3D Mesh Editing using Masked LRMs
Supplementary Material

Will Gao[1,2] Dilin Wang[2] Yuchen Fan[2] Aljaz Bozic[2] Tuur Stuyck[2]
Zhengqin Li[2] Zhao Dong[2] Rakesh Ranjan[2] Nikolaos Sarafianos[2]
[1]University of Chicago, [2]Meta Reality Labs
MaskedLRM Website

Introduction

We refer the interested reader to the supplementary video where we provide a plethora of qualitative results of our method. In the following sections we: i) conduct an ablation study that showcases the impact of our masking strategy, ii) showcase qualitative results of 2 recent methods (Nerfiller and Tailor3D) and explain some of their shortcomings, iii) provide implementation details of our method and iv) provide several figures with qualitative results.

Masking Ablation

In order to justify our masking strategy, we train our model masking patches uniformly randomly instead of using 3D occlusions. Figure 2 compares meshes extracted from a model trained using our 3D masking versus masking 25% of patches uniformly at random. We observe that uniformly random patch masking can still generate "roughly correct" shapes, especially adding a moustache to the face in the last row. This is because we add camera pose embeddings after masking, so the model can differentiate masked and non-masked tokens, regardless of their distribution within the image. Furthermore, the reconstruction outside of the masked region is still accurate. However, there still exists a train-test gap between random patches and contiguous patches created by selecting an editing region, which causes significant artifacts in the other three examples. In the first and second rows, we observe a blurring artifact, where the model cannot generate sharp features in the horns on the bird and between the slats of the chair. In the third row, using random patches causes the shape of the turtle shell to be malformed. In comparison, using our masking method produces accurate and sharp geometry in all examples.

Comparison to Tailor3D

Tailor3D [4] is a recent work in image-to-3D generation. Similar to InstantMesh [7], Tailor3D relies on a multiview diffusion model, namely Stable Zero-123 [3], to generate inputs that are then lifted into 3D. Tailor3D differs in that it only requires frontal and back views, using a novel transformer design to generate 3D assets from these sparse views. However, Tailor3D cannot replicate our method's mesh editing results due to two sources of error. First, as with other models that rely on multi-view synthesis, inaccuracies in

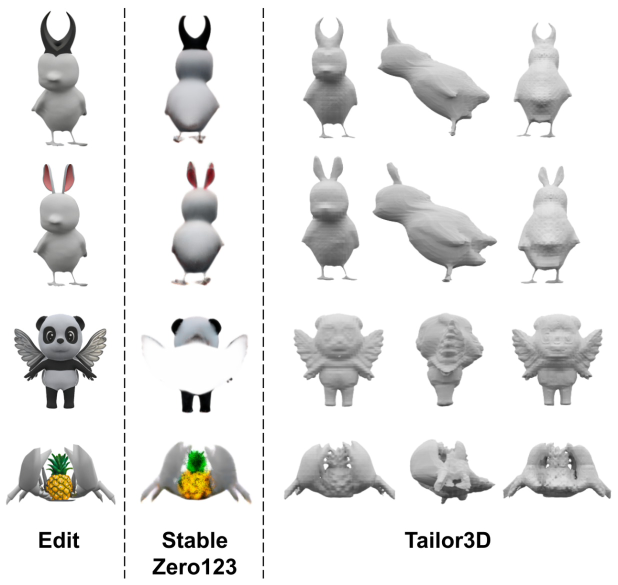

Figure 1. **Tailor3D Meshes**: Tailor3D results with the some of the conditions we used for our method throughout the paper. The left column shows the source image. The middle column shows a back view generated by Stable Zero-123 [3]. The right section shows the Tailor3D geometry rendered from 3 viewpoints. In the first two rows, Tailor3D suffers from ambiguity since it only sees the front and the back views and reconstructs an incorrectly elongated body. In the third row and fourth rows, Stable Zero-123 fails to generate a high-quality back view, failing completely for the wings on the panda. We observe the Janus effect in the generated panda and a lack of sharp features in the generated crab, especially viewed from the side.

