

Axolotl3D: a Unified Framework for Faithful 3D Shape Completion

Supplementary Material

Anita Hu¹  and Maria Shugrina¹ 

NVIDIA

<https://research.nvidia.com/labs/sil/projects/axolotl3d>

Contents

1	Model Architecture	1
1.1	Gated Feed-Forward Network	1
2	Training Data Processing	2
2.1	Surface Point Sampling	2
2.2	Data Augmentation Functions	2
2.3	Training Data Filtering	4
3	Experiment Details	5
3.1	Occlusion Mask Generation	5
3.2	View Selection for Hunyuan3D-Omni	6
3.3	Image-to-3D with MapAnything	6
3.4	3D Capture & Simulation	7
4	Additional Results	8
4.1	Quantitative Results without ICP	8
4.2	Object Edit Comparisons	8
4.3	Qualitative Comparisons	9
5	Limitations & Future Work	9

1 Model Architecture

1.1 Gated Feed-Forward Network

We adopt the feed-forward block used in LLaMA-3 [6] and SAM 3D [1]. Given an input $x \in \mathbb{R}^d$, the feed-forward layer is defined as:

$$\text{FFN}(x) = W_2 (\text{SiLU}(W_1 x) \odot (W_3 x)), \quad (1)$$

where $W_1, W_3 \in \mathbb{R}^{d \times h}$, $W_2 \in \mathbb{R}^{h \times d_{\text{out}}}$. Here, h is the hidden dimension, d and d_{out} are the input and output dimensions, $\odot$ denotes element-wise multiplication, and $\text{SiLU}(z) = z \cdot \sigma(z)$.

2 Training Data Processing

2.1 Surface Point Sampling

When preprocessing meshes for training, we generate surface point cloud P_{surface}^k to use with our data augmentation pipeline to generate conditioning points. To mimic the points a real-world depth sensor would capture, internal components that are fully or mostly enclosed within larger parts of the mesh are removed to avoid sampling from hidden geometry. Points are then uniformly sampled from the remaining external surfaces, along with their normals. To identify internal components, the mesh is first split into connected components. Each component is voxelized and filled to represent its full volume, then aligned in a global voxel grid for comparison. For each component, we compare against all components larger than itself to determine if it should be removed. Let V_i and V_j denote the sets of voxels occupied by components i and j , respectively. For a component i being checked against a larger component j , we define the *overlap fraction* as

$$f_{i,j} = \frac{|V_i \cap V_j|}{|V_i|}. \quad (2)$$

Component i is removed if there exists a larger component j such that $f_{i,j} \geq 0.95$.

2.2 Data Augmentation Functions

Here we provide details on algorithms used during data augmentation. We assume a setting in which per-view depth maps are not provided, as is often the case in rendering pipelines for 3D generation models, such as TRELLIS [22].

Let the mesh surface points P_{surface} consist of points $\{p_i\}_{i=1}^S$ and corresponding normals $\{n_i\}_{i=1}^S$ where S is the number of points. Alg. 1 filters the points to produce a binary point visibility mask $V \in \{0, 1\}^S$ for a given camera view with intrinsics K and extrinsics E . We first perform back-face culling using the point normals and the camera backward vector. Next, for each point p_i , the function $\text{PROJECT}(p_i, E, K)$ returns the corresponding normalized image coordinates (u_i, v_i) and depth d_i . Due to the sparsity of points, a coarse z-buffer Z is constructed by projecting the points onto a low-resolution image grid and storing the minimum depth in each cell. This provides an approximation of point visibility across the image. To incorporate the occlusion mask M_{occ} (a binary mask where 1 indicates an occluded pixel), the z-buffer is upsampled to Z_{up} , matching the resolution of M_{occ} . Finally, points whose depth is within a threshold $\epsilon = 0.05$ of the z-buffer and not occluded in M_{occ} are marked visible. The algorithm is applied for all N conditioning views and the resulting masks $\{V_j\}_{j=1}^N$ are combined with a point-wise logical OR to obtain the final visibility mask to obtain the conditioning points P .

Alg. 2 is used during the edit scenario to generate visibility masks $\{M_i\}_{i=1}^N$ from 3D bounding boxes. For each bounding box $B \in \mathbb{R}^{8 \times 3}$, the corners are projected onto the image and the convex hull of the projected points is used to create the visibility mask. While this efficiently approximates occluded regions,

Algorithm 1 Select Visible Points in a Camera View

Require: 3D points $\mathbf{P}_{\text{surface}} = \{\mathbf{p}_i\}_{i=1}^S$, normals $\{\mathbf{n}_i\}_{i=1}^S$
Require: Camera extrinsics $\mathbf{E}$, intrinsics $\mathbf{K}$
Require: Optional occlusion mask $\mathbf{M}_{\text{occ}} \in \mathbb{R}^{H \times W}$
Output: Point visibility mask $\mathbf{V} \in \{0, 1\}^S$

