

SAM 3D: 3Dfy Anything in Images

SAM 3D Team, Xingyu Chen*, Fu-Jen Chu*, Pierre Gleize*, Kevin J Liang*, Alexander Sax*, Hao Tang*, Weiyao Wang*, Michelle Guo, Thibaut Hardin, Xiang Li[°], Aohan Lin, Jiawei Liu, Ziqi Ma[°], Anushka Sagar, Bowen Song[°], Xiaodong Wang, Jianing Yang[°], Bowen Zhang[°], Piotr Dollár[†], Georgia Gkioxari[†], Matt Feiszli^{†§}, Jitendra Malik^{†§}

Meta Superintelligence Labs

*Core Contributor (Alphabetical, Equal Contribution), [°]Intern, [†]Project Lead, [§]Equal Contribution

We present SAM 3D, a generative model for visually grounded 3D object reconstruction, predicting geometry, texture, and layout from a single image. SAM 3D excels in natural images, where occlusion and scene clutter are common and visual recognition cues from context play a larger role. We achieve this with a human- and model-in-the-loop pipeline for annotating object shape, texture, and pose, providing visually grounded 3D reconstruction data at unprecedented scale. We learn from this data in a modern, multi-stage training framework that combines synthetic pretraining with real-world alignment, breaking the 3D “data barrier”. We obtain significant gains over recent work, with at least a 5 : 1 win rate in human preference tests on real-world objects and scenes. We will release our code and model weights, an online demo, and a new challenging benchmark for in-the-wild 3D object reconstruction.

Demo: <https://www.aidemos.meta.com/segment-anything/editor/convert-image-to-3d>

Code: <https://github.com/facebookresearch/sam-3d-objects>

Website: <https://ai.meta.com/sam3d>

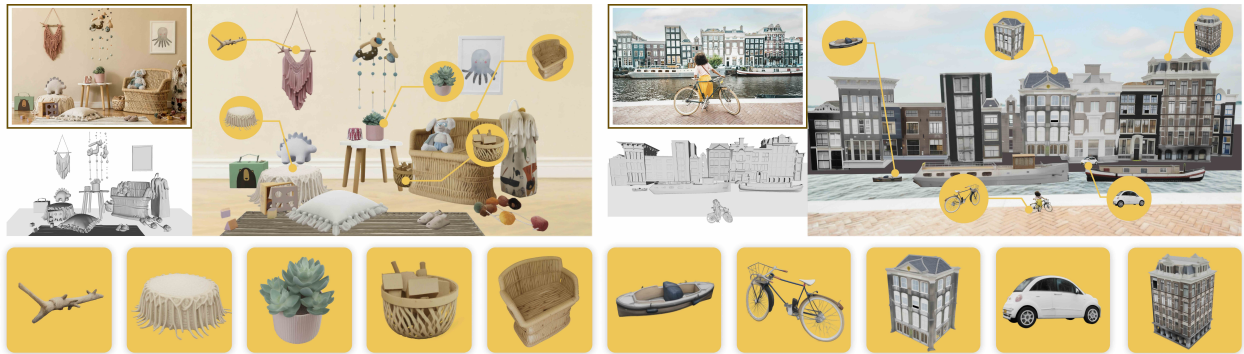


Figure 1 SAM 3D converts a single image into a composable 3D scene made of individual objects. Our method predicts per-object geometry, texture, and layout, enabling full scene reconstruction. Bottom: high-quality 3D assets recovered for each object.

1 Introduction

In this paper (see Figure 1) we present SAM 3D, a generative neural network for 3D reconstruction from a single image. The model can reconstruct 3D shape and texture for any object, as well as its layout with respect to the camera, even in complex scenes with significant clutter and occlusion. As the reconstruction is of full 3D shape, not just of the visible 2.5D surface, one can then re-render the object from any desired viewpoint.

Computer vision has traditionally focused on multi-view geometry as providing the primary signal for 3D shape. However psychologists (and artists before them) have long known that *humans* can perceive depth and

shape from a single image, *e.g.* Koenderink et al. (1992) demonstrated this elegantly by showing that humans can estimate surface normals at probe points on an object’s image, which can then be integrated to a full surface. In psychology textbooks these single image cues to 3D shape are called “pictorial cues”, and include information such as in shading and texture patterns, but also recognition - the “familiar object” cue. In computer vision, this line of research dates back to Roberts (1963), who showed that once an image pattern was recognized as a known object, its 3D shape and pose could be recovered. The central insight is that recognition enables 3D reconstruction, an idea that has since resurfaced in different technical instantiations (Debevec et al., 2023; Cashman and Fitzgibbon, 2012; Kar et al., 2015; Gkioxari et al., 2019; Xiang et al., 2025). Note that this permits generalization to novel objects, because even if a specific object has not been seen before, it is made up of parts seen before.

A fundamental challenge for learning such models is the lack of data: specifically, natural images paired with 3D ground truth are difficult to obtain at scale. Recent work (Yang et al., 2024b; Xiang et al., 2025) has shown strong reconstruction from single images. However, these models are trained on isolated objects and struggle with objects in natural scenes, where they may be distant or heavily occluded. To add such images to the training set, we need to find a way to associate specific objects in such images with 3D shape models, acknowledging that generalist human annotators find it hard to do so (unlike, say, attaching a label like “cat” or marking its boundary). Two insights made this possible:

- We can create synthetic scenes where 3D object models are rendered and pasted into images (inspired by Dosovitskiy et al. (2015)).
- While humans can’t easily *generate* 3D shape models for objects, they can *select* the likely best 3D model from a set of proffered choices and align its pose to the image (or declare that none of the choices is good).

