

Supplementary Material for Stable-Sim2Real

Outline

This supplementary document is arranged as follows:

- (1) Sec. **A** illustrates more qualitative comparisons of 3D instance reconstruction and depth/3D simulation;
- (2) Sec. **B** presents a user study of 3D simulation;
- (3) Sec. **C** provides more ablation studies of our method;
- (4) Sec. **D** elaborates our method implementations;
- (5) Sec. **E** discusses more potentials and limitations of our method.

A. More Qualitative Results

A.1. 3D Instance Reconstruction

Fig. **I** presents more qualitative comparisons of 3D instance reconstruction on LASA validation set.

Consequently, our method consistently achieves remarkable 3D instance reconstruction results across diverse objects, from coarse to fine geometry. Moreover, given the reconstruction model DisCo is a diffusion-based model, we observe that our simulated data can enhance DisCo’s proficiency in generating novel yet plausible geometries, such as the creation of a blanket on a bed.

We provide a 3D animated visualization of the qualitative comparison in a visually appealing video format, where we rotate the reconstructed mesh. You may refer to the supplementary file folder and access the video file named *lasa_reconstruction.mp4* to view our qualitative results.

A.2. Depth/3D Simulation

In Fig. **II**, we present additional qualitative results of depth/3D simulation on LASA [4] validation set. The 3D animated visualization of the qualitative comparison is presented in *lasa_sim2real.mp4*.

Furthermore, Fig. **III** shows more depth/3D simulation results on ShapeNet [1]. The 3D animated visualization of the qualitative comparison is presented in *shapenet_sim2real.mp4*.

B. User Study of Sim2Real

To supplement the evaluation of our 3D simulation quality in the main paper, we conduct a user study. We invited 17 subjects through an online questionnaire. All participants

Method	Cycle-GAN [11]	Pix2Pix [2]	Giga-GAN [3]	SD [9]	Ours	Real GT
Picked ratio (%)	22.5	33.1	55.4	62.7	88.6	94.8

Table I. The quantitative results of **user study**.

are *experienced* computer vision researchers, including senior professors or students. Half of them are experts in 3D vision, and none of them had seen our results before. To begin with, we first randomly select a set of 20 real-captured ground truths (GT) from LASA [4] validation set in 3D format, with the synthetic data as a reference, and instruct the participants to get familiar with the data pattern of GT.

During the evaluation, we randomly chose 10 samples each from the simulated data produced by Cycle-GAN [11], Pix2Pix [2], Giga-GAN [3], Stable-Diffusion [9] our method, and the real-captured GT. Subsequently, these 60 (10x6) samples were mixed together and presented to the participants. Each participant was tasked to *pick* the samples they perceived to be real captures based on their impressions and understanding of the initial samples. For each method, we calculate the ratio of the quantity picked as real captures among all the presented samples produced by these methods. As shown in Tab. **I**, our method surpasses other methods and approximates the result of real GT.

C. More Ablation Studies

We follow the main paper to conduct ablation studies on the 3D instance reconstruction task, specifically on the table category, using the zero-shot evaluation scheme.

3D binary classifier. We evaluate two key factors in the implementation of our 3D binary classifier, the size of split patches in the depth map and the network channel of the point classifier.

Tab. **II** reports the results using different patch sizes, where the classification accuracy of the binary classifier is also included. The best results are achieved with a 128x128 patch size, while using 64x64 or 256x256 sizes marginally reduces performance. Employing a 32x32 patch size results in very small projected point patches, making it challenging for the binary classifier to distinguish effectively, ultimately resulting in the poorest performance.



Figure I. Qualitative comparison on 3D instance reconstruction, using LASA [4] training data mixed with simulated data generated by different methods. “Rand.” means random perturbation. “w/o sim.” denotes using LASA training data without simulated data.

Tab. III presents the results of using different channels (*i.e.*, network width of each layer) in the 3D binary classifier (*i.e.*, PointNet [8]). Similarly, the binary classification accuracy is also reported. Consequently, setting the width to 64 yields the best result, while using 32 or 128 slightly degrades the performance. However, using either 16 or 256

may build a very weak or strong classifier, ultimately leading to inferior results.

Re-weighting diffusion loss. In Stage II of our framework, we re-weight the diffusion loss to prioritize the optimization of unsatisfactory areas. ω and λ are weight coefficients.

