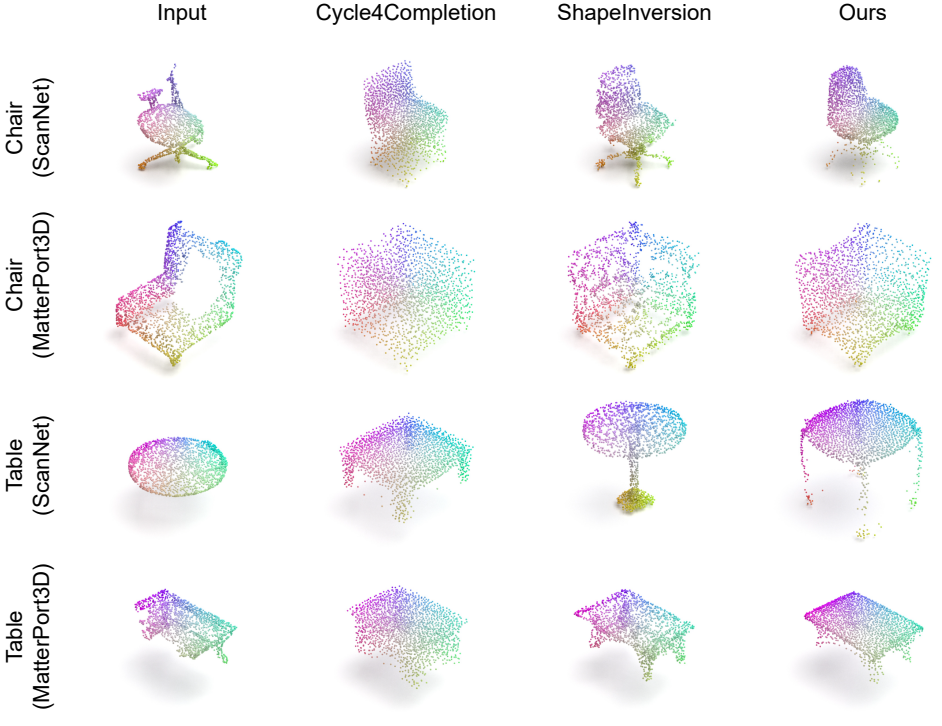

Figure 4: Qualitative Result on the real-world dataset. From left to right by column: incomplete point clouds, results from Cycle4Completion, ShapeInversion, and ours.

