Supplementary material

Point Cloud Upsampling via Disentangled Refinement

CVPR 2021

Overview

This supplemental material consists of the following sections:

- In Section A, we provide more visual comparisons with the SOTA methods on real-scanned inputs.
- In Section B, we provide more visual comparisons with the SOTA methods on synthetic inputs.
- In Section C, we present more results on input point sets of varying noise levels.
- In Section D, we show results on input point sets of varying sizes.
- In Section E, we show the evaluation curves in terms of the training epochs for the coarse outputs $\mathcal{Q}'$ and the refined outputs $\mathcal{Q}$ (see Section 3.1 in main paper).

A. More Visual Comparisons on Real-scanned Inputs

Figures 1-5 demonstrate more visual comparisons on real-scanned point clouds. We apply our method and two state-of-the-art methods, *i.e.*, MPU [2] and PU-GAN [1], to upsample points, as well as reconstruct the associated surfaces from the dense outputs. Clearly, our method outperforms others on the local uniformity, contributing to high-quality reconstructed surfaces.

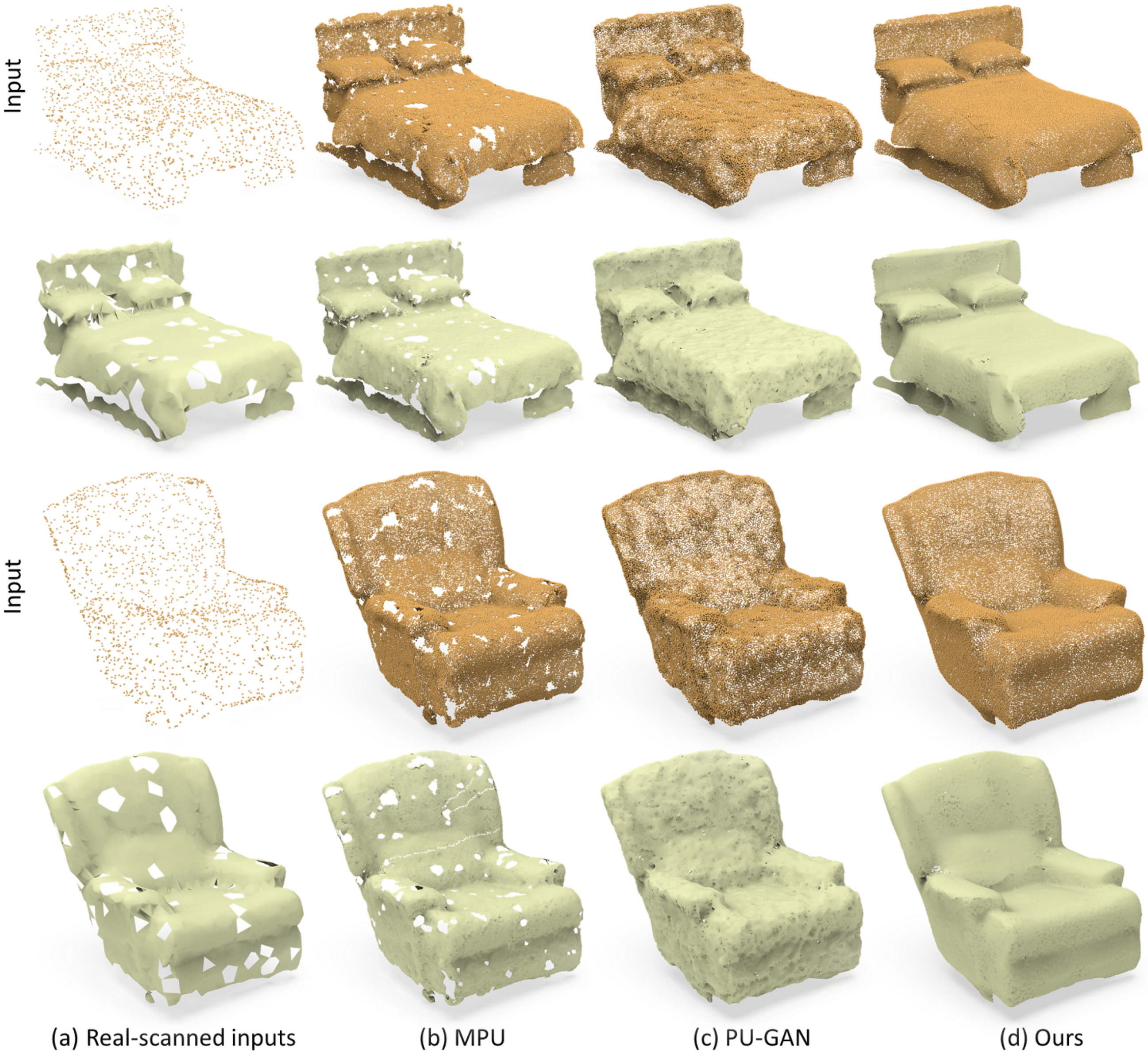


Figure 1. Comparing point set upsampling (16×) results and reconstructed 3D meshes using different methods (b-d) from real-scanned sparse inputs (a). For each object, the top row shows point clouds, while the bottom row shows the reconstructed meshes. Clearly, our method outperforms others on the local uniformity, contributing to high-quality reconstructed surfaces (1/5).

