ShapeShifter: 3D Variations Using Multiscale and Sparse Point-Voxel Diffusion

Nissim Maruani
Inria, Université Côte d'Azur
nissim.maruani@inria.fr

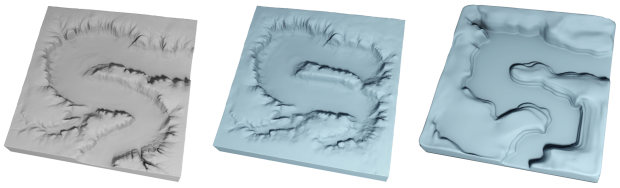
Wang Yifan
Adobe Research
yifwang@adobe.com

Matthew Fisher
Adobe Research
matfishe@adobe.com

Pierre Alliez
Inria, Université Côte d'Azur
pierre.alliez@inria.fr

Mathieu Desbrun
Inria/X, IP Paris
mathieu.desbrun@inria.fr

This supplementary material provides additional details, results, and comparisons.



(a) Input Geometry (b) Ours (c) Rodin

Figure 1. **ShapeShifter.** Given a 3D exemplar (left), we train a hierarchical diffusion model to create novel variations that preserve the geometric details and styles of the exemplar (center), whereas a large generative model such as Rodin [6] tends to lose the geometric details present in the input (right).

1. Additional results and renderings

We provide more results in Fig. 1, 4, and 5 to better illustrate the outputs of ShapeShifter on a variety of reference models. Note that we also show that ShapeShifter can generate purely geometric variants from untextured meshes, see last result in Fig. 5.

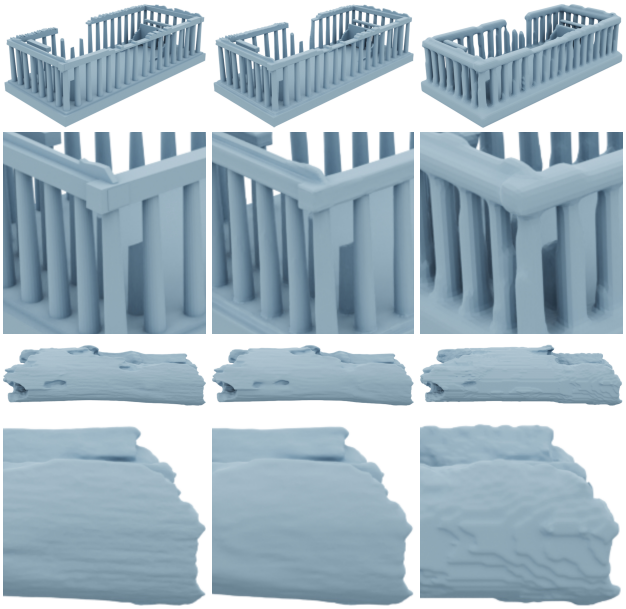
1.1. Comparison to SSG

We provide additional comparisons with SSG [4], which is a 3D generalization of SinGAN [3] trained on multiscale triplane occupancy fields. Tab. 1 shows quantitative evaluation on models for which SSG provides publicly available outputs, demonstrating the higher quality of ShapeShifter. Furthermore, we demonstrate in Fig. 2 that the typical results of this GAN-based method exhibit exaggerated smoothness like in all existing techniques, and often suffer from voxelized artifacts as well.

1.2. Data-intensive vs. exemplar-based generation

In this section, we discuss the value of exemplar-based 3D generation in light of the recent advancements in 3D generation models trained on millions of examples. The latter can be used to create highly diverse 3D assets and provide intuitive user controls through simple text and images. However, such models require immense computational resources for training and inference. Yet, as shown in Fig. 1, the state-of-the-art generator Rodin [6] (1.5B parameters) fails to create convincing geometric details comparable to those generated by our model.

Furthermore, the control provided by such models is limited, as the generation can only adhere to extremely coarse guidance. For example, in Fig. 1, we use the exemplar mesh as part of the inputs to Rodin for a conditioned generation.



(a) Input Geometry (b) ShapeShifter (c) SSG

Figure 2. **Visual inspection of SSG [4] results.** While SSG can generate 3D outputs with very short inference time, results are typically blobby or overly smooth, with spurious artifacts often visible due to its voxel-based generation process. In contrast, our method generates better sharp edges and subtle details.

Metric	Method	acropolis	house	small-town	wood
G-Qual. ↓	SSG	2.81	0.91	1.71	0.07
	ShapeShifter	**0.01**	**0.01**	**1.00**	**0.02**
G-Div. ↑	SSG	**0.081**	**0.01**	0.19	**0.11**
	ShapeShifter	0.04	**0.01**	**0.60**	0.08

Table 1. **Evaluating geometric quality and diversity using SSFID and pairwise IoU scores.** As we discussed in the main paper and in Sec. 2, both metrics have their blindspots: SSFID tends to overlook geometric details, while pairwise IoU systematically rewards artifacts.