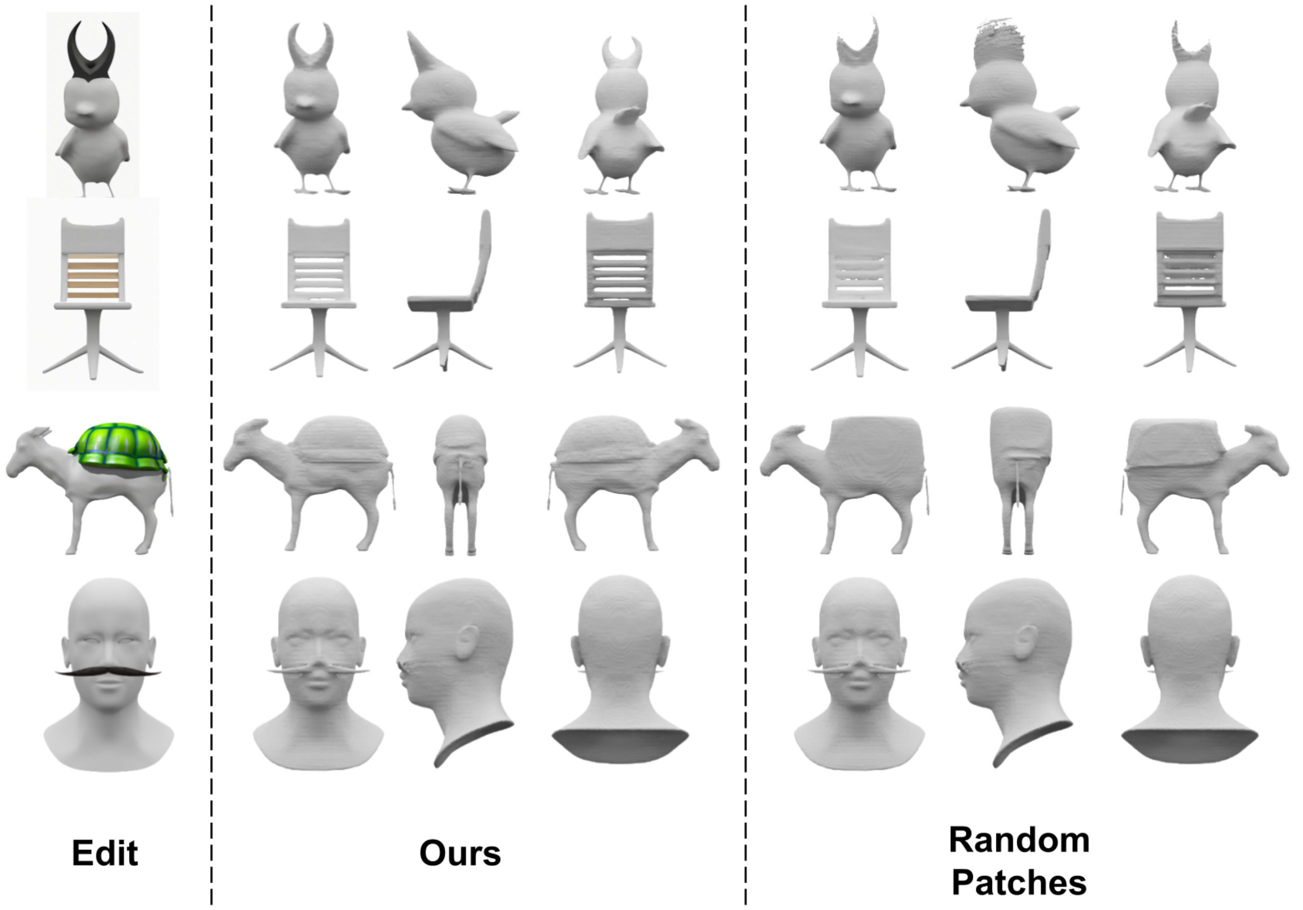

Figure 2. **Impact of Random Masking**: We test our choice of masking strategy by comparing it to masking 25% of patches uniformly at random. The left column shows the conditional image, the middle section shows our results, and the right section shows the results using uniformly random masking. While the model is still capable of generating correct geometry, there is a train-test gap in the masked patches since we define a contiguous 3D region to mask during inference. Thus, the model produces artifacts such as lack of sharp features (in the bird horns and chair slats) in rows 1 and 2, and overall incorrect shape in row 3 (square turtle shell).