- 1: $\mathbf{b} \leftarrow -\mathbf{E}_{3,1:3}$ ▷ Camera backward vector
- 2: **for** $i = 1$ to S **do** ▷ Back-face culling (normal filtering)
- 3: $f_i \leftarrow (\mathbf{n}_i^\top \mathbf{b} > 0)$
- 4: **end for**
- 5: **for** $i = 1$ to S **do** ▷ Project points to image coordinates and compute depth
- 6: $(u_i, v_i, d_i) \leftarrow \text{PROJECT}(\mathbf{p}_i, \mathbf{E}, \mathbf{K})$
- 7: **end for**
- 8: $h \leftarrow H/8, \quad w \leftarrow W/8$
- 9: Initialize z-buffer $\mathbf{Z} \in \mathbb{R}^{h \times w}$ with $+\infty$
- 10: **for** $i = 1$ to S **do** ▷ Coarse z-buffer construction
- 11: $x_i, y_i \leftarrow \text{clip}(\lfloor u_i \cdot (w - 1) \rfloor, 0, w - 1), \text{clip}(\lfloor v_i \cdot (h - 1) \rfloor, 0, h - 1)$
- 12: $\mathbf{Z}[y_i, x_i] \leftarrow \min(\mathbf{Z}[y_i, x_i], d_i)$
- 13: **end for**
- 14: $\mathbf{Z}_{\text{up}} \leftarrow \text{BILINEARUPSAMPLE}(\mathbf{Z}, H, W)$ ▷ Upsample z-buffer to full resolution
- 15: **if** $\mathbf{M}_{\text{occ}}$ is not provided **then** ▷ Set occlusion mask to all visible if not provided
- 16: $\mathbf{M}_{\text{occ}} \leftarrow \mathbf{0}^{H \times W}$
- 17: **end if**
- 18: Initialize $\mathbf{V} \leftarrow \mathbf{0}^S$ ▷ All pixels initially visible
- 19: **for** $i = 1$ to S **do** ▷ Set point visible if within depth threshold and not occluded
- 20: $x_i, y_i \leftarrow \text{clip}(\lfloor u_i \cdot (W - 1) \rfloor, 0, W - 1), \text{clip}(\lfloor v_i \cdot (H - 1) \rfloor, 0, H - 1)$
- 21: **if** $|d_i - \mathbf{Z}_{\text{up}}[y_i, x_i]| < \epsilon \wedge \mathbf{M}_{\text{occ}}[y_i, x_i] < 1$ **then**
- 22: $V[i] \leftarrow 1$
- 23: **end if**
- 24: **end for**
- 25: **return** $\mathbf{V}$



Fig. 1: Training Data Filtering: Example of objects that are removed.

it may over-occlude areas where the bounding box corners are behind or within the object.

Algorithm 2 Create Visibility Mask from 3D Bounding Box

Require: Bounding box points $\mathbf{B} \in \mathbb{R}^{8 \times 3}$
Require: Camera extrinsics $\mathbf{E}$, intrinsics $\mathbf{K}$
Output: Binary visibility mask $\mathbf{M} \in \{0, 1\}^{H \times W}$

- 1: Initialize $\mathbf{M} \leftarrow \mathbf{1}^{H \times W}$ ▷ All pixels initially visible
- 2: **for** $j = 1$ to 8 **do** ▷ Project bounding box points to image coordinates
- 3: $(u_j, v_j) \leftarrow \text{PROJECT}(\mathbf{B}_j, \mathbf{E}, \mathbf{K})$
- 4: $x_j, y_j \leftarrow \text{clip}(\lfloor u_j \cdot (W - 1) \rfloor, 0, W - 1), \text{clip}(\lfloor v_j \cdot (H - 1) \rfloor, 0, H - 1)$
- 5: **end for**
- 6: $p_{\text{corners}} \leftarrow \text{stack}(x_j, y_j)$ ▷ Pixel coordinates of box corners
- 7: $\text{hull} \leftarrow \text{ConvexHull}(p_{\text{corners}})$ ▷ Ordered list of points along the convex hull
- 8: $\text{FillContours}(\mathbf{M}, \text{hull}, 0)$ ▷ Set all mask pixels inside the convex hull to 0
- 9: **return** $\mathbf{M}$

2.3 Training Data Filtering

We train our model on 407k shapes from TRELLIS-500K dataset [22] which consists of aesthetics filtered objects from ObjaverseXL [3], ABO [2], 3D-FUTURE [4], and HSSD [10]. While the ObjaverseXL subset in TRELLIS-500K has already been filtered for aesthetics, it still contains many entries that correspond to scene-level assets rather than single objects, which are the focus of our work. To address this, we leverage the G-Objaverse [23] dataset, which provides human-provided category labels for the original Objaverse dataset. It defines 10 general categories, including Human-Shape (41,557), Animals (28,882), Daily-Used (220,222), Furnitures (19,284), Buildings & Outdoor (116,545), Transportations (20,075), Plants (7,195), Food (5,314), Electronics (13,252), and Poor-quality (107,001). To focus on single-object assets, we remove entries in the Buildings & Outdoor category. We accomplish this by training a classifier head on top of pre-trained DINOv3 embeddings. Given a rendered view of an object, the classifier predicts its general category, and we aggregate predictions across multiple views by averaging to obtain a robust per-object classification. This approach allows us to automatically filter out scene-like objects for training, removing 66k objects from our downloaded subset of TRELLIS-500K. Examples of objects removed are shown in Fig.1.



Fig. 2: 3D-Consistent Masks: Example of masks with generated occluders.