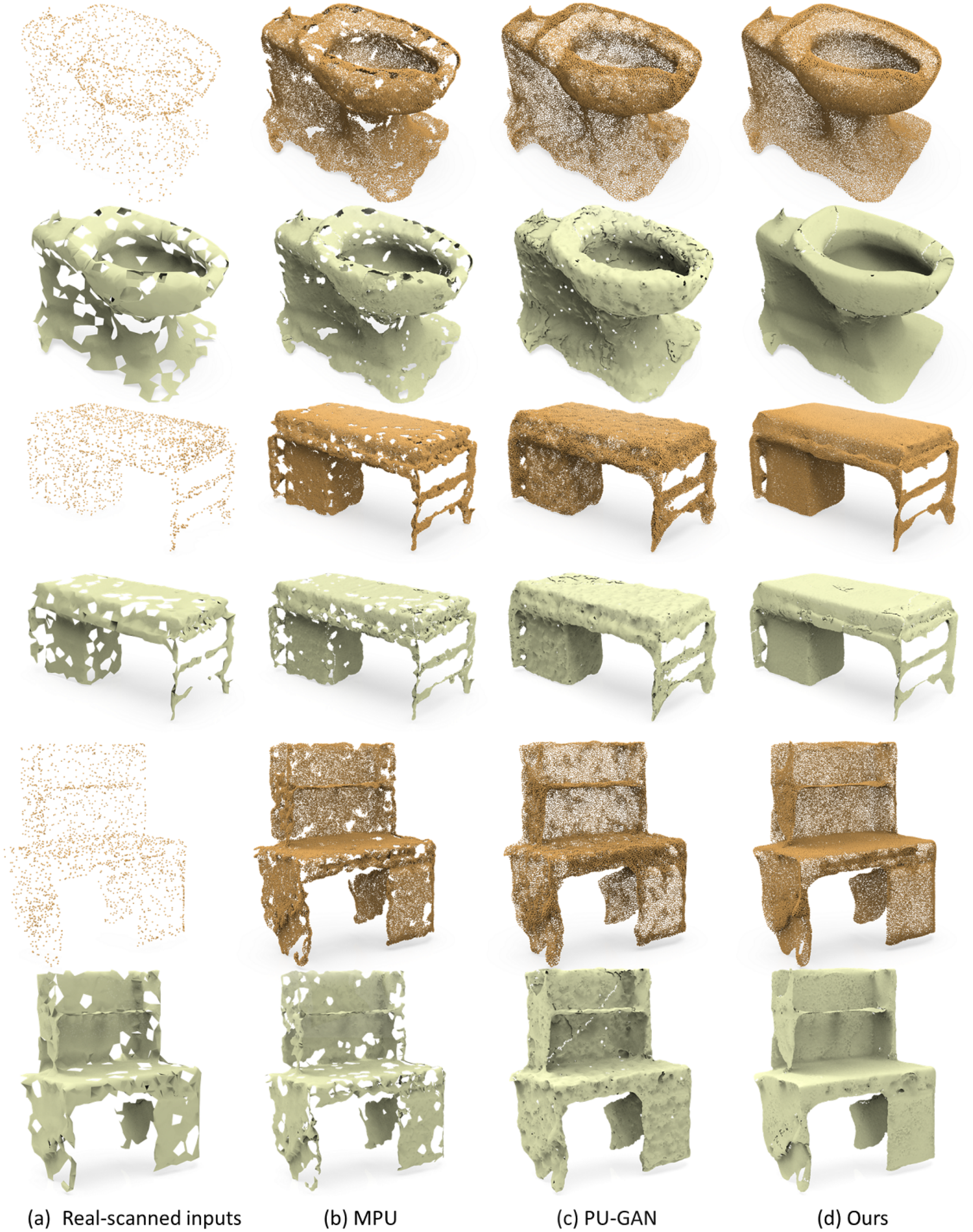


Figure 2. Comparing point set upsampling (16×) results and reconstructed 3D meshes using different methods (b-d) from real-scanned sparse inputs (a). For each object, the top row shows point clouds, while the bottom row shows the reconstructed meshes. Clearly, our method outperforms others on the local uniformity, contributing to high-quality reconstructed surfaces (2/5).