We design a training pipeline and data engine by adapting modern, multistage training recipes pioneered by LLMs (Minaei et al., 2025; Mo et al., 2025). As in recent works, we first train on a large collection of rendered synthetic objects. This is supervised pretraining: our model learns a rich vocabulary for object shape and texture, preparing it for real-world reconstruction. Next is mid-training with semi-synthetic data produced by pasting rendered models into natural images. Finally, post-training adapts the model to real images, using both a novel model-in-the-loop (MITL) pipeline and human 3D artists, and aligns it to human preference. We find that synthetic pretraining generalizes, given adequate post-training on natural images.

Our post-training data, obtained from our MITL data pipeline, is key to obtaining good performance in natural images. Generalist human annotators aren’t capable of producing 3D shape ground truth; hence our annotators select and align 3D models to objects in images from the output of modules – computational and retrieval-based – that produce multiple initial 3D shape proposals. Human annotators select from these proposals, or route them to human artists for a subset of hard instances. The vetted annotations feed back into model training, and the improved model is reintegrated into the data engine to further boost annotation quality. This virtuous cycle steadily improves the quality of 3D annotations, labeling rates, and model performance.

Due to the lack of prior benchmarks for real-world 3D reconstruction of object shape and layout, we propose a new evaluation set of 1,000 image and 3D pairs: SAM 3D Artist Objects (SA-3DAO). The objects in our benchmark range from churches, ski lifts, and large structures to animals, everyday household items, and rare objects, and are paired with the real-world images in which they naturally appear. Professional 3D artists create 3D shapes from the input image, representing an expert human upper bound for visually grounded 3D reconstruction. We hope that contributing such an evaluation benchmark helps accelerate subsequent research iteration of real-world 3D reconstruction models.

We summarize our contributions as follows:

- We introduce **SAM 3D**, a new foundation model for 3D that predicts object shape, texture, and pose from a single image. By releasing code, model weights, and a demo, we hope to stimulate further advancements in 3D reconstruction and downstream applications of 3D.
- We build a MITL pipeline for annotating shape, texture, and pose data, providing visually grounded 3D reconstruction data at unprecedented scale.

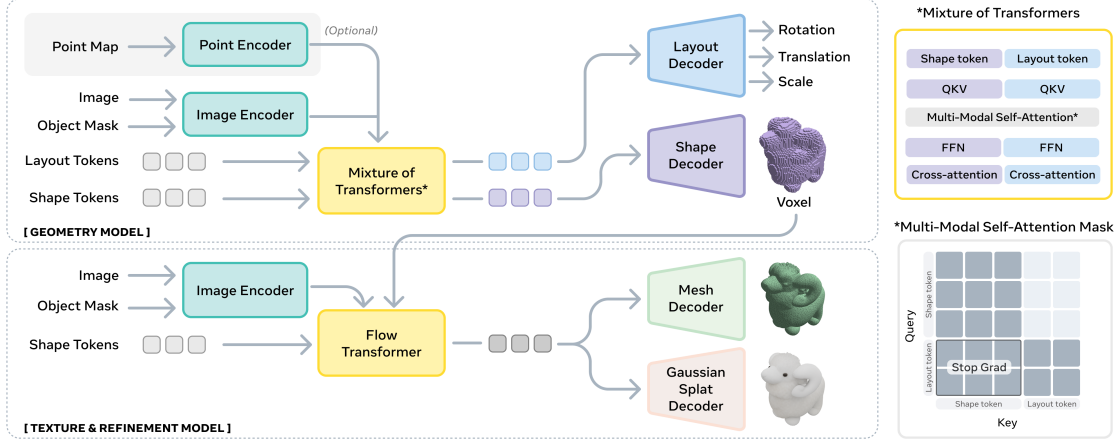


Figure 2 SAM 3D architecture. (top) SAM 3D first predicts coarse shape and layout with the Geometry model; (right) the mixture of transformers architecture apply a two-stream approach with information sharing in the multi-modal self-attention layer. (bottom) The voxels predicted by the Geometry model are passed to the Texture & Refinement model, which adds higher resolution detail and textures.

- We exploit this data via LLM-style pretraining and post-training in a novel framework for 3D reconstruction, combining synthetic pretraining with real-world alignment to overcome the orders of magnitude data gap between 3D and domains such as text, images, or video.
- We release a challenging benchmark for real-world 3D object reconstruction, SA-3DAO. Experiments show SAM 3D’s significant gains via metrics and large-scale human preference.

2 The SAM 3D Model

2.1 Problem Formulation

The act of taking a photograph maps a 3D object to a set of 2D pixels, specified by a mask M in an image I . We seek to invert this map. Let the object have shape S , texture T , and rotation, translation and scale (R, t, s) in camera coordinates. Since the 3D to 2D map is lossy, we model the reconstruction problem as a conditional distribution $p(S, T, R, t, s | I, M)$. Our goal is to train a generative model $q(S, T, R, t, s | I, M)$ that approximates p as closely as possible.