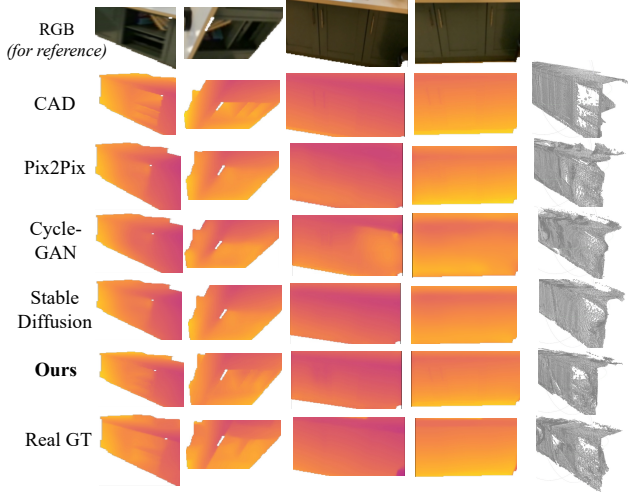


Figure II. Qualitative comparison of the depth/3D simulation results on LASA [4] validation set.

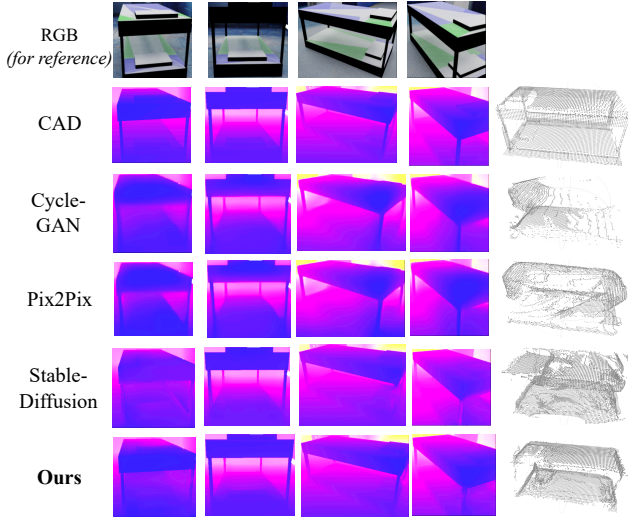


Figure III. Qualitative comparison of the depth/3D simulation results on ShapeNet [1].

Patch size	32x32	64x64	128x128	256x256
Acc (%)	42.7	65.4	67.8	70.2
Metrics	27.1/16.9/22.6	28.3/16.4/24.3	28.5/ 16.1/24.7	28.2/16.5/24.4

Table II. Quantitative ablations of the patch size to split the depth map. Evaluation metrics are **mIoU / Chamfer L2 / F-score** of 3D instance reconstruction respectively.

cients of similar regions and distinct (*i.e.*, unsatisfactory) areas, respectively, in which $\lambda > \omega$. Tab. IV presents the ablations of these two coefficients. Setting $\omega = 0.5$ and $\lambda = 1.5$ results in the optimal outcome, while other reasonable configurations also consistently deliver good results. Another

Width	16	32	64	128	256
Acc (%)	41.5	59.8	67.8	78.5	88.6
Metrics	26.3/17.2/22.8	28.3/16.4/24.2	28.5/ 16.1/24.7	28.1/16.7/23.9	26.4/18.3/22.5

Table III. Quantitative ablations of the network width (“Width”) in the 3D binary classifier. Evaluation metrics are **mIoU / Chamfer L2 / F-score** of 3D instance reconstruction respectively.

ω/λ	0.3/1.7	0.4/1.6	0.5/1.5	0.6/1.4	Conf.
Metrics	28.1/16.1/24.5	28.4/16.5/24.7	28.5/16.1/24.7	28.8/15.9/24.3	27.8/15.2/23.9

Table IV. Quantitative ablations of the weight coefficients to re-weight denoising loss. Evaluation metrics are **mIoU / Chamfer L2 / F-score** of 3D instance reconstruction respectively.

alternative way is to use the output confidence value from the binary classifier as the soft weight. However, we find that this operation may degrade the performance (Tab. IV: Conf.), which is probably due to its *less control* over the loss weight compared to directly setting weight.

Why simulating 3D via 2D? In fact, our pipeline exactly *mimics the real 3D data acquisition* where individual 2D depth maps (with noise) from different views are captured and fused into 3D. As for depth fusion, since our method simulates and adds *noise* to *view-consistent* rendered synthetic depth inputs, the view-variation of the resulting simulated depth maps *remains small*, making the simulated depth can always be effectively fused into 3D data in our experiment.

We also finetuned a recent 3D diffusion, DiT-3D [6] * to simulate 3D data for 3D instance reconstruction. It gets 26.9/19.1/22.3 mIoU/CD/F-Score on the table category under the zero-shot evaluation scheme, which is much worse than our method (28.5/16.1/24.7). This verifies the necessity to leverage 2D priors to mitigate the issue of 3D data scarcity.