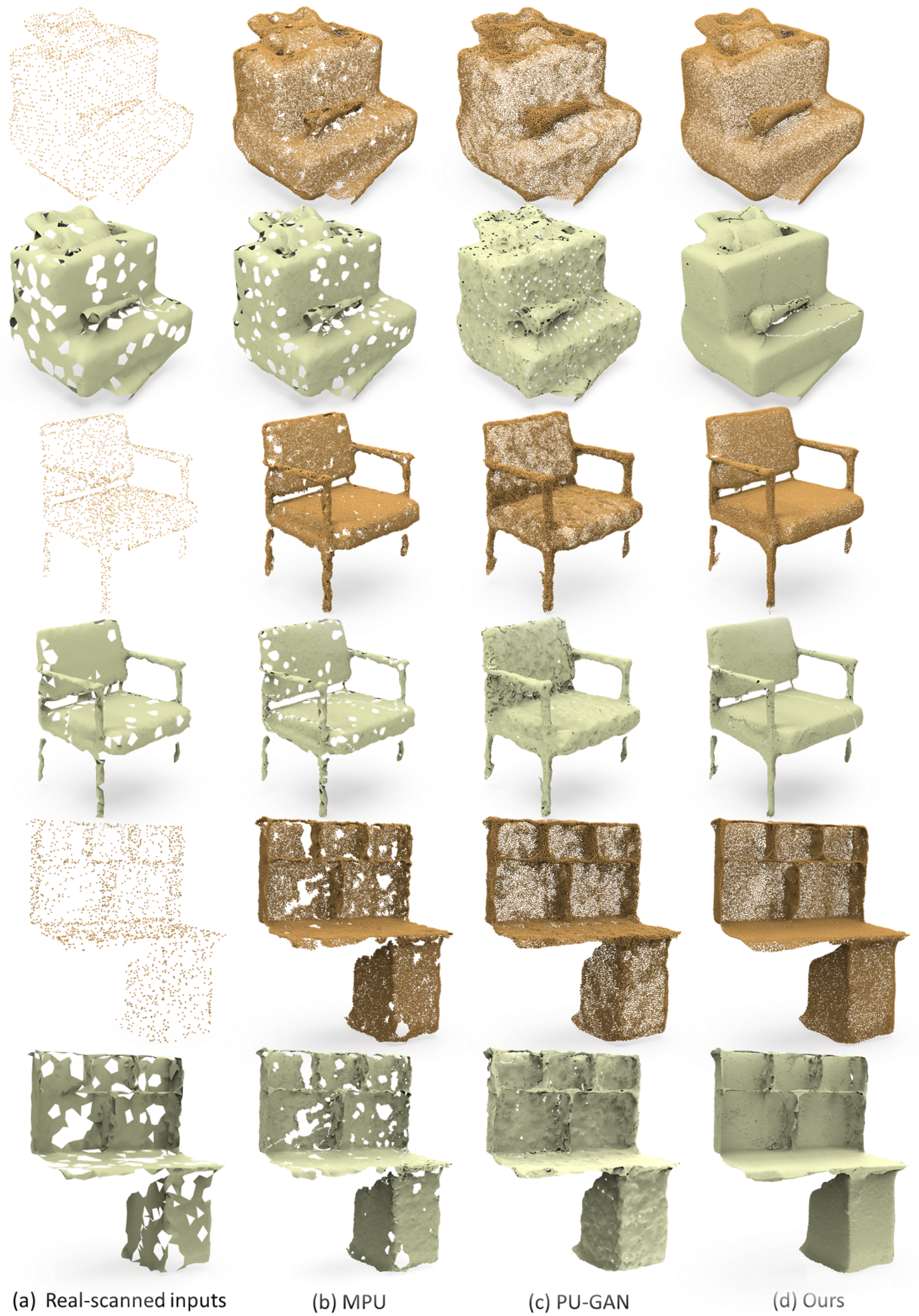


Figure 3. Comparing point set upsampling (16×) results and reconstructed 3D meshes using different methods (b-d) from real-scanned sparse inputs (a). For each object, the top row shows point clouds, while the bottom row shows the reconstructed meshes. Clearly, our method outperforms others on the local uniformity, contributing to high-quality reconstructed surfaces (3/5).