2.2 Architecture

We build upon recent SOTA two-stage latent flow matching architectures (Xiang et al., 2025). SAM 3D first jointly predicts object pose and coarse shape, then refines the shapes by integrating pictorial cues (see Figure 2). Unlike Xiang et al. (2025) that reconstructs isolated objects, SAM 3D predicts object layout, creating coherent multi-object scenes.

Input encoding. We use DINOv2 (Oquab et al., 2023) as an encoder to extract features from two pairs of images, resulting in 4 sets of conditioning tokens:

- **Cropped object:** We encode the cropped image I by mask M and its corresponding *cropped binary mask*, providing a focused, high-resolution view of the object.
- **Full image:** We encode the full image I and its *full image binary mask*, providing global scene context and recognition cues absent from the cropped view.

Optionally, the model supports conditioning on a coarse scene point map, P obtained via hardware sensors (e.g., LiDAR on an iPhone), or monocular depth estimation (Yang et al., 2024a; Wang et al., 2025a), enabling SAM 3D to integrate with other pipelines.

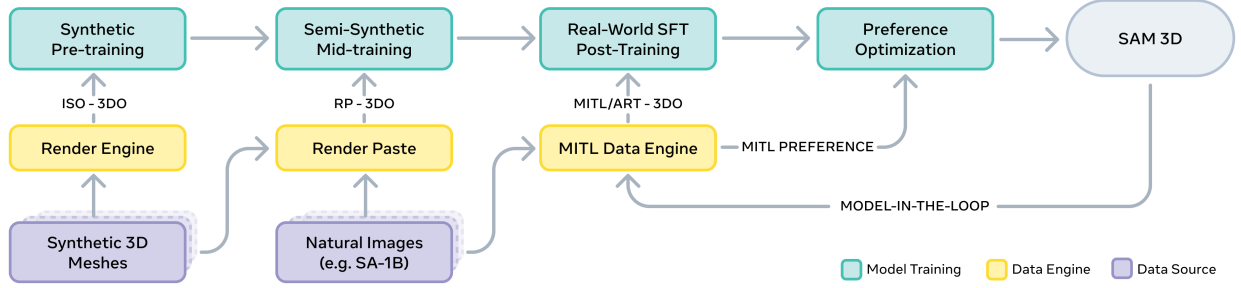


Figure 4 SAM 3D training paradigm. We employ a multi-stage pipeline incrementally exposing the model to increasingly complex data and modalities.

The Geometry Model models the conditional distribution $p(O, R, t, s|I, M)$, where $O \in \mathbb{R}^{64^3}$ is coarse shape, $R \in \mathbb{R}^6$ the 6D rotation (Zhou et al., 2019), $t \in \mathbb{R}^3$ the translation, and $s \in \mathbb{R}^3$ the scale. Conditioned on the input image and mask encodings, we employ a 1.2B parameter flow transformer with the Mixture-of-Transformers (MoT) architecture (Liang et al., 2025a; Deng et al., 2025), modeling geometry O and layout (R, t, s) using the attention mask in Figure 2. See Section C.1 for details.

The Texture & Refinement Model learns the conditional distribution $p(S, T|I, M, O)$. We first extract active voxels from the coarse shape O predicted by Geometry model. A 600M parameter sparse latent flow transformer (Xiang et al., 2025; Peebles and Xie, 2023) refines geometric details and synthesizes object texture.

3D Decoders. The latent representations from the Texture & Refinement Model can be decoded to either mesh or 3D Gaussian splats via a pair of VAE decoders $\mathcal{D}_m, \mathcal{D}_g$. These separately-trained decoders share the same VAE encoder and hence the same structured latent space (Xiang et al., 2025). We also detail several improvements in Section C.6.

3 Training SAM 3D

SAM 3D breaks the 3D data barrier using a recipe that progresses from synthetic pretraining to natural post-training, adapting the playbook from LLMs, robotics, and other large generative models. We build capabilities by stacking different training strategies in pre- and mid-training, and then align the model to real data and human-preferred behaviors through a post-training data flywheel. SAM 3D uses the following approach:

Step 1: Pretraining. This phase builds foundational capabilities, such as shape generation, into a base model.

Step 1.5: Mid-Training. Sometimes called continued pretraining, mid-training imparts general skills such as occlusion robustness, mask-following, and using visual cues.

Step 2: Post-Training. Post-training elicits target behavior, such as adapting the model from synthetic to real-world data or following human aesthetic preferences. We collect training samples $(I, M) \rightarrow (S, T, R, t, s)$ and preference data from humans and use them in both supervised finetuning (SFT) and direct preference optimization (DPO) (Rafailov et al., 2023).

This alignment (step 2) can be repeated, first collecting data with the current model and then improving the model with the new data. This creates a virtuous cycle with humans providing the supervision. Figure 10b shows that as we run the data engine longer, model performance steadily improves; dataset generation emerges as a byproduct of this alignment.