generating the back view propagate into the 3D model. Second, despite its unique architecture, Tailor3D still suffers from ambiguity artifacts due to the sparse input. Figure 1 demonstrates some of these artifacts. In the first two rows, we observe that Tailor3D fails to recover the geometry of the body of the bird, due to the lack of information in the front and back views, creating an incorrectly elongated shape. In the third row, we see that Stable Zero-123 completely fails to generate the back view of the wings, leading to a mirroring artifact in the final 3D shape. The fourth row suffers similar issues as the previous three, with the generated view being not only low-quality but also a mirror of the front view instead of a true back view.

Comparison to Nerfiller

Nerfiller [5] is a recent work that uses pre-trained image generation models for guidance in order to inpaint masked regions in NeRFs. Nerfiller begins by training a NeRF on unoccluded pixels, and then slowly updates the training set over time via generative inpainting. They adapt their method to image-conditional completion by simply prompting the generative process using a single inpainted image as reference. This is exactly analogous to the input image edits in our method. Figure 3 shows some of the images Nerfiller produces using our image edits as reference. We observe that, although the inpainted images are generally semantically correct, details are inconsistent such as the color of the hats in the first row. Some frames are even missing the hat or rabbit ears entirely. While training a NeRF may tolerate some noise within the training set, this causes blurriness artifacts in the resultant 3D asset and is not suitable for explicit geometry extraction. Furthermore, since Nerfiller repeats this process of training a NeRF and then updating the dataset several times, it is significantly more expensive to run than our method, taking over an hour on an A100 GPU.

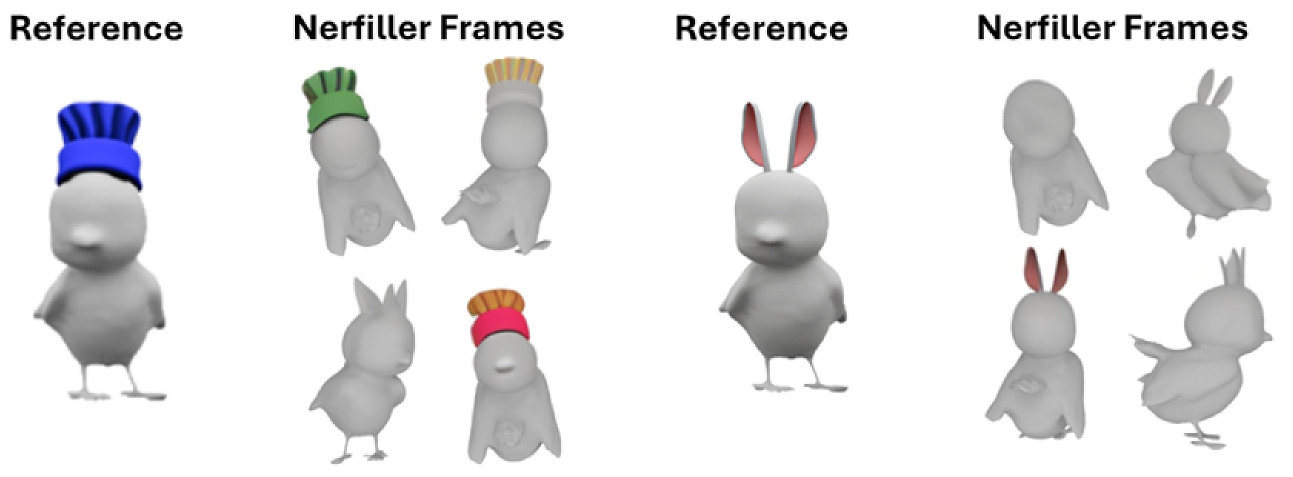

Figure 3. **Nerfiller Images**: Nerfiller results with a couple of bird edits we used for our method. We use their adapted method for reference-image based inpainting. The left column shows the reference image and the right column shows a collection of Nerfiller generated images. We observe that their inpainting method based on pre-trained diffusion models creates noisy output images. The semantics may be correct, but the details can be incorrect *e.g.* incorrect hat colors or completely missing *e.g.* missing hat and ears in a couple of the examples. NeRF training may be tolerant to somewhat noisy input data, but these viewpoints are not suitable for precise geometry reconstruction.

