Instant3dit: Supplementary Materials

The supplementary material for our submission consists of this document, a video showing our GUI for generative 3D editing, and an HTML webpage containing a gallery of results. The files corresponding to the video and to the webpage are within the supplemental material package with the names `demo.mp4` and `index.html`, respectively. In this document, we provide additional visual illustrations to complement the results in the main Paper.

Summary

- In Figure 1, we compare Instant3dit with various baselines and ablated version, to complement the quantitative results presented in Table 1 of the main paper.
- In Figure 2 and Figure 3, we ablate the importance of each type of training mask, complementing the quantitative results in Table 2.
- Figure 4, Figure 5, and Figure 6 show three galleries of generative editing results on meshes.
- Figure 7 provides additional Gaussian Splat reconstructions complementing the results of other figures showcasing NeRFs and meshes reconstructions.
- Figure 8 illustrates that our multiview inpainting diffusion model generalizes to novel camera angles, complementing the graph in the main paper.
- In Figure 9, we show a screen capture of our user study.
- Finally, Figure 10 illustrates how Instant3dit can be used to perform multiple generative edits in a row to the same asset.

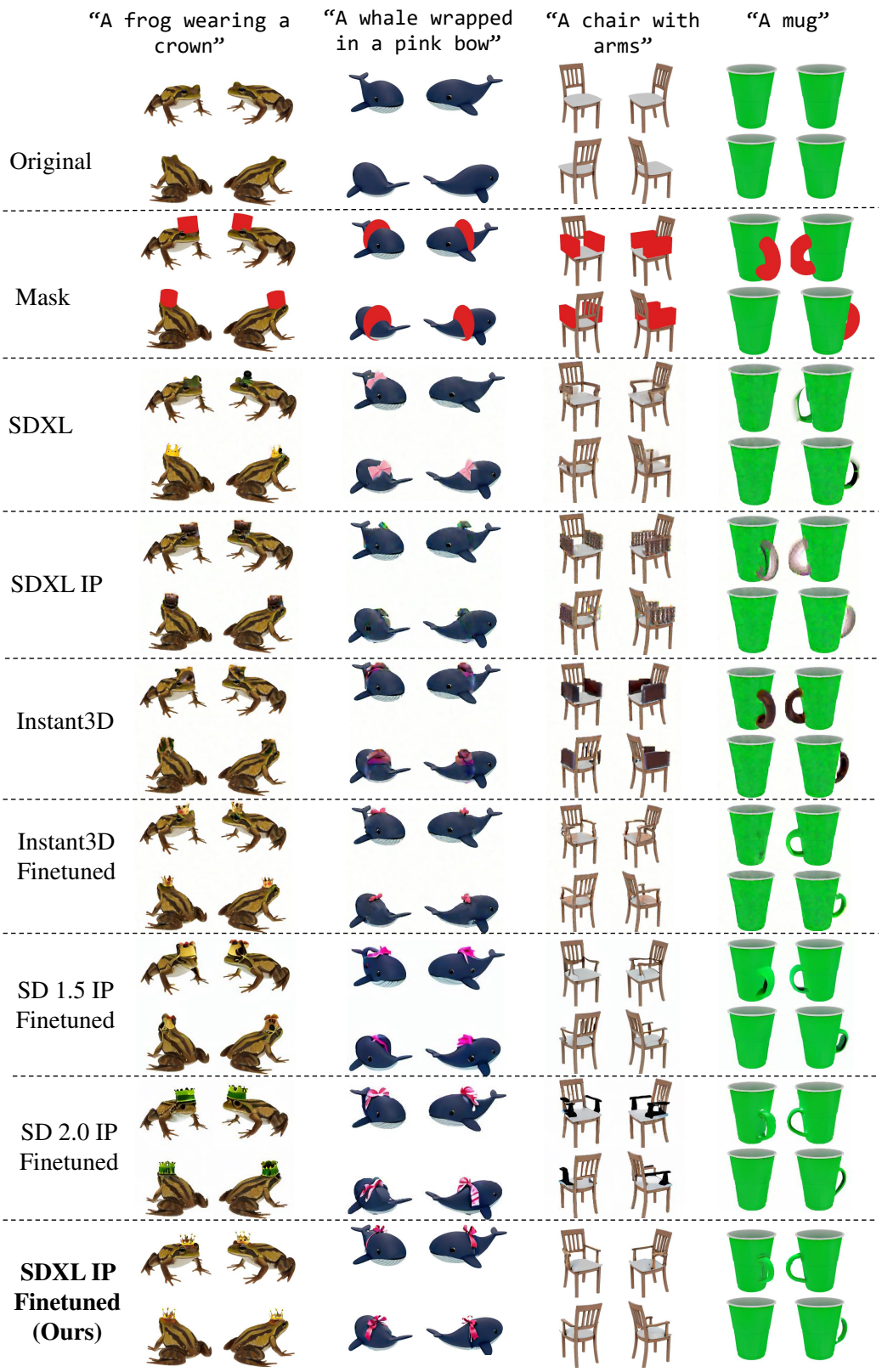

Figure 1. **Multiview text-to-image inpainting**. This figure illustrates the quantitative results in Table 1 of the main document and demonstrates the qualitative performance of several baselines and ablations of our method. Notice that our method has higher visual quality in the unpainted region - compare the inpainted bow on the whale (*middle column*), better multiview consistency - notice the missing or poor handle on the top left view of the mug (*rightmost column)* -, and better prompt adherence - in the crowned frog example (*left column*), only Instant3D Finetuned and SDXL IP Finetuned generate a recognizable crown.