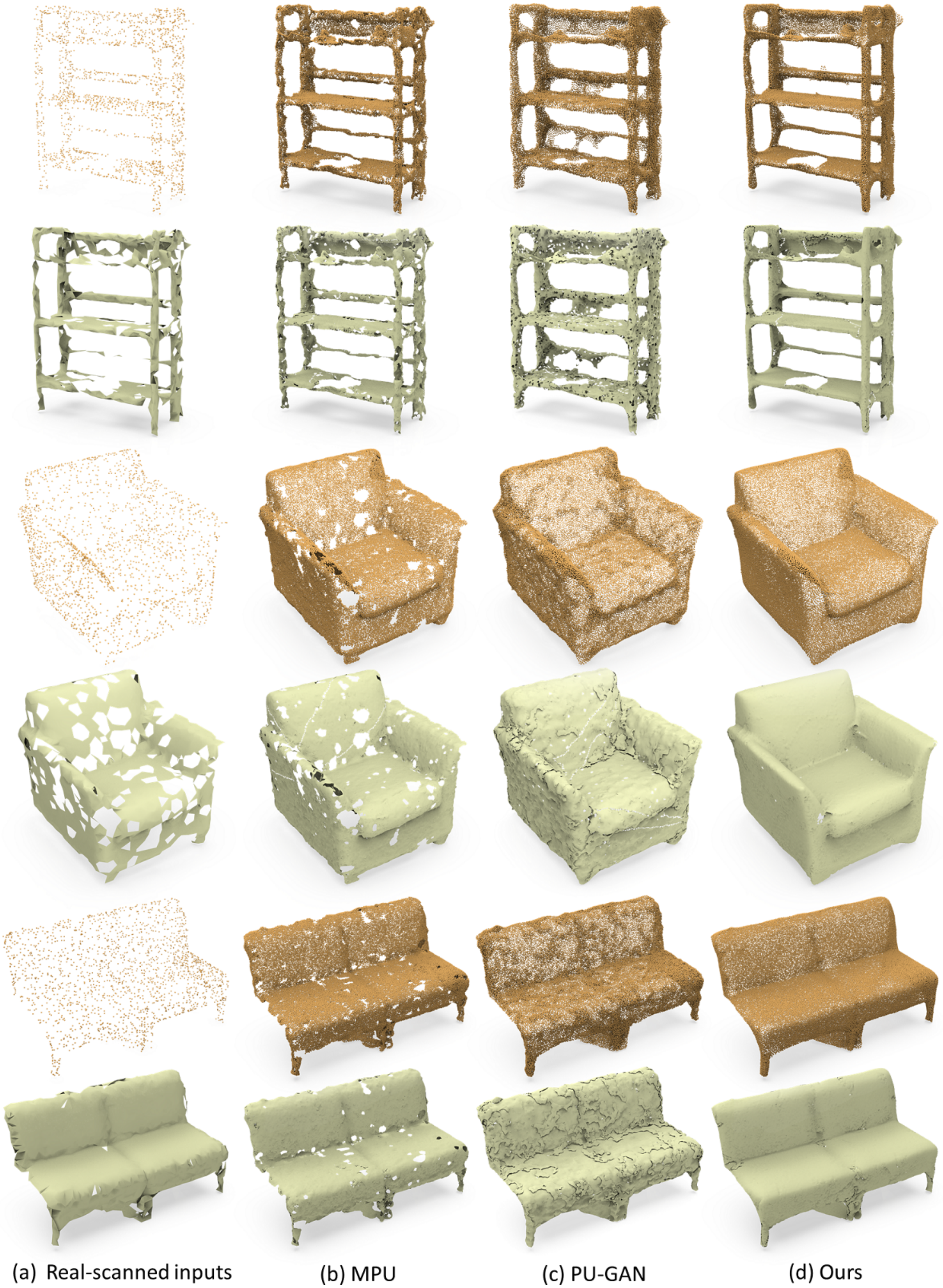
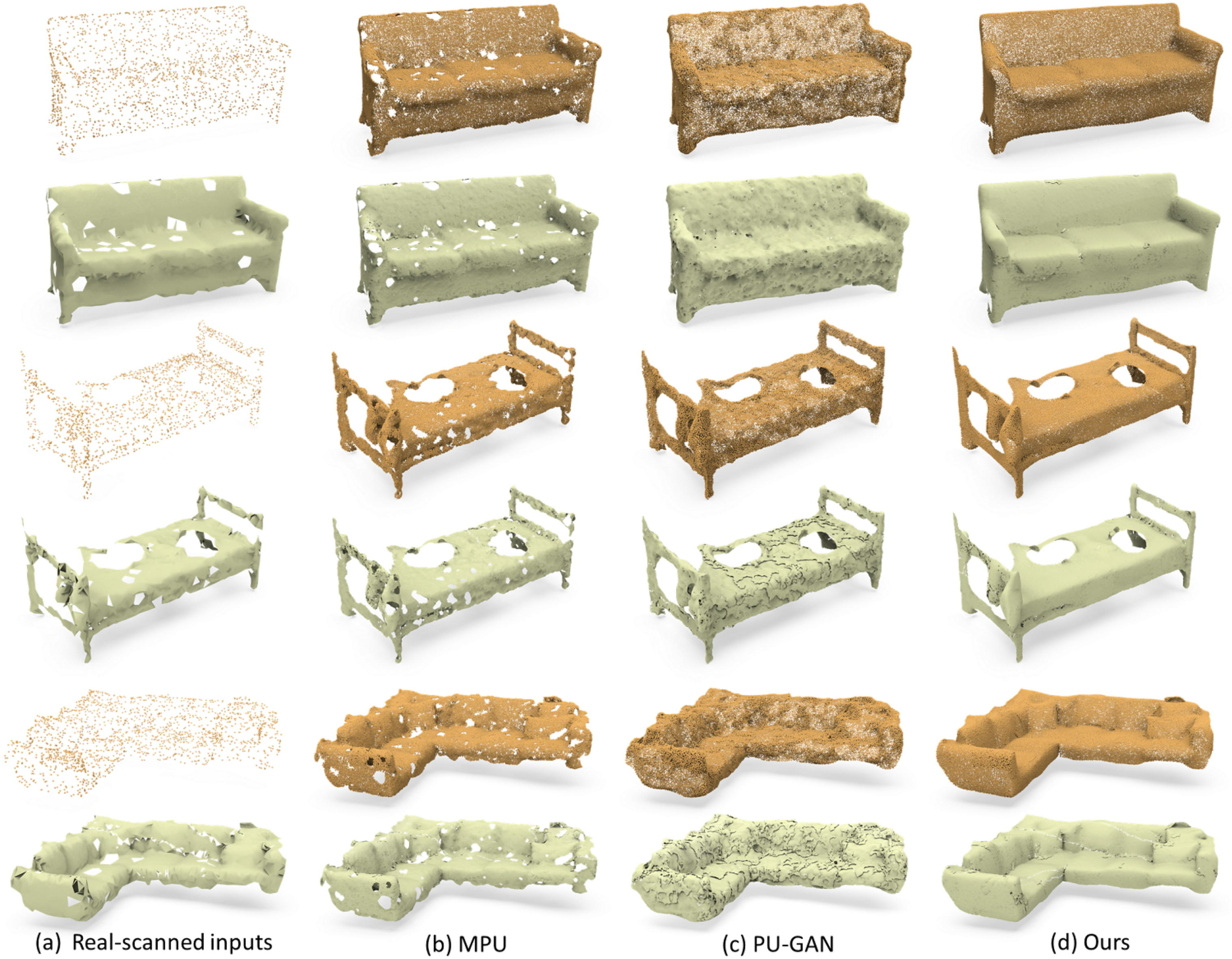


Figure 4. Comparing point set upsampling (16×) results and reconstructed 3D meshes using different methods (b-d) from real-scanned sparse inputs (a). For each object, the top row shows point clouds, while the bottom row shows the reconstructed meshes. Clearly, our method outperforms others on the local uniformity, contributing to high-quality reconstructed surfaces (4/5).



(a) Real-scanned inputs

(b) MPU

(c) PU-GAN

(d) Ours

Figure 5. Comparing point set upsampling (16×) results and reconstructed 3D meshes using different methods (b-d) from real-scanned sparse inputs (a). For each object, the top row shows point clouds, while the bottom row shows the reconstructed meshes. Clearly, our method outperforms others on the local uniformity, contributing to high-quality reconstructed surfaces (5/5).