D. Network and Implementation Details

Details of different simulation methods. For both CycleGAN [11] and Pix2Pix [2], we use their official PyTorch [7] code †, similarly for Giga-GAN ‡ and Stable-Diffusion §. In our approach, in both Stage I and II, we fine-tune the pre-trained Stable Diffusion V2.1 § [9] using the AdamW optimizer [5] with a fixed learning rate of 3e-5. To align with the VAE’s expected input range, we normalize all input maps to the range [-1, 1]. During training, we

* <https://github.com/DiT-3D/DiT-3D>

† <https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix>

‡ <https://github.com/lucidrains/gigagan-pytorch>

§ <https://huggingface.co/stabilityai/stable-diffusion-2-1>

employ random crops with various aspect ratios and pad the images to a resolution of 512x512 using black padding. The batch size is set to 48 for approximately 20,000 steps. The entire training process spans about two days utilizing four A100 GPUs. Our network architecture is based on the building blocks of ControlNet [10].

Details of 3D Binary Classifier. We use the original PointNet [8] as our 3D binary classifier, which is a network with several multi-layer-perceptrons (MLPs) with a max-pooling operation to extract the final global feature vector. During training, we also extract the feature of synthetic point patches using one single-layer MLP and a max-pooling layer. To provide guidance, the extracted synthetic features are concatenated with the global feature of real-captured or generated point patches, forming the final feature for optimization and prediction. The training strategy also follows PointNet’s original setting.

E. More Discussions

E.1. Potentials

Using our model for: denoise VS add-noise. Similar to the setting of conventional 2D tasks, we further apply our model to take noisy depth as inputs and *denoise* them into clean depth, and find that using one-stage diffusion is *sufficient* (PSNR: One-stage 42.9 VS Two-stage 43.2). To clarify, as highlighted in the main paper, different from conventional 2D tasks which often remove noise from the input into a clean output, our special task poses a *unique challenge* about adding noise to generate noisy output. To this end, our method is *customized* to tackle this challenge, and applying it to traditional 2D tasks may *not fully verify* its unique value. Conversely, our work potentially opens a new task of simulating real-world image/RGB noise/degradation.

Still benefit denoise. From another perspective of data, our clean-to-noisy method exactly *serves to improve* the inverse solution of noisy-to-clean: To train a noisy-to-clean model, it still requires large-scale clean-noisy *paired data*. To gain/scale up such paired data, our method actually serves as a reasonable solution, which generates real/noisy data (*hard to collect*) from synthetic input (*easy to gain*). In fact, this logic has been verified by our experiment of 3D instance reconstruction, where our clean-to-noisy method generates clean-noisy paired data that are used to train a better noisy-to-clean (*i.e.*, recovering clean surface from noisy observation) model. Our method fills/complements the *closed loop* of clean-noisy-clean.

E.2. Limitations

As our model is finetuned on a fixed dataset (*i.e.*, LASA [4]), transferring our model to other sensor data or new domains of objects/scenes that have a distinct gap with LASA may require re-training the model currently, and we consider this as an open question for future exploration. Additionally, we expect new paired datasets to explore our data-driven simulation in the future.

Moreover, the current two-stage pipeline is complex, and a *one-stage* pipeline could be achieved by training the *generator* and our 3D *classifier simultaneously* in an *adversarial* manner. We leave this for the future work.

Last, our model relies on *paired* data, yet we believe strong priors inherited from large pretrained models may reduce such reliance. This remains an open problem for future work.

References

- [1] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 1, 3
- [2] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017. 1, 3
- [3] Minguk Kang, Jun-Yan Zhu, Richard Zhang, Jaesik Park, Eli Shechtman, Sylvain Paris, and Taesung Park. Scaling up gans for text-to-image synthesis. In *CVPR*, 2023. 1
- [4] Haolin Liu, Chongjie Ye, Yinyu Nie, Yingfan He, and Xiaoguang Han. Lasa: Instance reconstruction from real scans using a large-scale aligned shape annotation dataset. In *CVPR*, 2024. 1, 2, 3, 4
- [5] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2019. 3
- [6] Shentong Mo, Enze Xie, Ruihang Chu, Lewei Yao, Lanqing Hong, Matthias Nießner, and Zhenguo Li. Dit-3d: Exploring plain diffusion transformers for 3d shape generation. *arXiv preprint arXiv: 2307.01831*, 2023. 3
- [7] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: an imperative style, high-performance deep learning library. In *NeurIPS*, 2019. 3
- [8] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017. 2, 4
- [9] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Bjorn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 1, 3
- [10] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding

conditional control to text-to-image diffusion models. In *ICCV*, 2023. 4

- [11] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017. 1, 3