The following sections detail the training objectives and data sources used in SAM 3D. We focus on the Geometry model; Texture & Refinement is trained similarly (details in Section C.5). Training hyper-parameters are in Section C.7.

| Training stage | Modalities | Datasets | Condition input |
|---|--------------|-------------------------------|-----------------------|
| Stage 1 Geometry model | | | |
| Pre-training | S, R | Iso-3DO | object-centric crop |
| Mid-training | S, R | RP-3DO [†] | full image |
| | S, R, t, s | ProcThor, RP-3DO [‡] | full image, pointmap* |
| SFT | S, R, t, s | MITL, Art-3DO | full image, pointmap* |
| Alignment | S | MITL preference | full image, pointmap* |
| Stage 2 Texture & Refinement model | | | |
| Pre-training | T | Iso-3DO-500K | object-centric crop |
| Mid-training | T | RP-3DO [§] | full image |
| SFT | T | MITL | full image |
| Alignment | T | MITL preference | full image |

Table 1 SAM 3D training stages. [†]Flying Occlusion (FO) from RP-3DO. [‡]Object Swap - Random (OS-R) from RP-3DO. [§]Object Swap - Annotated (OS-A) from RP-3DO. *optional. See Section B.2 for details.

3.1 Pre & Mid-Training: Building a Base Model

Training begins with synthetic pretraining and mid-training, leveraging available large-scale datasets to learn strong priors for shape and texture, and skills such as mask-following, occlusion handling, and pose estimation. The rich features learned here drastically reduce the number of labeled real-world samples required in post-training (Hernandez et al., 2021), which generally incur acquisition costs. In pre- and mid-training, models are trained with rectified conditional flow matching (Liu et al., 2022) to generate multiple 3D modalities (see Section C.2).

3.1.1 Pretraining: Single Isolated 3D Assets

Pretraining trains the model to reconstruct accurate 3D shapes and textures from renders of isolated synthetic objects, following the successful recipes from (Xiang et al., 2025; Yang et al., 2024b; Wu et al., 2024). Specifically, we gather a set of image I , shape S , and texture T triplets, using 2.7 million object meshes from Objaverse-XL (Deitke et al., 2023) and licensed datasets, and render them from 24 viewpoints, each producing a high-resolution image of a single centered object; more detail in Section B.1. We call this dataset *Iso-3DO* and train for 2.5 trillion training tokens.

3.1.2 Mid-Training: Semi-Synthetic Capabilities

Next, mid-training builds up foundational skills that will enable the model to handle objects in real-world images:

- *Mask-following*: We train the model to reconstruct a target object, defined by a binary mask on the input image.
- *Occlusion robustness*: The artificial occluders in our dataset incentivize learning shape completion.
- *Layout estimation*: We train the model to produce translation and scale in normalized camera coordinates.

We construct our data by rendering textured meshes into natural images using alpha compositing. This “render-paste” dataset contains one subset of occluder-occludee pairs, and another subset where we replace real objects with synthetic objects at similar location and scale, creating physically-plausible data with accurate 3D ground truth. We call this data *RP-3DO*; it contains 61 million samples with 2.8 million unique meshes; Figure 3 shows examples. See Section B.2 for more details.

After mid-training (2.7 trillion training tokens), the model has now been trained with all input and output modalities for visually grounded 3D reconstruction. However, all data used has been (semi-)synthetic; to both close the domain gap and fully leverage real-world cues, we need real images.

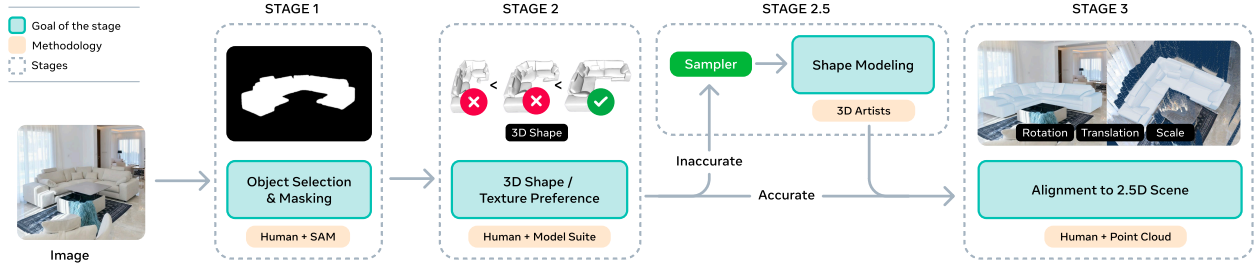


Figure 5 Life of an example going through the data collection pipeline. We streamline annotation by breaking it into subtasks: annotators first choose target objects (Stage 1); rank and select 3D model candidates (Stage 2); then pose these models within a 2.5D scene (Stage 3). Stages 2 and 3 use model-in-the-loop.

3.2 Post-Training: Real-World Alignment

In post-training, we have two goals. The first is to close the domain gap between (semi-)synthetic data and natural images. The second is to align with human preference for shape quality. We adapt the model by using our data engine iteratively; we first **(i) collect training data** with the current model, and then **(ii) update our model** using multi-stage post-training on this collected data. We then repeat.

3.2.1 Post-Training: Collection Step

The core challenge with collecting data for 3D visual grounding is that most people cannot create meshes directly; this requires skilled 3D artists, who even then can take multiple hours. This is different from the segmentation masks collected in SAM (Kirillov et al., 2023). However, given options, most people *can* choose which mesh resembles an object in the image most accurately. This fact forms the foundation of our data collection for SAM 3D. We convert preferences into training data as follows: sample from our post-trained model, ask annotators to choose the best candidate and then grade its overall quality according to a rubric which we define and update. If the quality meets the (evolving) bar, the candidate becomes a training sample.

Unfortunately at the first iteration, our initial model yields few high-quality candidates. This is because before the first collection step, very little real-world data for 3D visual grounding exists. We deal with this cold start problem by leveraging a suite of existing learned and retrieval-based models to produce candidates. In early stages, we draw mostly from the ensemble, but as training progresses our best model dominates, eventually producing about 80% of the annotated data seen by SAM 3D.