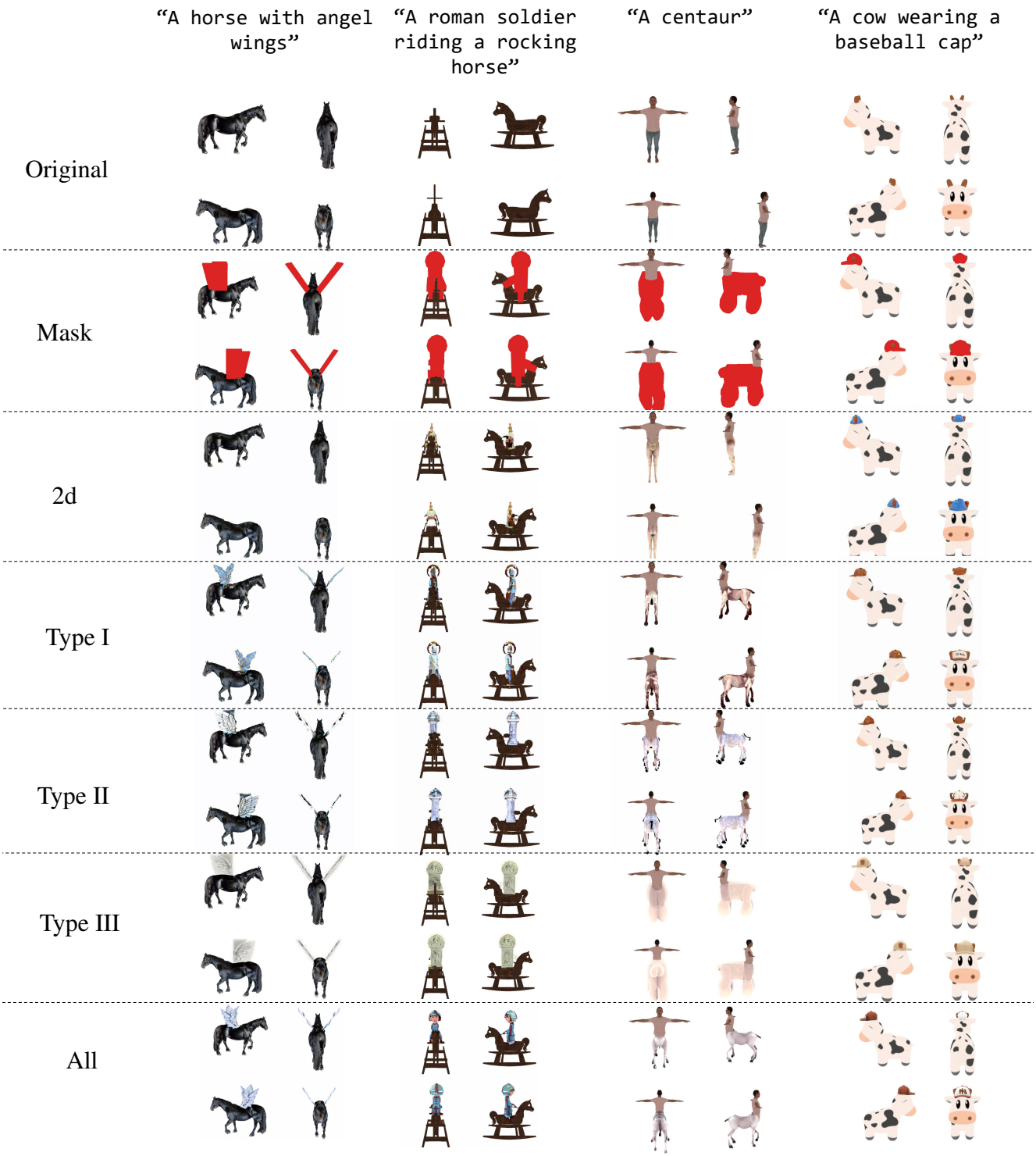

Figure 2. **Mask Ablation - Qualitative Examples.** This figure illustrates the quantitative results in Table 2 of the main document. It demonstrates the importance of each type of training mask on four samples from our benchmark of user-generated masks. As confirmed by the quantitative analysis, training with random 2D masks leads to ignoring the prompt most of the time or poor multiview consistency (*row #3*), Type II mask and Type III mask have low visual quality when trying to add a new part to the model (*row #4 and #5*), a common use case in our benchmark. Finally, the Type I mask produces reasonable results, but, as expected, the best visual quality and consistency are obtained when training on all types of masks (*row #2 and bottom row*).