B. More Visual Comparisons on Synthetic Inputs

Figures 6-7 demonstrate more visual comparisons on synthetic sparse inputs with three state-of-the-art methods, *i.e.*, PU-Net [3] MPU [2], and PU-GAN [1]. For each object, We show both the upsampled dense points and the associated error map. Clearly, our method produces results that are visually most similar to the target points (bottom row in (a)) with the lowest errors, and our dense points can well preserve tiny local structures with a uniform point distribution; see particularly the blown-up views.

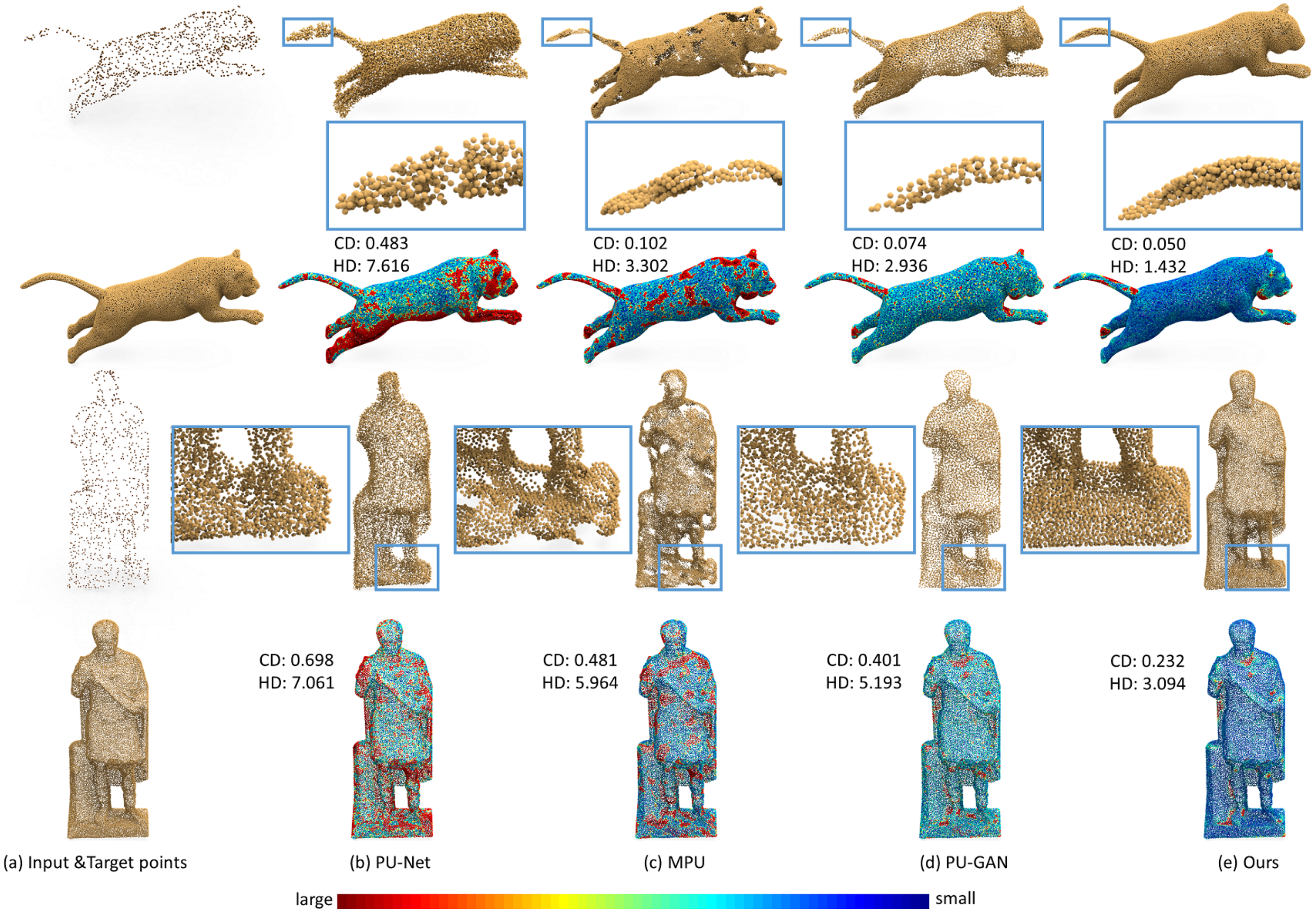


Figure 6. Comparing point set upsampling (16×) results from synthetic sparse inputs (a) using different methods (b-e). We also show the associated error maps, where the colors reveal the nearest distance for each target point to the predicted point set generated by each method. (1/2).