However, the output (right) completely loses the styles and details present in the exemplar mesh.

2. Additional comments on metrics

While we use the two commonly-used metrics (geometric quality and diversity through SSFID and pairwise IoU scores) to evaluate our results and compare them to prior art, a few comments are in order.

First, the validity of these two scores is debatable. While geometric quality is arguably fair but cannot really gauge the diversity of the results, the measure of diversity itself is quite delicate to analyze. In a sense, the diversity score rewards noise, not just real diversity. For instance, ten grids of random binary values would get a diversity of 0.66, while ten grids of axis-aligned planes that are not overlapping would have a score of 1.0 — so a diversity score mixes different properties. This partial inadequacy of the score is the reason why we state in the main paper that geometric quality and geometric diversity should really be considered together to infer the success of an approach. Moreover, we also point out that the diversity scores should be clearly smaller for very structured models (like the acropolis model) than for free-form or organic shapes; our results have scores in line with this expected behavior, which seems more meaningful than systematic high scores which would point to noise artifacts instead of good results.

Second, we wish to point out that our scores of Sin3DM [5] are *different* from the ones they publish. The reason is that Sin3DM applies a pre-processing step to make the input meshes watertight. *This initial step systematically inflates small details and thin surfaces* such as the roof of the house or the entablature of the acropolis, which negates many of the advantages of one-shot generative modeling: it degrades (at times severely) the input, losing the very reason why creating variants of a carefully-designed input model is highly sought after, i.e., the high-quality geometry of the exemplar. So we compared their results to the unprocessed input models, and did not re-train their neural network because we assumed that they made their best efforts to fit ground-truth shapes. So one should be aware that the low geometric quality scores we provide reflect *both* the degradations of the pre-processing step and of their SDF-based generative approach — again, to account for the real use of these generative approaches.

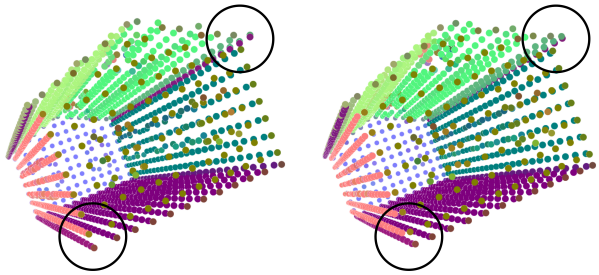
Method	Level 0	Level 1	Level 2	Level 3	Level 4
ShapeShifter	0.49	0.17	0.18	0.26	0.78
Sin3DM	-	5.18	-	-	-

Table 2. **Inference timing for generating a single variation.** We report the inference time at each level for generating a single variation and compare it with Sin3DM, which has a grid resolution equivalent to our second level (level 1). Note that in the main paper, we reported the inference time for 10 variations instead. DDIM sampling is used for both methods.

3. Inference timings

In the main paper, the inference times for ShapeShifter and Sin3DM are reported for the generation of 10 variants. Here, we provide the inference timing for generating a single variant (i.e., using a batch size of 1 instead of 10) as shown in Tab. 2.

Our method generates a single variant in less than 2 seconds: approximately 0.5 seconds for the coarsest level, followed by less than 1.5 seconds in total for the four finer levels. In comparison, Sin3DM requires around 5 seconds, while Sin3DGen takes over 3 minutes, excluding the time needed for optimizing the input plenoxels and converting them to a mesh. Notably, our method produces the coarsest level in under half a second, which can be directly splatted using [2] (see the video for live demonstrations). In contrast, Sin3DM[5] takes 5.18 seconds to process an equivalent grid size (32^3).



(a) QEM averaging (b) standard averaging

Figure 3. **QEM-averaging ablation.** While QEM-averaging (proposed in [1]) keeps sharp features (like corners or spikes) in place helping our generative approach maintain these local details, a usual averaging would move the "corner" points inwards, increasing the probability of smoothing features out in generated variants.

4. QEM averaging

Finally, we demonstrate why our use of QEM averaging during our fine-to-coarse analysis of the input mod-

els helps preserve sharp features of the ground truth. As Fig. 3 demonstrates, standard scale-by-scale averaging of the points and normals from the finest sparse voxel grid all the way to the coarsest grid leads to drifts of the salient features: for instance, the bottom left corner of the house has migrated inwards, which may create rounding of the corner. Instead, applying the QEM averaging defined in the PoNQ method [1] places the coarsest point on the corner, and of the intermediate points to remain right there as well — resulting in outputs which will better preserve this geometric feature.