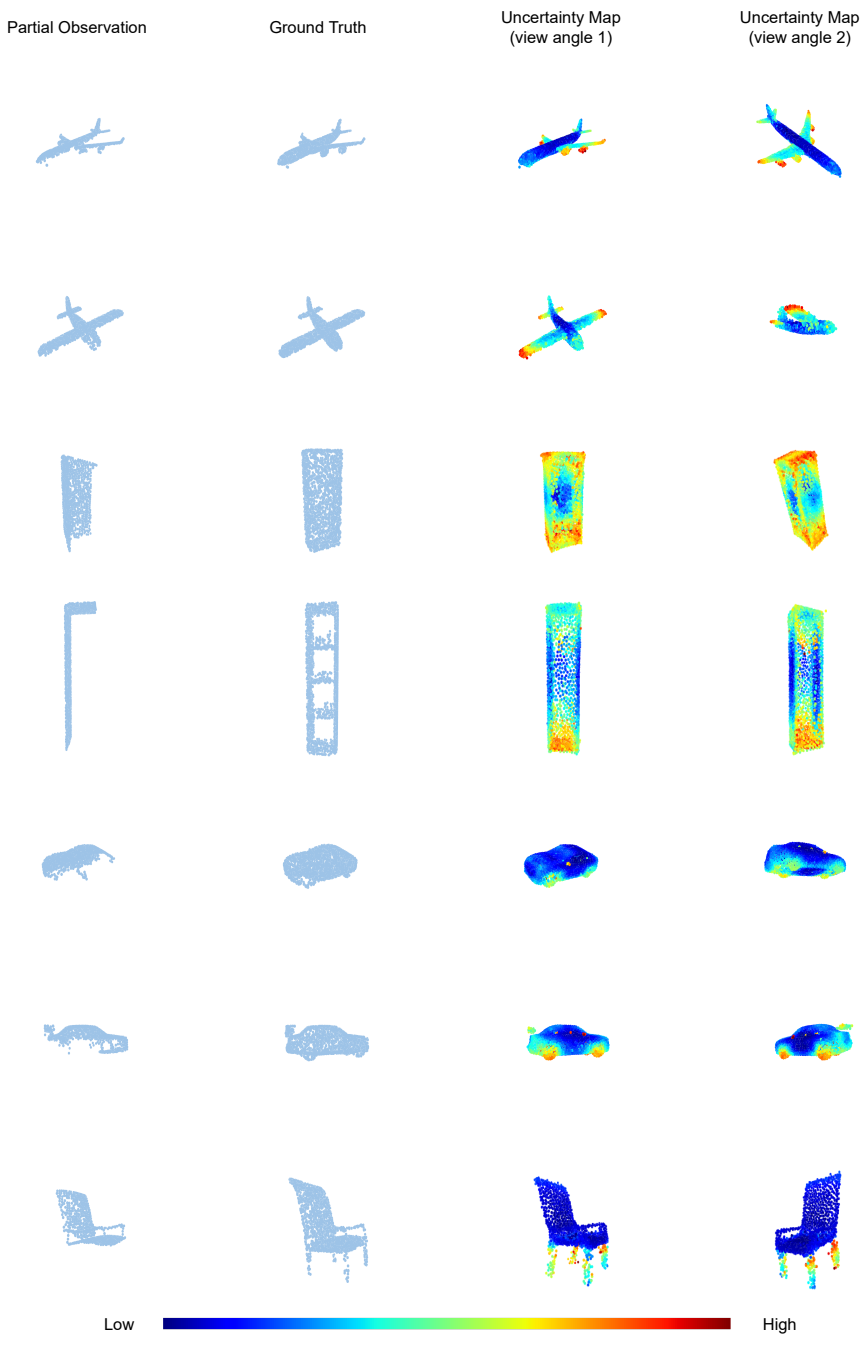

Figure 5: Additional uncertainty maps produced by our method. The input and ground truth are shown in the first and second rows followed by two viewing angles of their uncertainty map.