Our annotation pipeline collects 3D object shape S , texture T , orientation R , 3D location t , and scale s from real-world images. We streamline the process by dividing into subtasks and leveraging existing appropriate models and human annotators within each (see Figure 5): identifying target objects, 3D model ranking and selection, and posing these within a 3D scene (relative to a point map). We outline each stage of the data engine below and present details in Section A. In total, we annotate almost 1 million images with ~ 3.14 million untextured meshes and $\sim 100K$ textured meshes—unprecedented scale for 3D data paired with natural images.

Stage 1: Choosing target objects (I, M). The goal of this stage is to identify a large, diverse set of images I and object masks M to lift to 3D. To ensure generalization across objects and scenes, we sample images from several diverse real-world datasets, and utilize a 3D-oriented taxonomy to balance the object distribution. To obtain object segmentation masks, we use a combination of pre-existing annotations (Kirillov et al., 2023) and human labelers selecting objects of interest.

Stage 2: Object model ranking and selection (S, T). The goal of this stage is to collect image-grounded 3D shape S and texture T . As described above, human annotators choose shape and texture candidates which best match the input image and mask. Annotators rate the example r and reject chosen examples that do not meet a predefined quality threshold, *i.e.* $r < \alpha$. Bad candidates also become negative examples for preference alignment.

Our data engine maximizes the likelihood of a successful annotation, $r > \alpha$, by asking annotators to choose

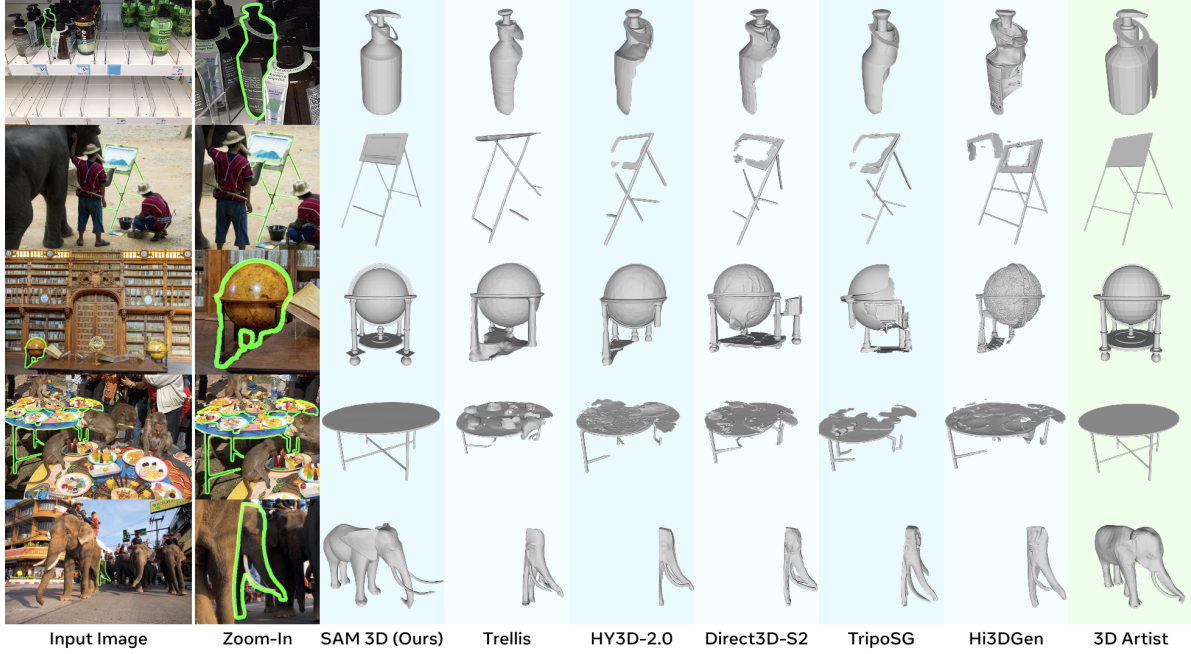


Figure 6 Qualitative comparison to competing image-to-3D asset methods. We compare to the recent Trellis (Xiang et al., 2025), Hunyuan3D-2.1 (Hunyuan3D et al., 2025), Direct3D-S2 (Wu et al., 2025) and Hi3DGen (Ye et al., 2025) on the artist-generated SA-3DAO for single shape reconstruction; we provide the 3D artist-created ground truth mesh as reference.

between $N = 8$ candidates from the ensemble; a form of best-of- N search (Ouyang et al., 2022) using humans. The expected quality of this best candidate improves with N , and we further increase N by first filtering using a model, and then filtering using the human (Anthony et al., 2017); we show results in Section A.7.

Stage 2.5: Hard example triage (Artists). When no model produces a reasonable object shape, our non-specialist annotators cannot correct the meshes, resulting in a lack of data precisely where the model needs it most. We route a small percentage of these hardest cases to professional 3D artists for direct annotation, and we denote this set *Art-3DO*.

Stage 3: Aligning objects to 2.5D scene (R, t, s). The previous stages produce a 3D shape for the object, but not its layout in the scene. For each stage 2 shape, annotators label the object pose by manipulating the 3D object’s translation, rotation, and scale relative to a point cloud. We find that point clouds provide enough structure to enable consistent shape placement and orientation.