3 Experiment Details

3.1 Occlusion Mask Generation

For quantitative and qualitative evaluation, we simulate realistic 3D-consistent occlusions by adding random occluders from a set of basic shapes (box, cylinder, sphere, and cone). In Amodal3R [21], a 3D-consistent mask is generated with a random-walk strategy to iteratively select neighboring faces of a mesh. Instead of sampling directly from the mesh surface—which can reveal the geometry of the occluded region—we use occluder shapes to prevent the model from inferring the hidden structure from the mask. For each mesh, we sample 3 occluders from the set of basic shapes with equal probability.

For each mesh, we sample three occluders from the set of basic shapes with equal probability. To create an occluder, we first compute the object axis-aligned bounding box and its volume V_{obj} . We then sample a linear scale $s \sim U(0.25, 5)$ and define the target occluder volume as

$$V_{occ} = s^3 \cdot V_{obj}. \quad (3)$$

To determine the occluder position, we uniformly sample 100k points from the mesh surface and apply voxel downsampling (with voxel size 0.1) to obtain a spatially distributed set of candidate locations. One point is randomly selected as the occluder center, ensuring the occluder is placed near the object geometry.

Given the target volume V_{occ} , the dimensions of each occluder type are sampled as follows:

- **Box.** Independent aspect ratios $r_x, r_y, r_z \sim U(0.6, 1.4)$ are sampled, and the side lengths are scaled to match the target volume V_{occ} , yielding

$$l_i = r_i \left(\frac{V_{occ}}{r_x r_y r_z} \right)^{1/3}, \quad i \in \{x, y, z\}.$$

This ensures that $l_x \cdot l_y \cdot l_z = V_{occ}$.

- **Sphere.** The radius is computed directly from the target volume:

$$r = \left(\frac{3V_{occ}}{4\pi} \right)^{1/3}.$$

- **Cylinder.** The aspect ratio between height and diameter is sampled as $a \sim U(2.0, 6.0)$. The radius is then

$$r = \left(\frac{V_{occ}}{2\pi a} \right)^{1/3},$$

and the height is $h = 2ar$.

- **Cone.** The height-to-radius ratio is sampled as $a \sim U(1.5, 4.0)$. The radius is

$$r = \left(\frac{3V_{occ}}{\pi a} \right)^{1/3},$$

and the height is $h = ar$.

Each occluder is centered at the sampled location and used to render the final occlusion mask in each camera view. Example masks are shown in Fig. 2.

3.2 View Selection for Hunyuan3D-Omni

During evaluation, we used Hunyuan3D-Omni (Hy3D-Omni) [7] as a baseline for both single- and multi-view completion. Although Hy3D-Omni is designed to take only a single-view image as input, it can be conditioned on point clouds from multiple views. To leverage this capability, we select the single input image that provides the most informative view while conditioning on multi-view points. Below, we detail our view selection strategy.

For each view, with occlusion mask M_{occ} and foreground object mask M_{obj} , we compute two metrics:

$$F_m = \frac{\sum M_{occ}}{\sum M_{obj}}, \quad F_c = \frac{\sum M_{obj}}{R^2}, \quad (4)$$

where F_m denotes the fraction of foreground pixels that are occluded, F_c is the fraction of image pixels occupied by the object, and R represents the image resolution. For unoccluded scenarios, the view with the largest object coverage, *i.e.*, the view that maximizes F_c , is selected. In occluded scenarios, the view is selected to maximize both object coverage and visibility by choosing the view that maximizes $F_c \cdot (1 - F_m)$. This procedure ensures that the selected image provides the most visible and informative view of the object for Hy3D-Omni.

3.3 Image-to-3D with MapAnything

In our paper, we showed results on 3D shape generation directly from images using MapAnything [8] outputs. Here we provide more details on the pipeline, specifically regarding object orientation. The input image is first segmented using SAM 2 [16] to obtain the object segment and ground floor segment. The input image is then passed to MapAnything [8] to produce points and camera parameters. The object and ground points are obtained using the segmentation masks. To match the orientation of objects in the training data (+Y up, +Z forward), the ground points are first used to estimate a ground plane, and the points are rotated so that its normal aligns with the +Y axis. The object points are then projected onto the X-Z plane, and their dominant horizontal axis (first principal component) is aligned with +Z using singular value decomposition (SVD). Alternatively, when ground points are unavailable, Orient Anything

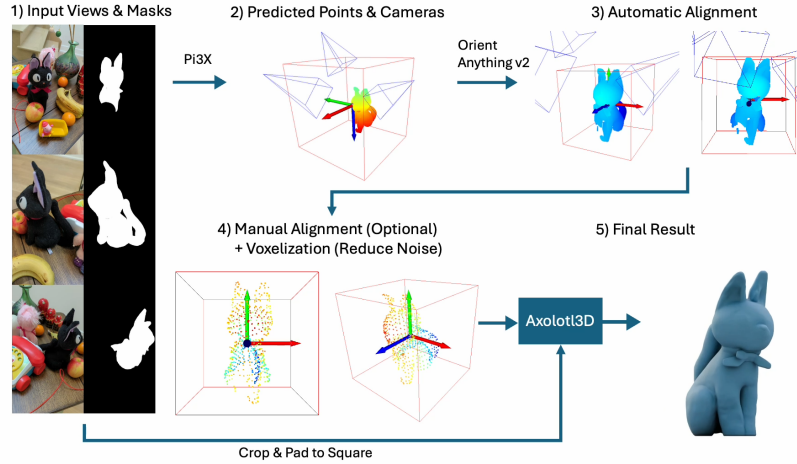


Fig. 3: Image-to-3D Pipeline: Fully automated with SAM 2 [16] segmentation, MapAnything [8] points and cameras, and Orient Anything v2 [19] point alignment.

v2 [19] can be applied to a view to predict the orientation as shown in Fig. 3 but may require manual correction. Finally, the points are centered and scaled to $[-1, 1]$. Depending on the surface quality of the predicted points, we apply voxel downsampling and outlier point removal to reduce surface artifacts in our predict output. See §5 for further discussion regarding limitations.