Mesh Editing Comparisons						
	ViT-L-14			ViT-BigG-14		
	Ours	Instant3Dit	MagicClay	Ours	Instant3Dit	MagicClay
CLIP Cos.Sim. ↑	0.323	0.303	0.285	0.337	0.309	0.286

Table 1. **Mesh Editing Comparisons**: We provide CLIP cosine similarity metrics of our proposed MaskedLRM approach against two recent 3d editing techniques.

Mesh Editing Metrics

Large scale quantitative editing comparisons are difficult as there is no standard benchmark. In the Table above we compare against MagicClay [2] and Instant3Dit [1], generating edits using a text prompt. We then use the same text input for generating our conditional edited views and use CLIP similarity averaged across multiple views to measure the faithfulness of each method to the prompt.

Additional Implementation Details

Our model implementation details are based mostly on [6]. Our model tokenizes 16×16 sized patches. The token embedding size and transformer width are 1024. The transformer depth is 24 layers. Each attention and cross attention module use multi-head attention with 16 heads. Our model uses LayerNorm and GeLU activations with a Pre-LN architecture.

We trained our models using 64 H100 GPUs with 80GB of RAM each. We use an AdamW optimizer with $(\beta_1, \beta_2) = (0.9, 0.95)$ and a weight decay of 0.01. During stage 1 of training, we train for 30 epochs. For each batch consisting of 12 shapes, we randomly sample the number of input views for the batch uniformly at random between 6 and 8, not including the 1 view for the conditional view. We use another 4 views for supervision. Over the first 1500 iterations, we linearly warm up to a peak learning rate of $4e-4$ and then use cosine learning rate decay. During stage 2, to account for increased rendering costs, we reduce the batch size to 8 shapes. We train for 20 epochs, with a peak learning rate of $5e-6$.

Additional Qualitative Examples

We present some additional qualitative examples of our model in Figures 4 and 5. We show the network inputs *i.e.* the masked views and the edited image on the left, and the network outputs *i.e.* the resulting geometry with RGB volumetrc renders inset on the right.

References

[1] Amir Barda, Matheus Gadelha, Vladimir G. Kim, Noam Aigerman, Amit H. Bermano, and Thibault Groueix. Instant3dit: Multiview inpainting for fast editing of 3d objects, 2024. 3

[2] Amir Barda, Vladimir G. Kim, Noam Aigerman, Amit Bermano, and Thibault Groueix. Magicclay: Sculpting meshes with generative neural fields. In *ACM Transactions on Graphics (SIGGRAPH Asia)*, 2024. 3

[3] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3d object. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9298–9309, 2023. 1