In general, we can think of the data collection as an API that takes a current best model, $q(S, T, R, t, s \mid I, M)$, and returns (i) training samples $D^+ = (I, M, S, T, R, t, s)$, (ii) a quality rating $r \in [0, 1]$, and (iii) a set of less preferred candidates ($D^- = (I, M, S', T', R', t', s')$) that are all worse than the training sample.

3.2.2 Post-Training: Model Improvement Step

The model improvement step in SAM 3D uses these training samples and preference results to update the base model through multiple stages of finetuning and preference alignment. Within each post-training iteration we aggregate data from all previous collection steps; keeping only samples where D^+ is above some quality threshold α . As training progresses, α can increase over time, similar to the cross-entropy method for optimization (de Boer et al., 2005). Our final post-training iteration uses 0.5 trillion training tokens.

Supervised Fine-Tuning (SFT). When post-training begins, the base model has only seen synthetic data. Due to the large domain gap between synthetic and real-world data, we begin by finetuning on our aligned meshes from Stage 3.

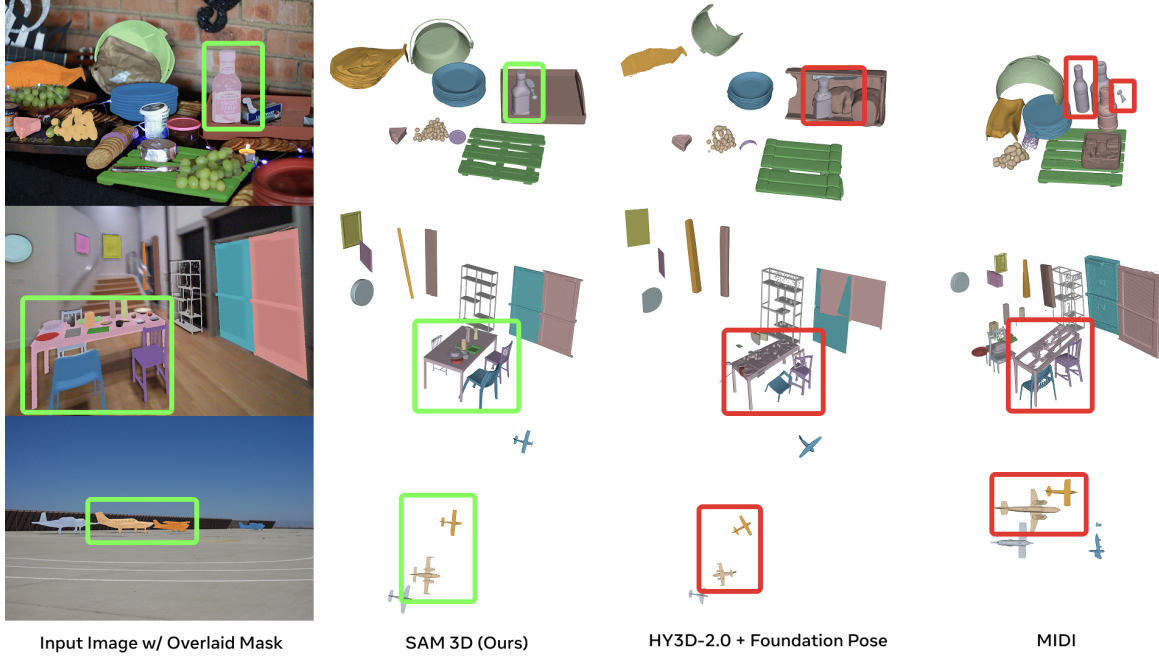


Figure 7 Qualitative comparison to competing scene reconstruction methods. We show SAM 3D’s full 3D scene reconstructions versus alternatives (Wen et al., 2024; Huang et al., 2025).

We begin SFT with the noisier non-expert labels (MITL-3DO), followed by the smaller, high-quality set from 3D artists (Art-3DO). The high quality Art-3DO data enhances model quality by aligning outputs with artists’ aesthetic preferences. We find this helps suppress common failures, *e.g.* floaters, bottomless meshes, and missing symmetry.

Preference optimization (alignment). After fine-tuning, the model can robustly generate shape and layout for diverse objects and real-world images. However, humans are sensitive to properties like symmetry, closure, etc. which are difficult to capture with generic objectives like flow matching. Thus, we follow SFT with a stage of direct preference optimization (DPO) (Rafailov et al., 2023), using D^+/D^- pairs from Stage 2 of our data engine. We found this off-policy data was effective at eliminating undesirable model outputs, even after SFT on Art-3DO. DPO training details are in Section C.3.

Distillation. Finally, to enable sub-second shape and layout from the Geometry model, we finish a short distillation stage to reduce the number of function evaluations (NFE) required during inference from $25 \rightarrow 4$. We adapt Frans et al. (2024) to our setting, and describe the details in Section C.4.

4 Experiments

Dataset. To comprehensively evaluate the model capability under real-world scenarios, we carefully build a new benchmark **SA-3DAO**, consisting of 1K 3D artist-created meshes created from natural images. We also include **ISO3D** from 3D Arena (Ebert, 2025) for quantitatively evaluating shape and texture, and Aria Digital Twin (**ADT**) (Pan et al., 2023) for layout. We further conduct human preference evaluation on two curated sets for both scene-level and object-level reconstruction. The **PrefSet** uses real-world images from MetaCLIP (Xu et al., 2024) and SA-1B (Kirillov et al., 2023), as well as a set based on LVIS (Gupta et al., 2019). Refer to Section D for details on evaluation sets.