3.4 3D Capture & Simulation

Accurate geometry is important for applications such as robot simulation in digital twins or interaction in captured environments. Here we show extraction of objects from a real-world captured scene and simulation in the trained gaussian splat scene.

Generating Meshes from Captures: For each object, we manually select a set of representative views and segment the object using SAM 2 [16]. The segmented regions are then cropped and padded to form the input images. Since multi-view depth prediction methods often suffer from view alignment inconsistencies, we instead estimate depth from a single image using Depth Anything 3 [11]. To enable comparison with the gaussian splatting [9] scene representation, we scale and shift the predicted depth to align with the depth rendered from the gaussian splat scene. The aligned depth map is then masked using object segmentation masks and back-projected using the corresponding camera parameters from the gaussian splat scene to obtain aligned conditioning points. These points are subsequently manually oriented to $+Y$ up, $+Z$ forward and rescaled to the range $[-1, 1]$ to match the input convention of our method. The inverse transformation will be applied to the predicted output to revert to the original scene coordinate frame. Depending on the quality of the predicted depth,

Table 1: Quantitative evaluation without ICP alignment on Toys4k.

Scenario: Method	Single-View			Multi-View		
	F-score $\uparrow$	vIoU $\uparrow$	CD $\downarrow_{\times 10}$	F-score $\uparrow$	vIoU $\uparrow$	CD $\downarrow_{\times 10}$
<i>w/ occlusion:</i>						
Hy3D-Omni	0.6166 ± 0.2762	0.1800 ± 0.1291	0.8635 ± 0.8270	0.8646 ± 0.1809	0.3346 ± 0.1574	0.3092 ± 0.3803
ShapeR (concurrent)	0.6851 ± 0.2545	0.2282 ± 0.1203	0.5237 ± 0.4498	0.8901 ± 0.1896	0.3998 ± 0.1554	0.2448 ± 0.3854
Ours	0.8898 ± 0.1328	0.3277 ± 0.1035	0.2559 ± 0.2207	0.9615 ± 0.0762	0.4196 ± 0.1126	0.1540 ± 0.0865
<i>w/o occlusion:</i>						
Hy3D-Omni	0.7512 ± 0.2257	0.2486 ± 0.1462	0.4858 ± 0.4408	0.9071 ± 0.1351	0.3849 ± 0.1591	0.2333 ± 0.2254
ShapeR (concurrent)	0.7386 ± 0.2445	0.2566 ± 0.1230	0.4549 ± 0.4318	0.9294 ± 0.1546	0.4470 ± 0.1450	0.1897 ± 0.3343
Ours	0.9118 ± 0.1171	0.3481 ± 0.0986	0.2278 ± 0.1942	0.9767 ± 0.0468	0.4438 ± 0.1078	0.1372 ± 0.0571

we apply voxel downsampling and outlier point removal to reduce noise. To incorporate cropped multi-view image conditions, we apply our method without camera conditioning, as described in the ablation section of the main paper.

Simulation: We follow the Kaolin hand-on lab [12] that utilizes representation-agnostic physics simulation [13] and mixed splat-mesh rendering [14, 20] to show our results; splat objects are segmented following an algorithm similar to [5]. The resulting simulation is shown in the supplementary video. Unlike gaussian splat object segments which contain large missing areas, our geometries are complete, and can be used for more accurate physics simulation of the captured environment.

4 Additional Results

4.1 Quantitative Results without ICP

We apply ICP to all methods for fairness as Amodal3R does not use point conditioning and SAM3D uses camera-space pointmaps. Metrics without alignment would largely reflect coordinate-frame mismatch rather than reconstruction accuracy. Tb. 1 shows results without ICP. Scores decrease for all methods and relative rankings mostly unchanged.

4.2 Object Edit Comparisons

In Fig. 4, we show comparisons against point conditioned baselines in object edit task. The baselines were not explicitly trained for object editing, and therefore struggle to handle more challenging edits. Hunyuan3D-Omni (Hy3D-Omni) lacks camera conditioning, which results in ambiguity in object orientation. In

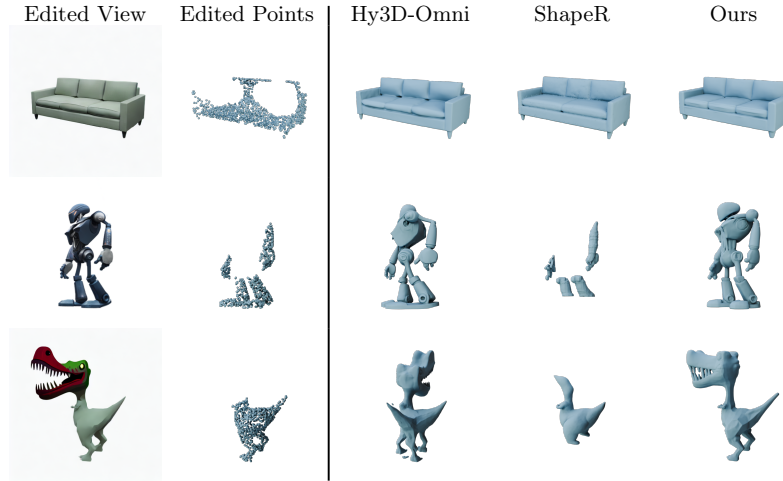


Fig. 4: Object edit comparison against baselines with 2048 input points.

contrast, our method respects the conditioning points to preserve unedited regions, while jointly leveraging camera parameters and the input image to produce faithful edits.