References

[1] Nissim Maruani, Maks Ovsjanikov, Pierre Alliez, and Mathieu Desbrun. PoNQ: A Neural QEM-Based Mesh Representation. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3647–3657, Seattle, WA, USA, 2024. IEEE. 2, 3

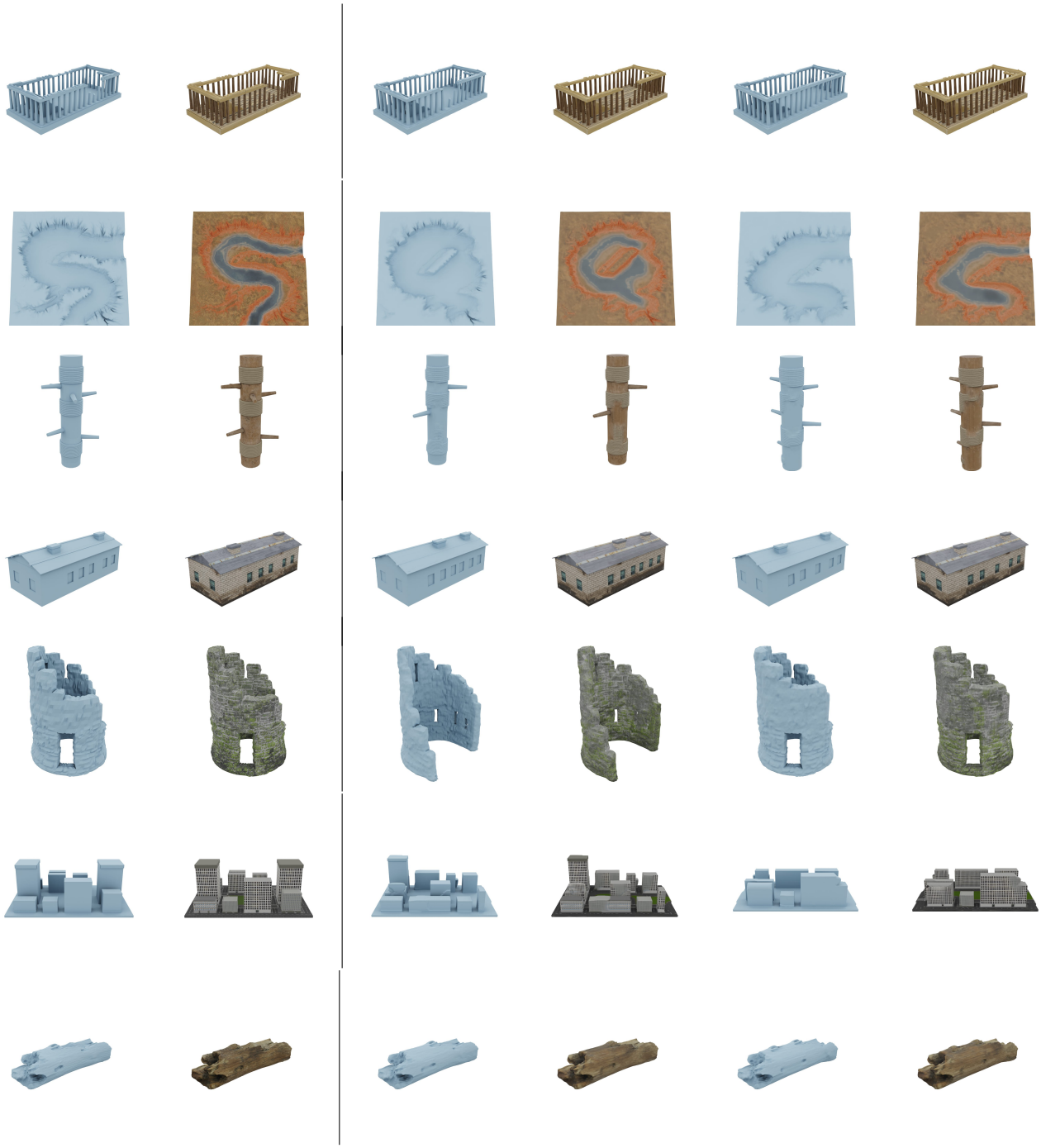
[2] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv:2007.08501*, 2020. 2

[3] Tamar Rott Shaham, Tali Dekel, and Tomer Michaeli. Singan: Learning a generative model from a single natural image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4570–4580, 2019. 1

[4] Rundi Wu and Changxi Zheng. Learning to generate 3d shapes from a single example. *ACM Transactions on Graphics (TOG)*, 41(6):1–19, 2022. 1