Settings. We conduct experiments with a Geometry model that is trained to condition on pointmaps. For datasets where pointmaps are unavailable, we estimate them with Wang et al. (2025a). We found that shape and texture quality do not depend on whether the model is trained with pointmap conditioning (see Section E.5), but layout (translation/scale) evaluation in Table 3 requires ground-truth depth/pointmap as

| Model | SA-3DAO | | | | ISO3D Eval Set | |
|---------------|------------------------|---------------------|--------------------------|----------------------|---------------------|----------------------|
| | F1@0.01 (\uparrow) | vIoU (\uparrow) | Chamfer (\downarrow) | EMD (\downarrow) | ULIP (\uparrow) | Uni3D (\uparrow) |
| Trellis | 0.1475 | 0.1392 | 0.0902 | 0.2131 | 0.1473 | 0.3698 |
| HY3D-2.1 | 0.1399 | 0.1266 | 0.1126 | 0.2432 | 0.1293 | 0.3546 |
| HY3D-2.0 | 0.1574 | 0.1504 | 0.0866 | 0.2049 | 0.1484 | 0.3662 |
| Direct3D-S2 | 0.1513 | 0.1465 | 0.0962 | 0.2160 | 0.1405 | 0.3653 |
| TripoSG | 0.1533 | 0.1445 | 0.0844 | 0.2057 | 0.1529 | 0.3687 |
| Hi3DGen | 0.1629 | 0.1531 | 0.0937 | 0.2134 | 0.1419 | 0.3594 |
| SAM 3D | 0.2344 | 0.2311 | 0.0400 | 0.1211 | 0.1488 | 0.3707 |

Table 2 3D shape quantitative comparison to competing image-to-3D methods, including Trellis (Xiang et al., 2025), HY3D-2.1 (Hunyuan3D et al., 2025), HY3D-2.0 (Team, 2025), Direct3D-S2 (Wu et al., 2025), TripoSG (Li et al., 2025), Hi3DGen (Ye et al., 2025). SA-3DAO shows metrics that measure accuracy against GT geometry; ISO3D (Ebert, 2025) has no geometric GT and so we show perceptual similarities between 3D and input images (ULIP (Xue et al., 2023) and Uni3D (Zhou et al., 2023)). TripoSG uses a significantly higher mesh resolution, which is rewarded in perceptual metrics.

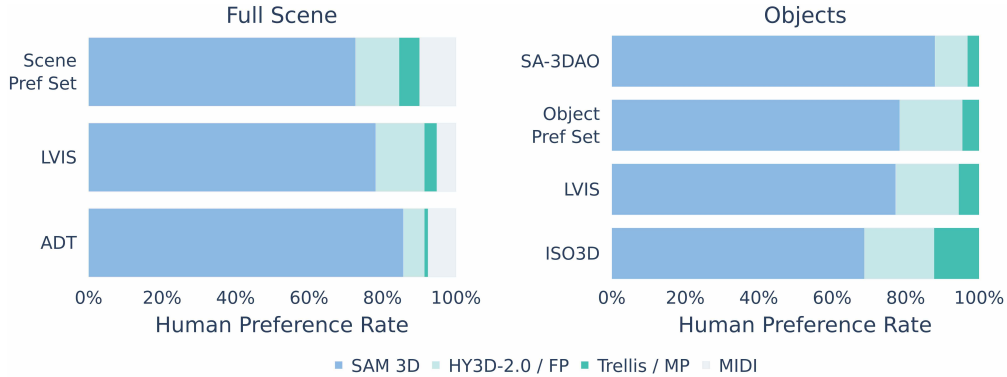


Figure 8 Preference comparison on scene-level and object-level reconstruction. Numbers indicate human preference rates. Objects comparisons are done on textured meshes. SAM 3D is significantly preferred over others on all fronts.

reference.

4.1 Comparison with SOTA

3D shape and texture. We evaluate single-object generation by comparing SAM 3D with prior state-of-the-art (SOTA) methods. In human preference studies, SAM 3D achieves an 5 : 1 head-to-head win rate on real images (see Figure 8). Table 2 presents quantitative evaluation on shape quality, where SAM 3D matches or exceeds previous SOTA performance on isolated object images (**ISO3D**), and significantly outperforms all baselines on challenging real-world inputs (**SA-3DAO**). Qualitative examples in Figure 6 further illustrate the model’s strong generalization under heavy occlusion. In Figure 9, we compare SAM 3D texture vs. other texture models, given SAM 3D shapes (SAM 3D’s improved shape actually benefits other methods in this eval). Annotators significantly prefer SAM 3D texture (details in Section E.2).