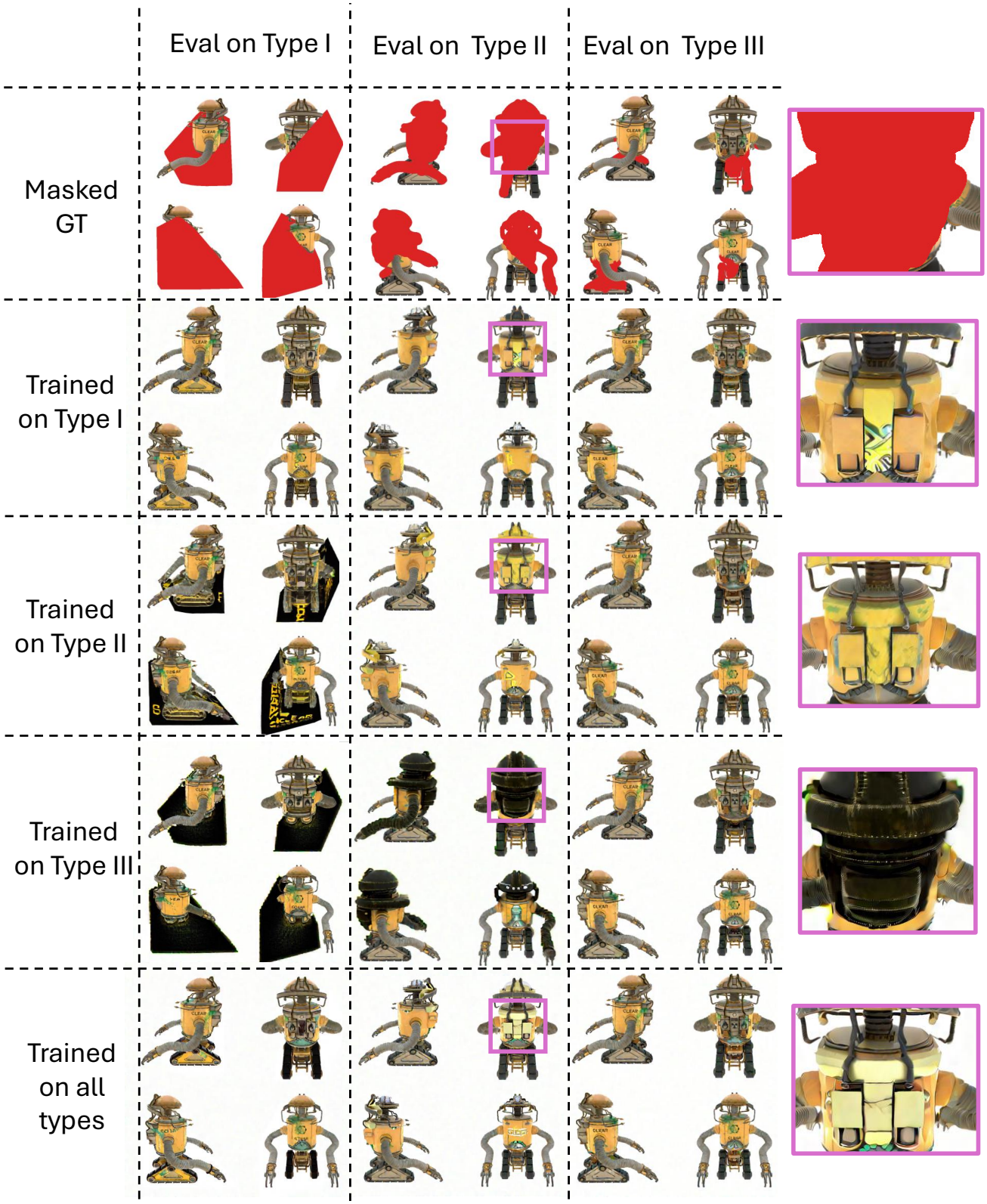

Figure 3. **Training Mask Ablation** Corresponding to Table 2 in the main paper. We fine-tune on one type of mask (in rows), and test on all three types (columns). The last column shows a close-up view for the evaluations on a type II mask, showing the improvement in texture when fine-tuning on all masks.

Figure 4. **Gallery of results 1/3.** From left to right, we show the original mesh (*column 1*), the projected mask into four views (*column 2*), input to Instant3dit to generate a multiview inpainted image (*column 3*), reconstructed with NeRF-LRM (*column 4*), and Mesh-LRM (*column 5*), and finally, we show the adaptive remeshing results, for which we use the output of MeshLRM in a lightweight optimization that only allows part of the mesh to be modified, while retaining all pre-existing attributes (triangulation, UVs etc.) (*column 6*).