4.3 Qualitative Comparisons

In this section, we provide additional qualitative comparisons against baseline methods to further demonstrate the performance of our method. Fig. 5 and 6 show results on the OmniObject3D dataset under occluded and unoccluded conditions. Fig. 7 illustrates unoccluded scenarios on Toys4k.

5 Limitations & Future Work

While Axolotl3D demonstrates strong performance in occlusion handling and downstream applications, there remain several avenues for further improvement that we plan to explore in future work.

- **Camera and view robustness:** Currently, our model is trained with fixed camera distances and intrinsics, which requires the object to be fully in view. Future work could incorporate more diverse camera positioning and image augmentations or real-world captured datasets to improve robustness to camera viewpoints.
- **Surface quality and fine details:** While our approach produces high-quality reconstructions overall, extremely thin structures can sometimes be disconnected and fine surface details may be smoothed or thicker than in reality. Leveraging more powerful base models or image encoders could enable better reconstruction of finer geometry.
- **Robustness to noise:** Our current model is trained to follow geometric cues faithfully and has not yet been explicitly trained to handle noisy input.

Extending the model to handle different noise characteristics is an important direction for more general applicability. Examples of such characteristics include view misalignment in 3D reconstruction models [8, 18], irregular surfaces and outliers seen in Gaussian splats [9] or oversmoothed surfaces, perspective-induced distortions, and artifacts near depth discontinuities observed in monocular depth estimation [11].

- **Handling diverse point characteristics:** Our method is trained on partial point clouds downsampled to 8192 points using Furthest Point Sampling [15], with padding for smaller inputs. While this allows handling moderately sparse inputs, the model is not explicitly trained for very sparse or sharp-edge points (i.e., from Structure-from-Motion [17]). Adapting the model to handle a wider range of point distributions and characteristics would further broaden its applicability to more complex real-world datasets.
- **Texture completion:** Axolotl3D focuses solely on shape generation and does not predict textures, which could be explored in future work.

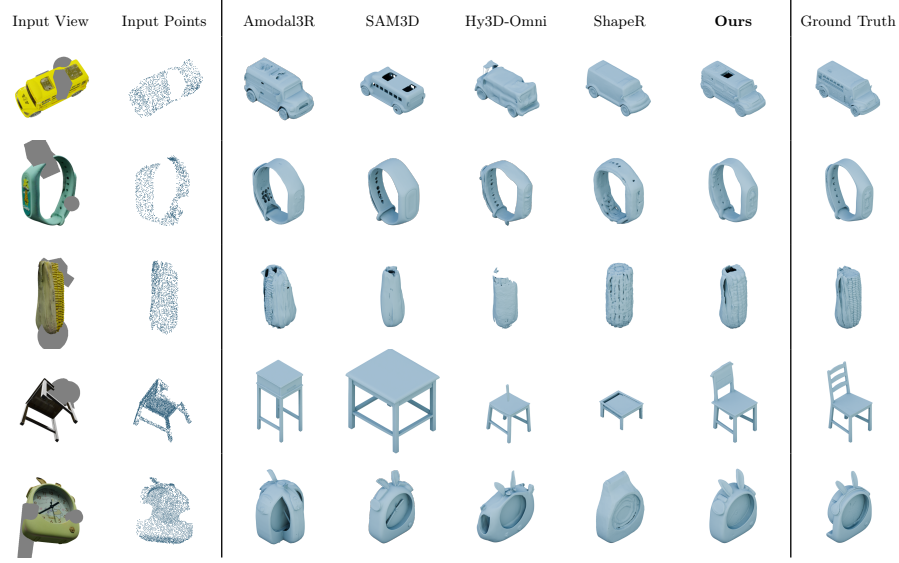
Overall, these limitations highlight exciting opportunities to further enhance Axolotl3D, extending its applicability and robustness to a wider range of challenging scenarios.