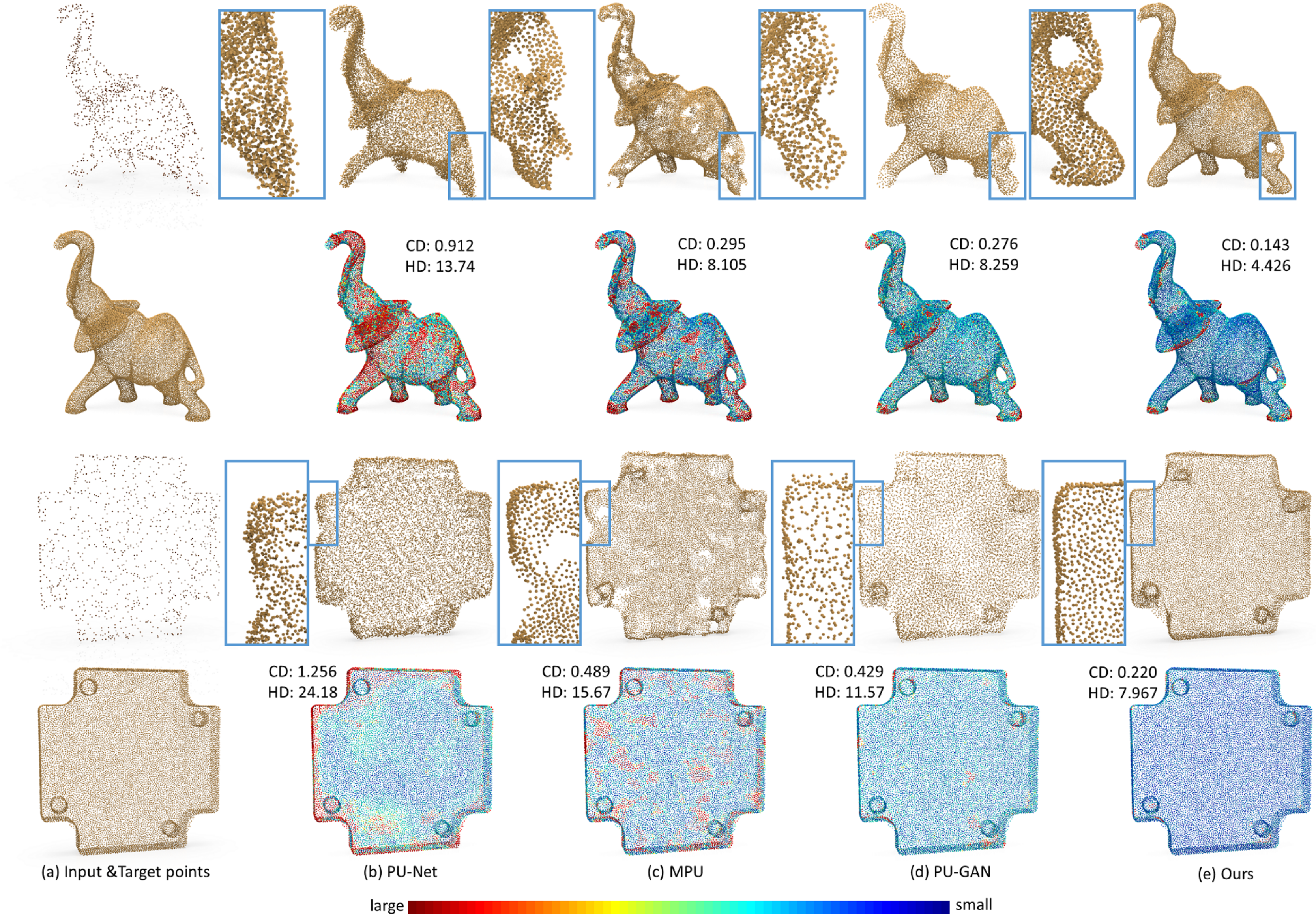


Figure 7. Comparing point set upsampling (16×) results from synthetic sparse inputs (a) using different methods (b-e). We also show the associated error maps, where the colors reveal the nearest distance for each target point to the predicted point set generated by each method. (2/2).

C. More Upsampling Results on Inputs of Varying Noise Levels

Figure 8 shows more visual comparisons of using our method, MPU [2], and PU-GAN [1] to upsample sparse inputs that are corrupted by Gaussian noise of increasing levels. Clearly, our method achieves more uniform upsampling results without excessive noise, under both upsampling rates.