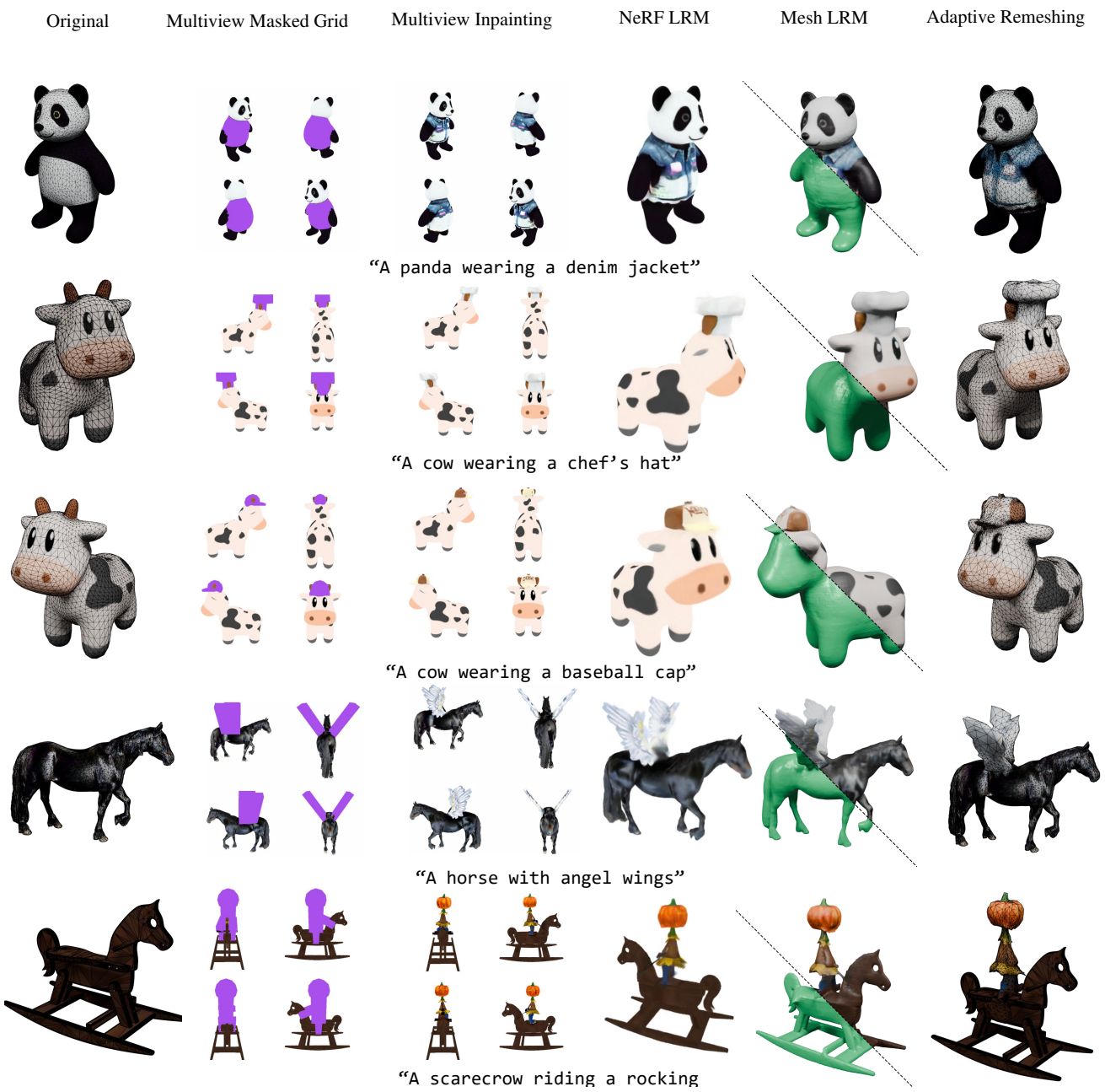

Figure 5. **Gallery of results 2/3.** From left to right, we show the original mesh (*column 1*), the projected mask into four views (*column 2*), input to Instant3dit to generate a multiview inpainted image (*column 3*), reconstructed with NeRF-LRM (*column 4*), and Mesh-LRM (*column 5*), and finally, we show the adaptive remeshing results, for which we use the output of MeshLRM in a lightweight optimization that only allows part of the mesh to be modified, while retaining all pre-existing attributes (triangulation, UVs etc.) (*column 6*).