References

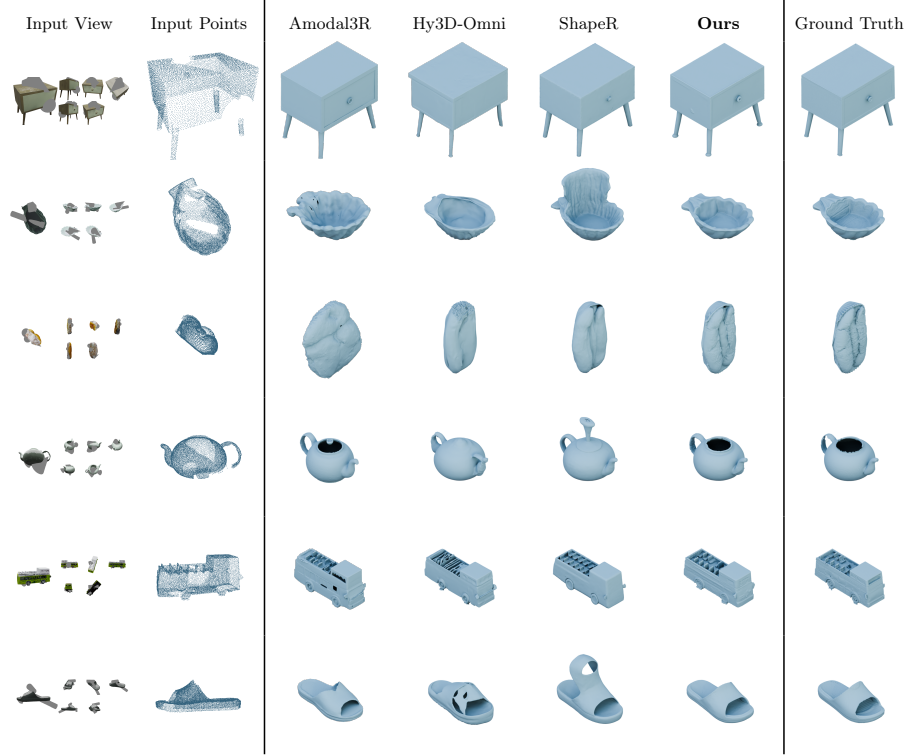
1. Chen, X., Chu, F.J., Gleize, P., Liang, K.J., Sax, A., Tang, H., Wang, W., Guo, M., Hardin, T., Li, X., et al.: Sam 3d: 3dfy anything in images. arXiv preprint arXiv:2511.16624 (2025)
2. Collins, J., Goel, S., Deng, K., Luthra, A., Xu, L., Gundogdu, E., Zhang, X., Vicente, T.F.Y., Dideriksen, T., Arora, H., Guillaumin, M., Malik, J.: Abo: Dataset and benchmarks for real-world 3d object understanding. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 21126–21136 (June 2022)
3. Deitke, M., Liu, R., Wallingford, M., Ngo, H., Michel, O., Kusupati, A., Fan, A., Laforte, C., Voleti, V., Gadre, S.Y., et al.: Objaverse-xl: A universe of 10m+ 3d objects. Advances in Neural Information Processing Systems **36**, 35799–35813 (2023)
4. Fu, H., Jia, R., Gao, L., Gong, M., Zhao, B., Maybank, S., Tao, D.: 3d-future: 3d furniture shape with texture. International Journal of Computer Vision **129**(12), 3313–3337 (2021). <https://doi.org/10.1007/s11263-021-01534-z>
5. Fuji Tsang, C., Hu, A., Perel, O., Kolve, C., Shugrina, M.: Artisans: Interactive tools for gaussian splat selection with ai and human in the loop. arXiv preprint arXiv:2602.10173 (2026)
6. Grattafiori, A., Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Vaughan, A., et al.: The llama 3 herd of models. arXiv preprint arXiv:2407.21783 (2024)
7. Hunyuan3D, T., Zhang, B., Guo, C., Liu, H., Yan, H., Shi, H., Huang, J., Yu, J., Li, K., Wang, P., et al.: Hunyuan3d-omni: A unified framework for controllable generation of 3d assets. arXiv preprint arXiv:2509.21245 (2025)

8. Keetha, N., Müller, N., Schönberger, J., Porzi, L., Zhang, Y., Fischer, T., Knapitsch, A., Zaus, D., Weber, E., Antunes, N., Luiten, J., Lopez-Antequera, M., Bulò, S.R., Richardt, C., Ramanan, D., Scherer, S., Kotschieder, P.: MapAnything: Universal feed-forward metric 3D reconstruction. In: International Conference on 3D Vision (3DV). IEEE (2026)
9. Kerbl, B., Kopanas, G., Leimkuehler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions On Graphics (TOG)* **42**(4) (2023). <https://doi.org/10.1145/3592433>, <https://doi.org/10.1145/3592433>
10. Khanna, M., Mao, Y., Jiang, H., Haresh, S., Shacklett, B., Batra, D., Clegg, A., Undersander, E., Chang, A.X., Savva, M.: Habitat synthetic scenes dataset (hssd-200): An analysis of 3d scene scale and realism tradeoffs for objectgoal navigation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 16384–16393 (June 2024)
11. Lin, H., Chen, S., Liew, J.H., Chen, D.Y., Li, Z., Zhao, Y., Peng, S., Guo, H., Zhou, X., Shi, G., Feng, J., Kang, B.: Depth anything 3: Recovering the visual space from any views. In: The Fourteenth International Conference on Learning Representations (2026), <https://openreview.net/forum?id=yirunib8l8>
12. Modi, V., Fuji Tsang, C., Shugrina, M., Daviet, G.: Tackling gaussian splats, physics simulation and visualization with nvidia kaolin and warp libraries. In: Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Labs. SIGGRAPH Labs '25, Association for Computing Machinery, New York, NY, USA (2025). <https://doi.org/10.1145/3721251.3736529>, <https://doi.org/10.1145/3721251.3736529>
13. Modi, V., Sharp, N., Perel, O., Sueda, S., Levin, D.I.: Simplicits: Mesh-free, geometry-agnostic elastic simulation. *ACM Transactions on Graphics (TOG)* **43**(4), 1–11 (2024)
14. Moenne-Loccoz, N., Mirzaei, A., Perel, O., de Lutio, R., Martinez Esturo, J., State, G., Fidler, S., Sharp, N., Gojcic, Z.: 3d gaussian ray tracing: Fast tracing of particle scenes. *ACM Transactions on Graphics (TOG)* **43**(6), 1–19 (2024)
15. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In: Advances in Neural Information Processing Systems. vol. 30 (2017), https://proceedings.neurips.cc/paper_files/paper/2017/file/d8bf84be3800d12f74d8b05e9b89836f-Paper.pdf
16. Ravi, N., Gabeur, V., Hu, Y.T., Hu, R., Ryali, C., Ma, T., Khedr, H., Rädle, R., Rolland, C., Gustafson, L., Mintun, E., Pan, J., Alwala, K.V., Carion, N., Wu, C.Y., Girshick, R., Dollar, P., Feichtenhofer, C.: SAM 2: Segment anything in images and videos. In: The Thirteenth International Conference on Learning Representations (2025), <https://openreview.net/forum?id=Ha6RTeWMd0>
17. Schonberger, J.L., Frahm, J.M.: Structure-from-motion revisited. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2016)
18. Wang, J., Chen, M., Karaev, N., Vedaldi, A., Rupprecht, C., Novotny, D.: Vggt: Visual geometry grounded transformer. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5294–5306 (June 2025)
19. Wang, Z., Zhang, Z., Xu, J., Wang, J., Pang, T., Du, C., Zhao, H., Zhao, Z.: Orient anything v2: Unifying orientation and rotation understanding. In: The Thirtieth Annual Conference on Neural Information Processing Systems (2026), <https://openreview.net/forum?id=n3armuTFit>