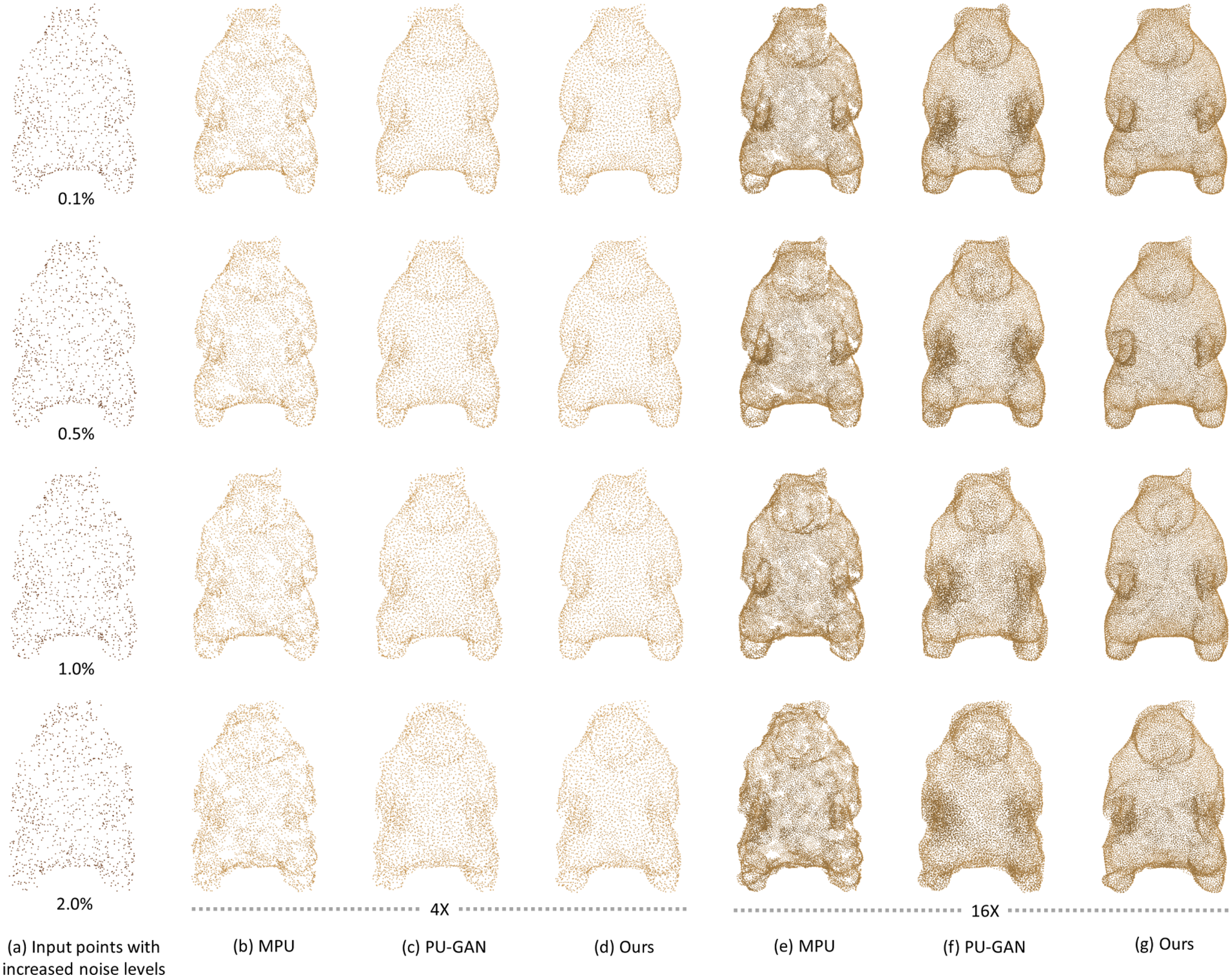


Figure 8. Comparing point set upsampling results produced using different methods under different upsampling rate, when given noisy sparse inputs with increasing noise level, *i.e.*, 0.1%, 0.5%, 1.0%, and 2.0%.

D. Upsampling Input Point Sizes of Varying Sizes

Figure 9 shows the upsampling results by applying our method to upsample inputs of decreasing numbers of points. From these results, we can see that our method produces stable performance without introducing noise into the results, even for inputs of very small number of points.

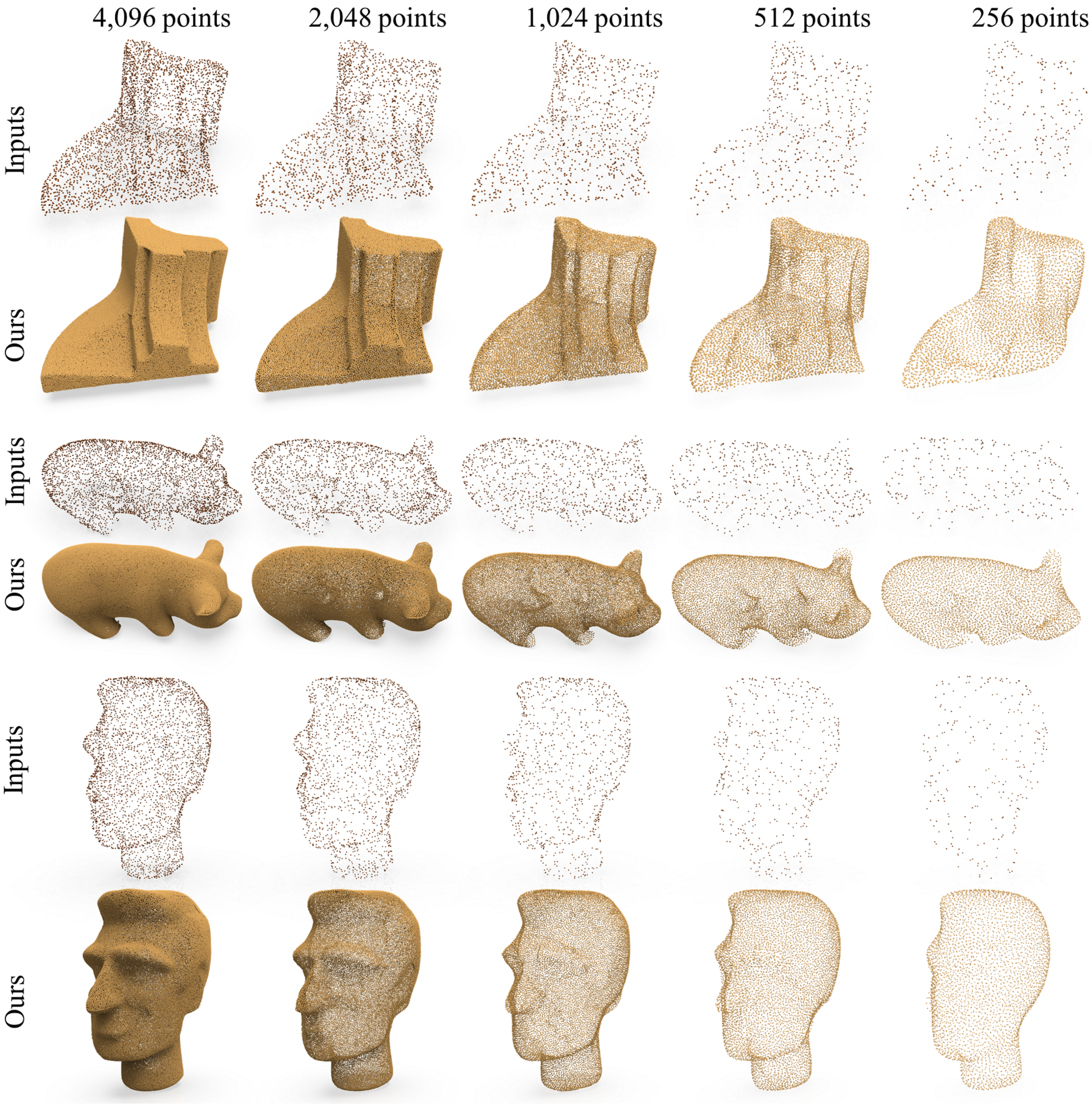


Figure 9. Upsampling from inputs of varying sizes.