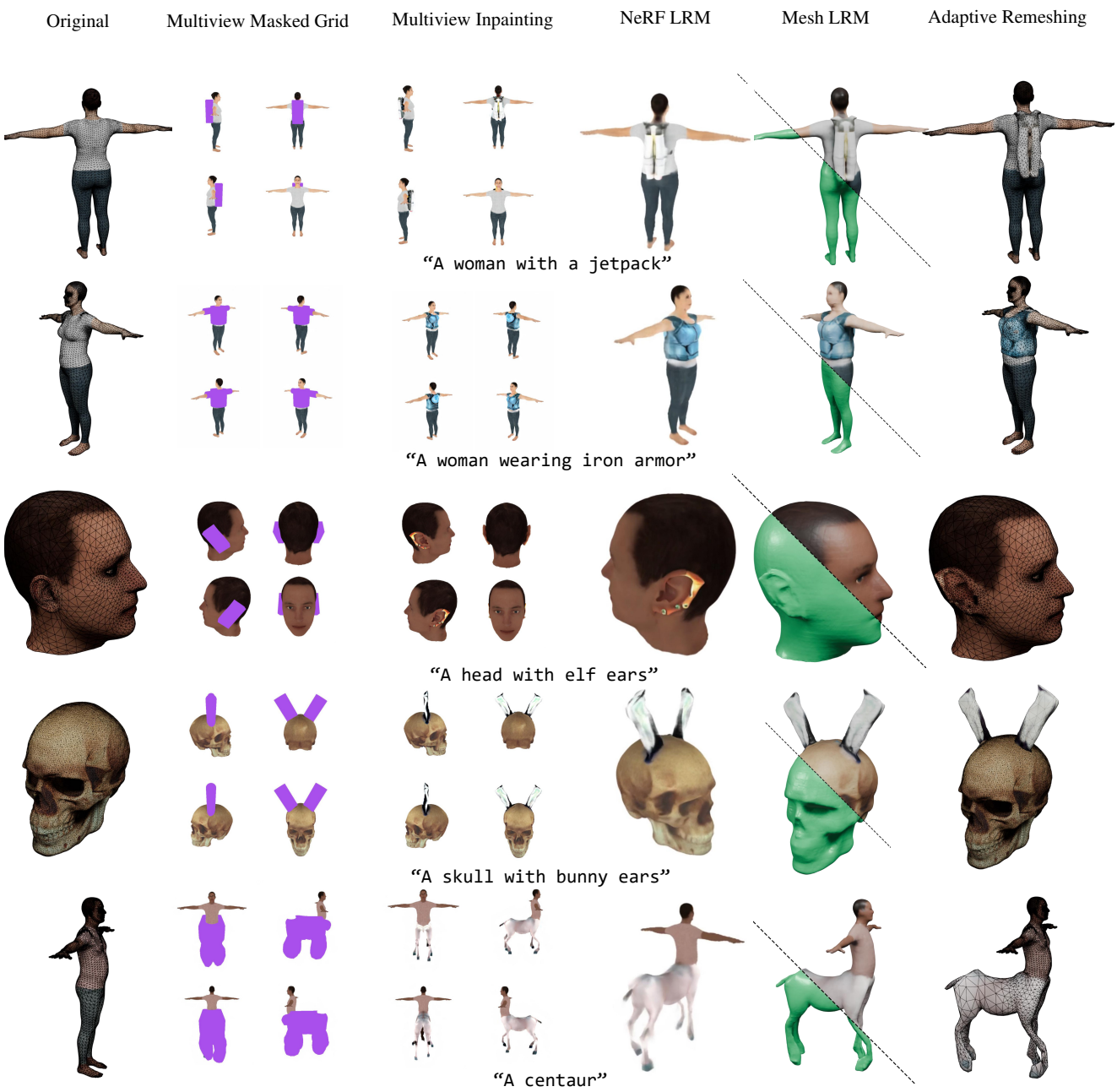

Figure 6. **Gallery of results 3/3.** From left to right, we show the original mesh (*column 1*), the projected mask into four views (*column 2*), input to Instant3dit to generate a multiview inpainted image (*column 3*), reconstructed with NeRF-LRM (*column 4*), and Mesh-LRM (*column 5*), and finally, we show the adaptive remeshing results, for which we use the output of MeshLRM in a lightweight optimization that only allows part of the mesh to be modified, while retaining all pre-existing attributes (triangulation, UVs etc.) (*column 6*).

Figure 7. **Additional Gaussian Splat Reconstruction**. To complete the results presented in the teaser of the main document, we present additional reconstruction with GS-LRM, on the right side of the dotted line.