20. Wu, Q., Martinez Esturo, J., Mirzaei, A., Moenne-Loccoz, N., Gojcic, Z.: 3dgut: Enabling distorted cameras and secondary rays in gaussian splatting. Conference on Computer Vision and Pattern Recognition (CVPR) (2025)
21. Wu, T., Zheng, C., Guan, F., Vedaldi, A., Cham, T.J.: Amodal3r: Amodal 3d reconstruction from occluded 2d images. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9181–9193 (2025)
22. Xiang, J., Lv, Z., Xu, S., Deng, Y., Wang, R., Zhang, B., Chen, D., Tong, X., Yang, J.: Structured 3d latents for scalable and versatile 3d generation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 21469–21480 (2025)
23. Zuo, Q., Gu, X., Dong, Y., Zhao, Z., Yuan, W., Qiu, L., Bo, L., Dong, Z.: High-fidelity 3d textured shapes generation by sparse encoding and adversarial decoding. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 52–69 (2024). https://doi.org/10.1007/978-3-031-72684-2_4

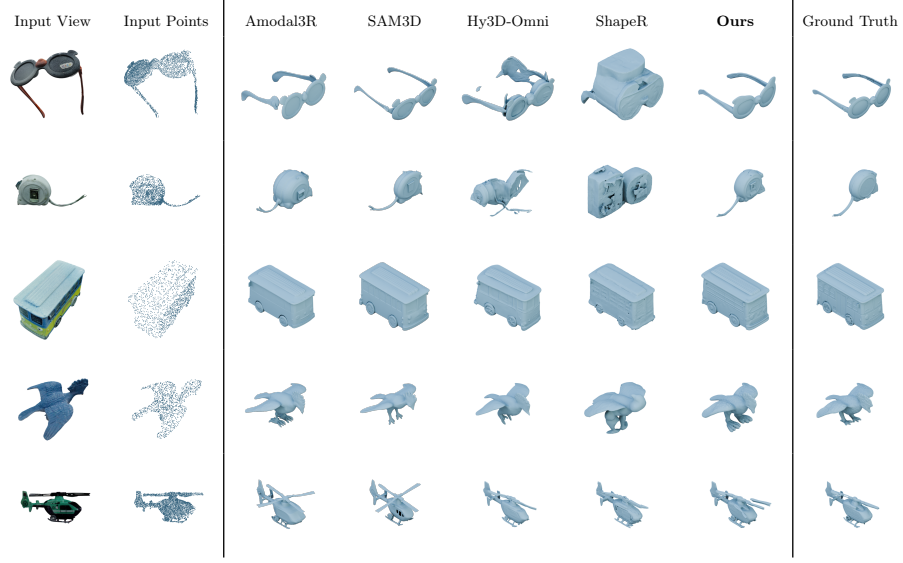


(a) Single-view input results

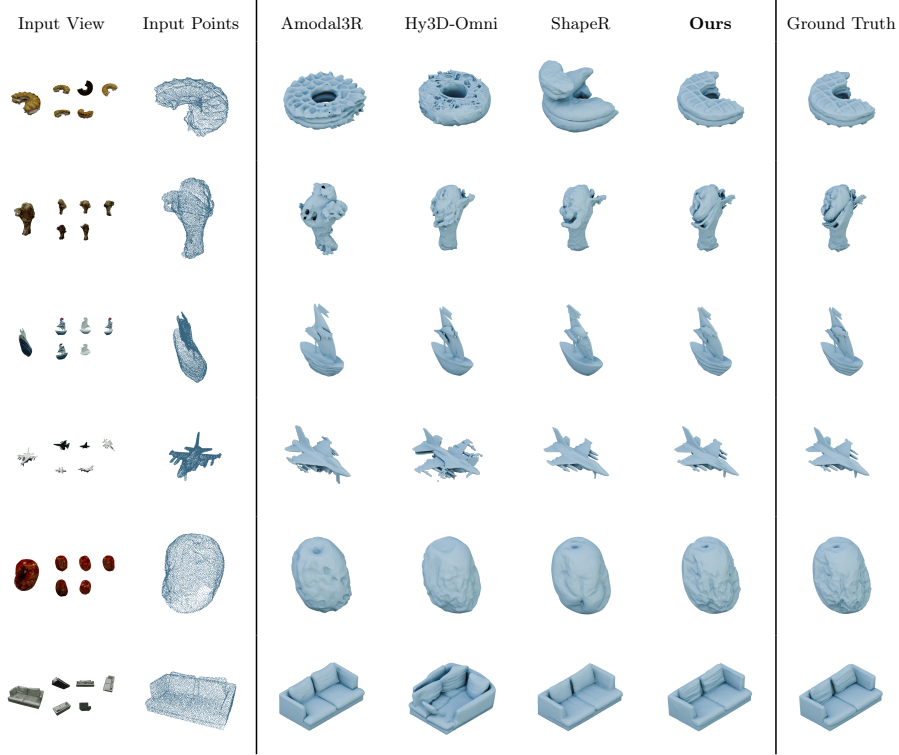


(b) Multi-view input results

Fig. 5: Qualitative Comparisons on OmniObject3D: Single- and multi-view completion results against baselines with occlusions.

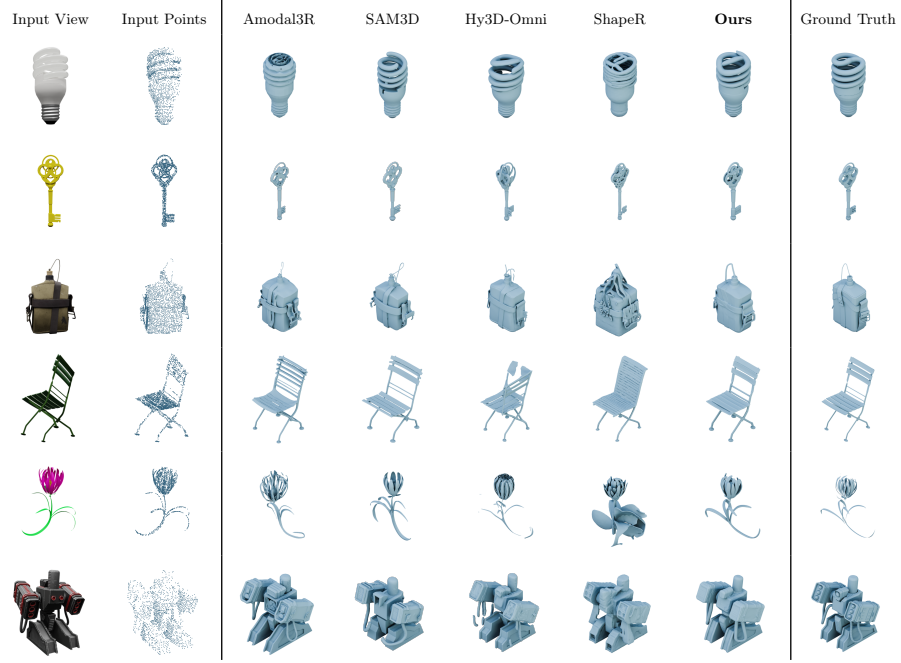


(a) Single-view input results

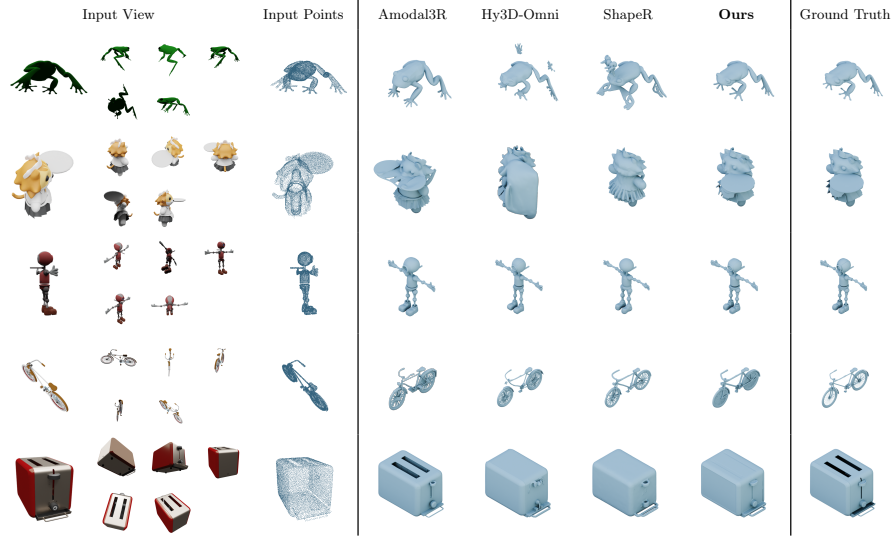


(b) Multi-view input results

Fig. 6: Qualitative Comparisons on OmniObject3D: Single- and multi-view completion results against baselines without occlusions.



(a) Single-view input results



(b) Multi-view input results

Fig. 7: Qualitative Comparisons on Toys4k: Single- and multi-view completion results against baselines without occlusions.