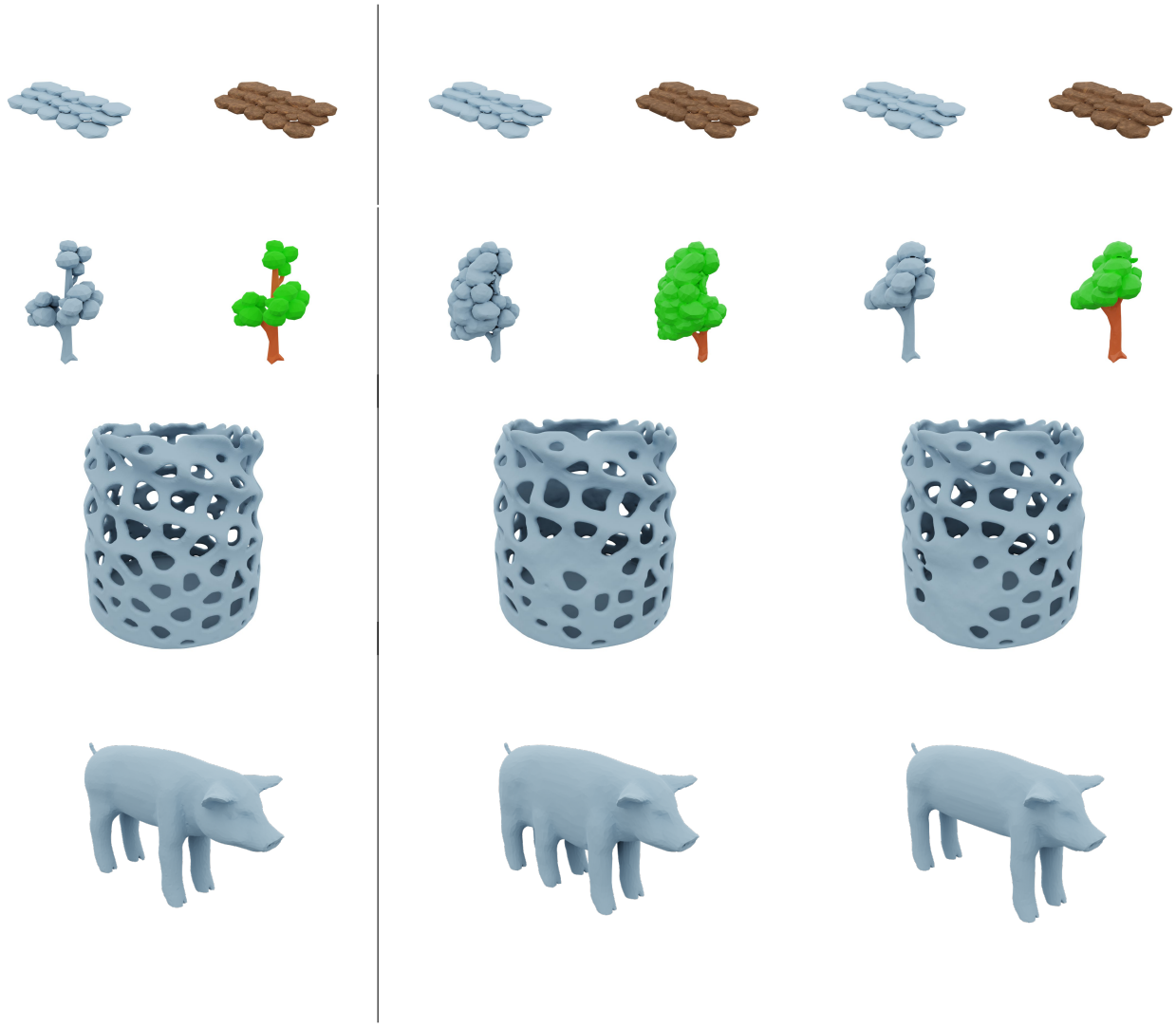
[5] Rundi Wu, Ruoshi Liu, Carl Vondrick, and Changxi Zheng. Sin3DM: Learning a diffusion model from a single 3d textured shape. In *International Conference on Learning Representations*, 2024. 2

[6] Longwen Zhang, Ziyu Wang, Qixuan Zhang, Qiwei Qiu, Anqi Pang, Haoran Jiang, Wei Yang, Lan Xu, and Jingyi Yu. Clay: A controllable large-scale generative model for creating high-quality 3d assets. *ACM Transactions on Graphics (TOG)*, 43 (4):1–20, 2024. 1



(a) Input Shape (b) Generated Shape (ours) (c) Generated Shape (ours)

Figure 4. **Samples of our results I.** This figure shows a variety of input models and some of the generated variants (both shown without and with texture to facilitate visual inspection) ShapeShifter outputs.



(a) Input Shape (b) Generated Shape (ours) (c) Generated Shape (ours)

Figure 5. **Samples of our results II.** This figure shows input models that were not used in the main paper, and some of the generated variants (both shown without and with texture to facilitate visual inspection) ShapeShifter outputs. Note that the last two examples (vase and pig) are an ablation test where we do not use colors among the per-voxel features in our approach.