[4] Zhangyang Qi, Yunhan Yang, Mengchen Zhang, Long Xing,

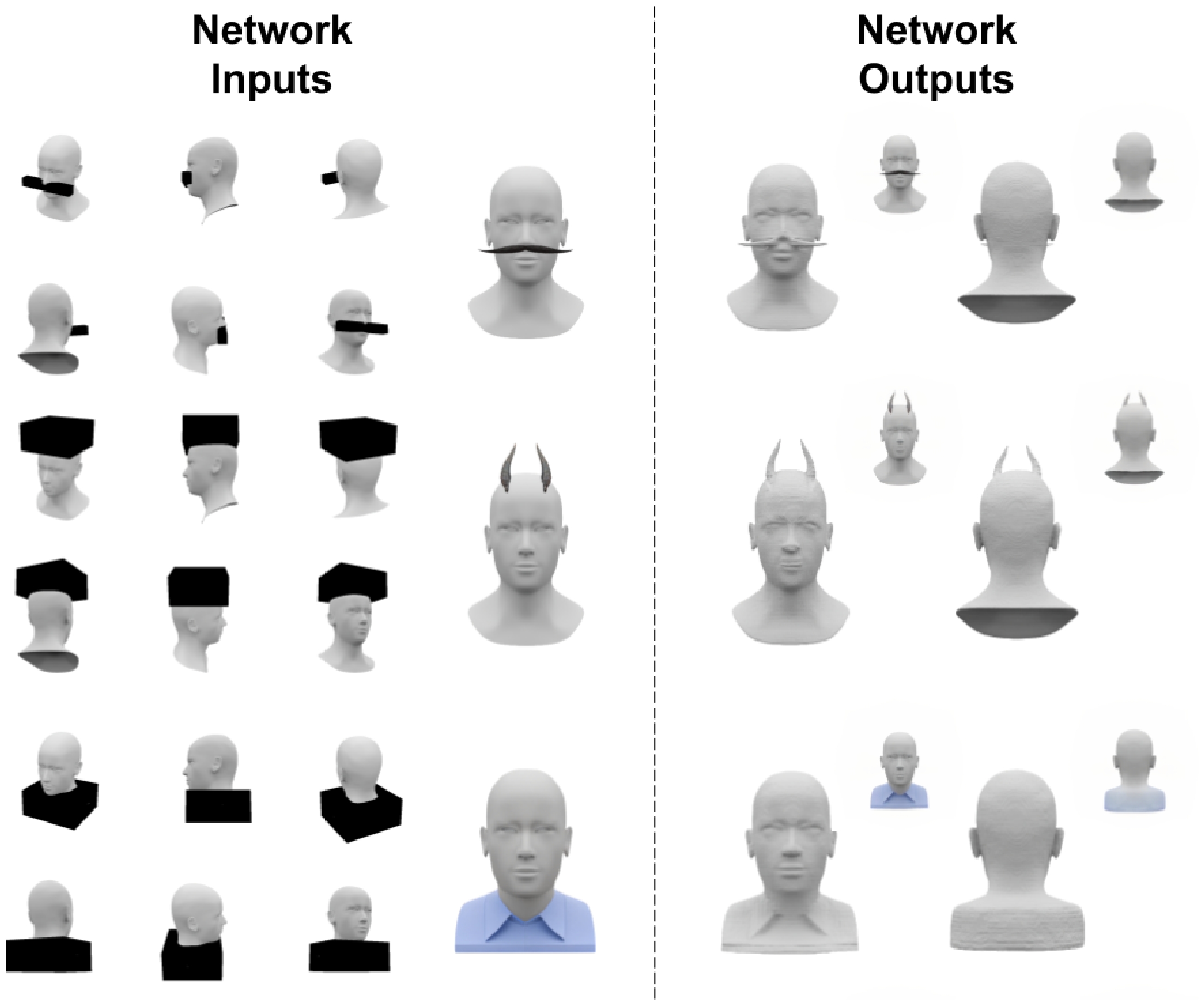

Figure 4. **Additional Qualitative Examples**: Additional qualitative examples editing a person's head. The left section shows the masked views and the edited conditional image. The right section shows the mesh extracted from the network output with the volumetric renders of the SDF inset.

Xiaoyang Wu, Tong Wu, Dahua Lin, Xihui Liu, Jiaqi Wang, and Hengshuang Zhao. Tailor3d: Customized 3d assets editing and generation with dual-side images, 2024. 1

[5] Ethan Weber, Aleksander Holynski, Varun Jampani, Saurabh Saxena, Noah Snavely, Abhishek Kar, and Angjoo Kanazawa. Nerfiller: Completing scenes via generative 3d inpainting. In *CVPR*, 2024. 2

[6] Xinyue Wei, Kai Zhang, Sai Bi, Hao Tan, Fujun Luan, Valentin Deschaintre, Kalyan Sunkavalli, Hao Su, and Zexiang Xu. Meshlrm: Large reconstruction model for high-quality mesh. *arXiv preprint arXiv:2404.12385*, 2024. 3

[7] Jiale Xu, Weihao Cheng, Yiming Gao, Xintao Wang, Shenghua Gao, and Ying Shan. Instantmesh: Efficient 3d mesh generation from a single image with sparse-view large reconstruction models. *arXiv preprint arXiv:2404.07191*, 2024. 1