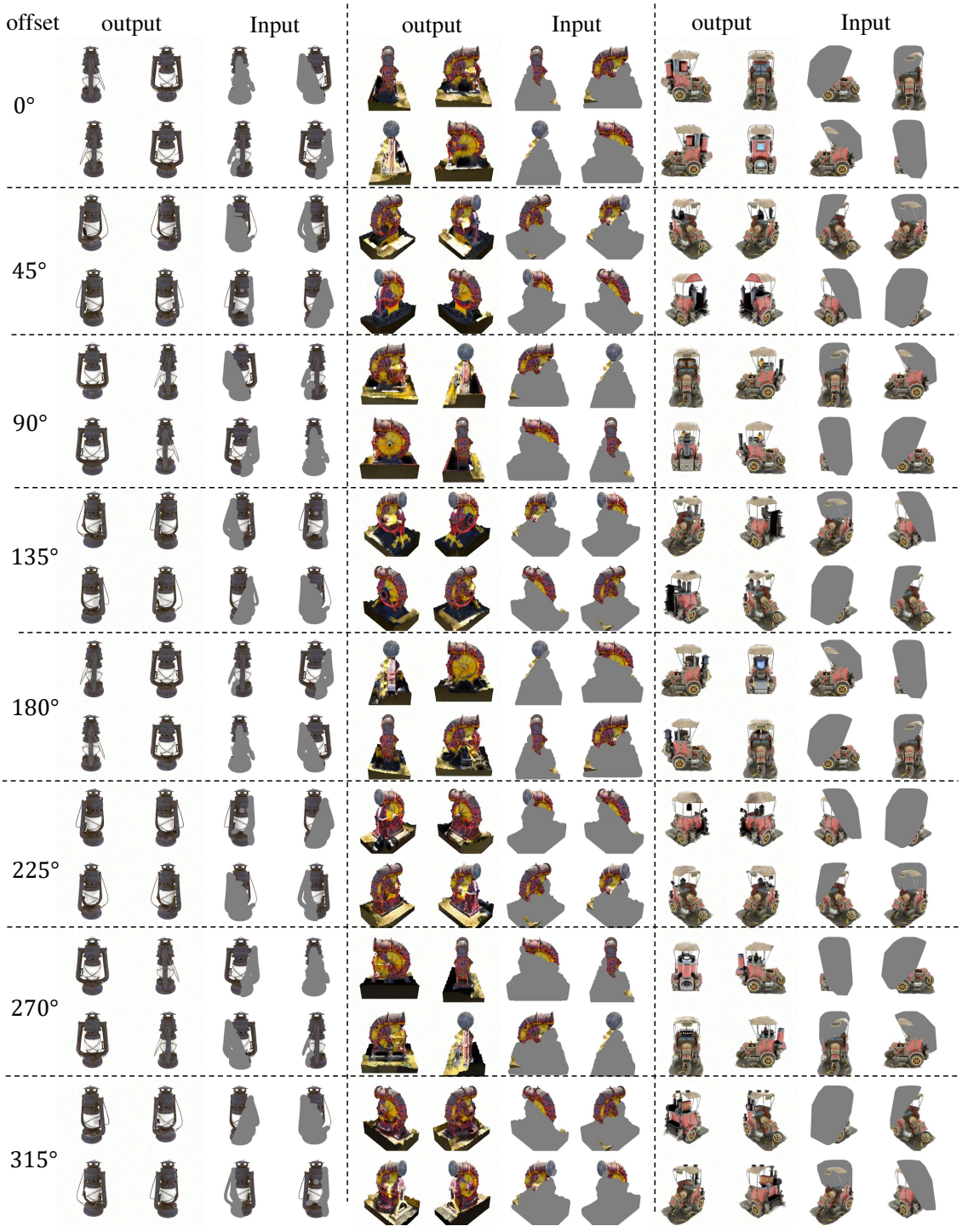

Figure 8. **Generalization to Novel Camera Angles.** We complement the quantitative graph presented in the Experiment section of the main document with qualitative results shown on 3 randomly selected objects from the validation set. We offset the azimuth of the cameras during multiview rendering from 0 to 315 degrees. On the right, we show the rendered views with corresponding masks. Notice that the mask is defined in 3d, so it is the same mask in each 'Input' column, and is simply rotated. On the left, the inpainted results exhibit consistent visual quality, prompt adherence, and multiview consistency for all azimuth offsets despite the model being trained on canonical views, corresponding to the zero offset. Note, that each input has a prompt that is not shown here, as it is identical for all offsets.