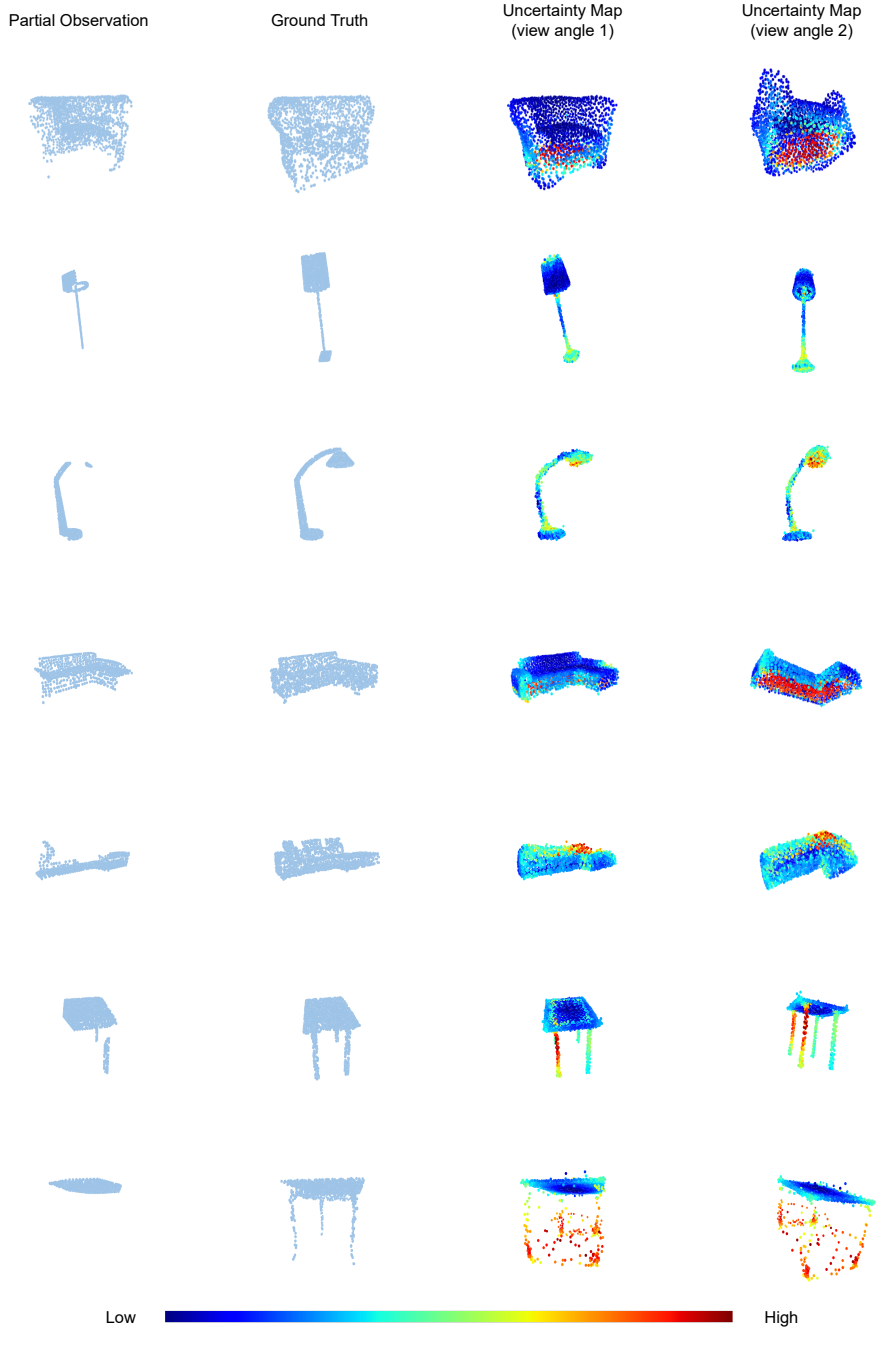

Figure 6: Additional uncertainty maps produced by our method (continued). The input and ground truth are shown in the first and second rows followed by two viewing angles of their uncertainty map.

References

[1] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. *International Conference on 3D Vision (3DV)*, 2017.

[2] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017.

[3] Angela Dai, Charles Ruizhongtai Qi, and Matthias Nießner. Shape completion using 3d-encoder-predictor cnns and shape synthesis. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017.

[4] Yilun Du and Igor Mordatch. Implicit generation and modeling with energy based models. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[5] Lyne P. Tchapmi, Vineet Kosaraju, Hamid Rezatofighi, Ian Reid, and Silvio Savarese. Topnet: Structural point cloud decoder. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 383–392, 2019. doi: 10.1109/CVPR.2019.00047.

[6] Xin Wen, Zhizhong Han, Yan-Pei Cao, Pengfei Wan, Wen Zheng, and Yu-Shen Liu. Cycle4completion: Unpaired point cloud completion using cycle transformation with missing region coding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[7] Junzhe Zhang, Xinyi Chen, Zhongang Cai, Liang Pan, Haiyu Zhao, Shuai Yi, Chai Kiat Yeo, Bo Dai, and Chen Change Loy. Unsupervised 3d shape completion through gan inversion. In *CVPR*, 2021.