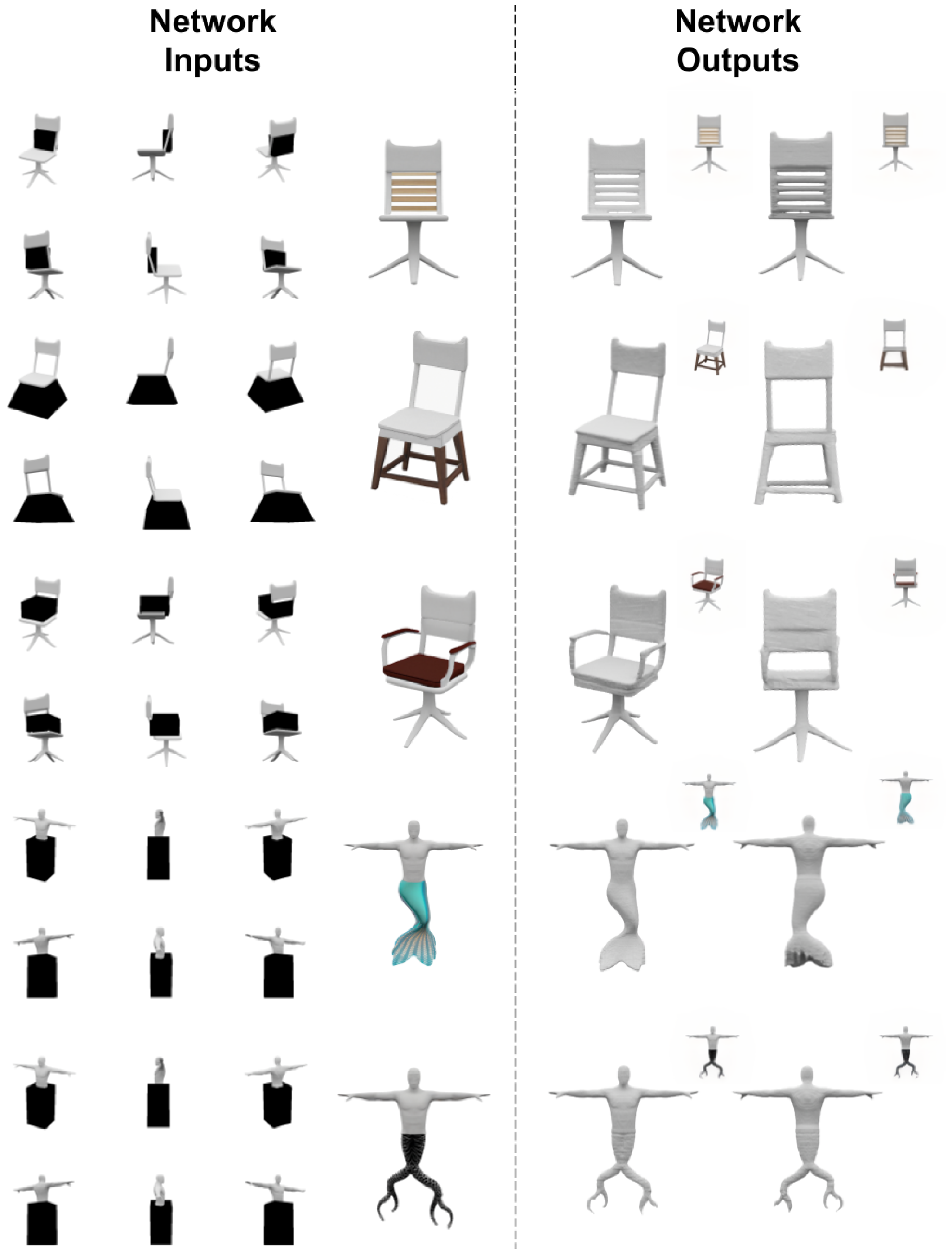

Figure 5. **Additional Qualitative Examples**: Additional qualitative examples editing a chair and a full human. The left section shows the masked views and the edited conditional image. The right section shows the mesh extracted from the network output with the volumetric renders of the SDF inset.