E. Evaluation Curves of $\mathcal{Q}'$ and $\mathcal{Q}$

Figure 10 plots the evaluation curves (16×) in terms of the training epochs for the coarse outputs $\mathcal{Q}'$ (blue) and the refined results $\mathcal{Q}$ (orange); see Section 3.1 in main paper. At the early training stage, our framework first focuses more on $\mathcal{Q}'$ (with a relatively larger weight), so $\mathcal{Q}'$ initially has a lower CD value, but as the training progresses, $\mathcal{Q}$ rapidly improves and surpasses $\mathcal{Q}'$ as shown in the plots, and both eventually converge to a similar level. This shows the complementary strengths of our dense generator and spatial refiner, and verifies the effectiveness of our disentangled design.

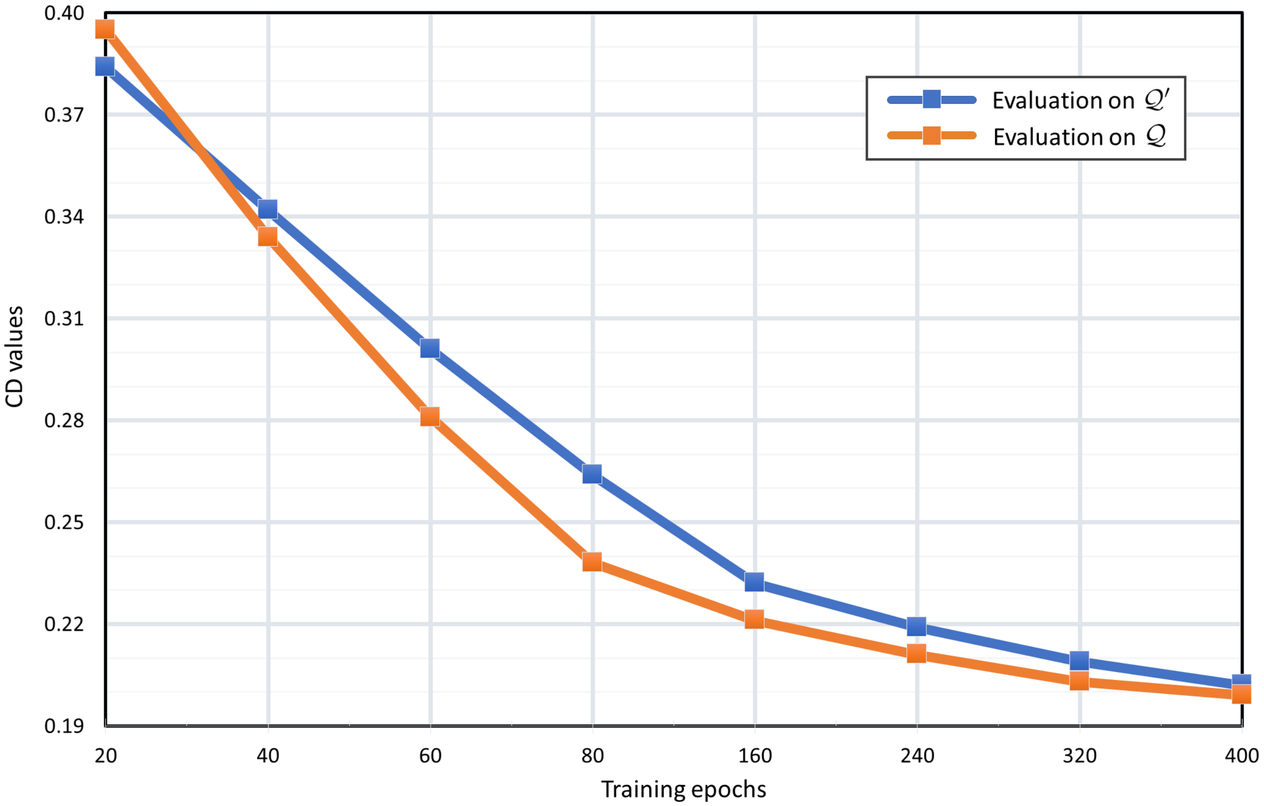


Figure 10. Evaluation curves in terms of the CD values over the training epochs for the coarse outputs $\mathcal{Q}'$ (blue) and refined results $\mathcal{Q}$ (orange). The unit of CD is 10^{-3}.

References

[1] Ruihui Li, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng. PU-GAN: A point cloud upsampling adversarial network. In *IEEE Int. Conf. on Computer Vision (ICCV)*, pages 7203–7212, 2019. 2, 7, 9

[2] Wang Yifan, Shihao Wu, Hui Huang, Daniel Cohen-Or, and Olga Sorkine-Hornung. Patch-based progressive 3D point set upsampling. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 5958–5967, 2019. 2, 7, 9

[3] Lequan Yu, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng. PU-Net: Point cloud upsampling network. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2790–2799, 2018. 7

The End