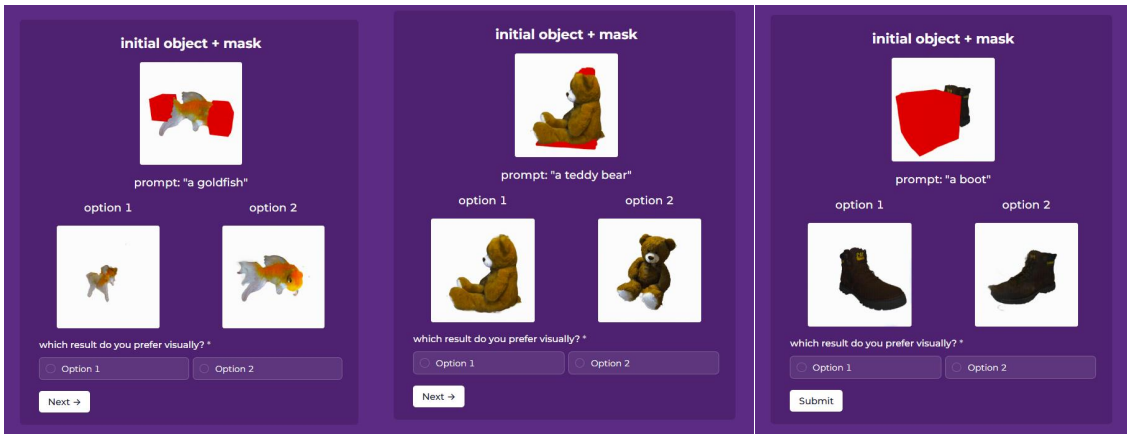

Figure 9. **User study.** We compare Instant3dit against NeRFiller on user-generated masks for scene completion tasks, which NeRFiller is designed for. Above, we show three samples from the 13 scenes used in the study.

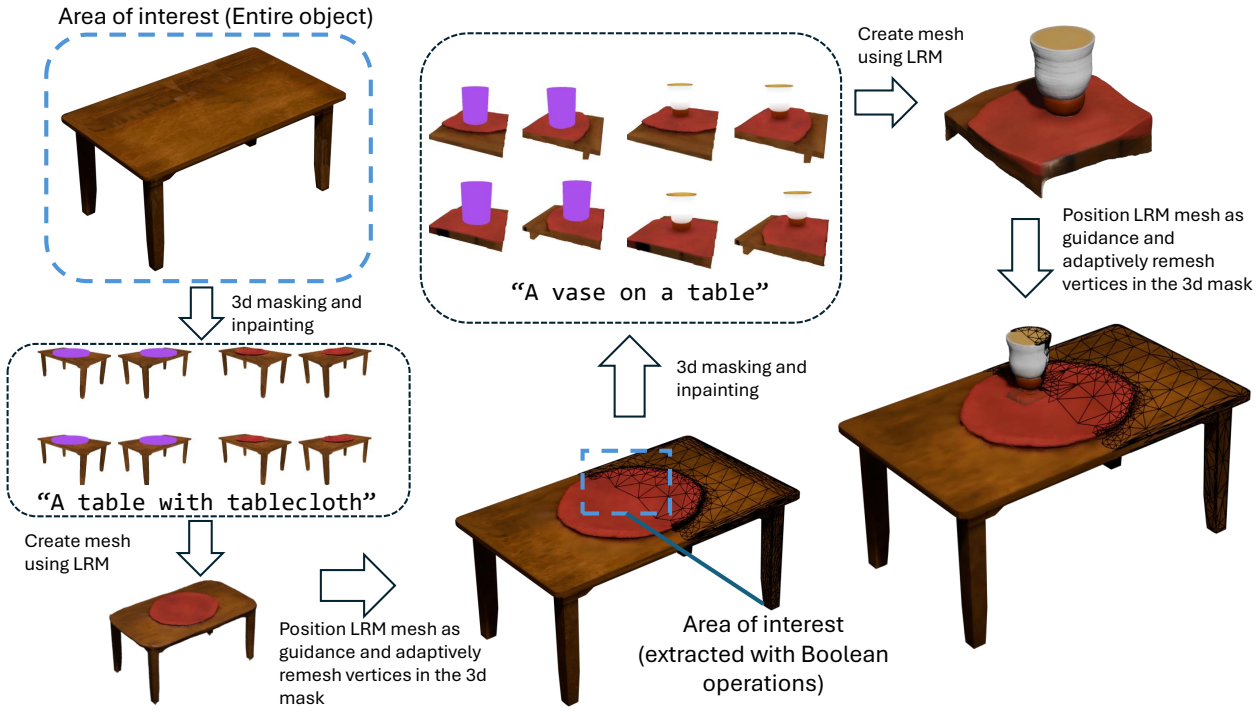

Figure 10. **Sequential mesh editing**. We illustrate how Instant3dit can be applied iteratively to perform multiple edits to a mesh. To get finer details out of our pipeline, we run it on a submesh, centered on the editable region. The submesh is normalized, then passed through Instant3dit and mesh-LRM, and unnormalize using the same mean and scale parameters, allowing for aligning the edited submesh with the original mesh. The repositioned submesh then guides local adaptive re-meshing. Each edit takes about 25 seconds.