3D scene reconstruction. In preference tests on three evaluation sets, users prefer scene reconstructions from SAM 3D by 6 : 1 over prior SOTA (Figure 8). Figure 7 and Figure 20 in the appendix shows qualitative comparisons, while Table 3 shows quantitative metrics for object layout. On real-world data like **SA-3DAO** and **ADT**, the improvement is fairly stark and persists even when *pipeline* approaches use SAM 3D meshes. SAM 3D introduces a new real-world capability to generate shape and layout *jointly* (ADD-S @ 0.1 2% \rightarrow 77%), and a sample-then-optimize approach, as in the render-and-compare approaches (Labbé et al., 2022; Wen et al., 2024) can further improve performance (Section E.3). The strong results for layout and scene reconstruction demonstrate that SAM 3D can robustly handle both RGB-only inputs (e.g., **SA-3DAO**, **LVIS**, **Pref Set**) as well

| Generation | Model | SA-3DAO | | | | Aria Digital Twin | | | |
|------------|---------------------------|-----------------------|---------------------------|------------------------|----------------------------|-----------------------|---------------------------|------------------------|----------------------------|
| | | 3D IoU (\uparrow) | ICP-Rot. (\downarrow) | ADD-S (\downarrow) | ADD-S @ 0.1 (\uparrow) | 3D IoU (\uparrow) | ICP-Rot. (\downarrow) | ADD-S (\downarrow) | ADD-S @ 0.1 (\uparrow) |
| Pipeline | Trellis + Megapose | 0.2449 | 39.3866 | 0.5391 | 0.2831 | 0.2531 | 33.6114 | 0.4358 | 0.1971 |
| Pipeline | HY3D-2.0 + Megapose | 0.2518 | 33.8307 | 0.7146 | 0.3647 | 0.3794 | 29.0066 | 0.1457 | 0.4211 |
| Pipeline | HY3D-2.0 + FoundationPose | 0.2937 | 32.9444 | 0.3705 | 0.5396 | 0.3864 | 25.1435 | 0.1026 | 0.5992 |
| Pipeline | HY3D-2.1 + FoundationPose | 0.2395 | 39.8357 | 0.4186 | 0.4177 | 0.2795 | 33.1197 | 0.2135 | 0.4129 |
| Pipeline | SAM 3D + FoundationPose | 0.2837 | 32.9168 | 0.3848 | 0.5079 | 0.3661 | 18.9102 | 0.0930 | 0.6495 |
| Joint | MIDI | - | - | - | - | 0.0336 | 44.2353 | 2.5278 | 0.0175 |
| Joint | SAM 3D | 0.4254 | 20.7667 | 0.2661 | 0.7232 | 0.4970 | 15.2515 | 0.0765 | 0.7673 |

Table 3 3D layout quantitative comparison to competing layout prediction methods on SA-3DAO and Aria Digital Twin (Pan et al., 2023). SAM 3D significantly outperforms both *pipeline* approaches used in robotics (Labbé et al., 2022; Wen et al., 2024) and *joint* generative models (MIDI (Huang et al., 2025)). Most SA-3DAO scenes only contain one object so we do not show MIDI results that require multi-object alignment. The metrics measure bounding box overlap, rotation error, and chamfer-like distances normalized by object diameter.

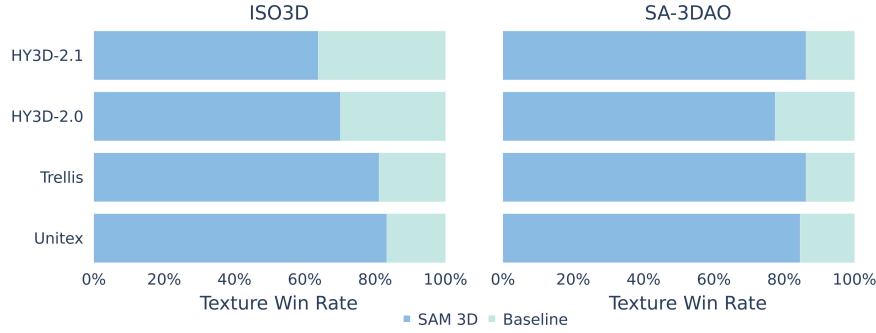


Figure 9 Preference comparison on texture. Since SAM 3D provides higher quality shape, we use the geometry results from SAM 3D and only perform texture generations for all methods. SAM 3D significantly outperforms others.

as provided pointmaps (e.g., ADT).

4.2 Analysis Studies

Post-training iterations steadily improve performance. We observed steady improvements as we ran the data engine for longer, with near-linear Elo scaling shown in the historical comparisons from Stage 2 of our data engine (Figure 10a). We found it important to scale all stages simultaneously. The cumulatively linear effect results from more data engine iterations, along with scaling up pretraining, mid-training, and adding new post-training stages. Figure 10b shows that iterating MITL-3DO data alone yields consistent improvements but with decreasing marginal impact.

Multi-stage training improves performance. SAM 3D’s real-world performance emerges through multi-stage training. Table 4 reveals near-monotonic 3D shape improvements as each training stage is added, validating the approach that leads to the final model (last row). In the appendix, Figure 17 shows similar results for texture and Table 7 shows the contribution of each individual real-world data stage, by knocking out the MITL-3DO, Art-3DO data, or DPO stages.

Other ablations. Please see the appendix for additional ablations on rotation representation (Section E.4), DPO (Section C.3), distillation (Section C.4), pointmaps (Section E.5), and scaling best-of- N in the data engine (Section A.7).

5 Related Work

3D reconstruction has been a longstanding challenge in computer vision. Classical methods include binocular stereopsis (Wheatstone, 1838), structure-from-motion (Hartley and Zisserman, 2003; Szeliski, 2022; Scharstein and Szeliski, 2002; Torresani et al., 2008; Tomasi and Kanade, 1992), and SLAM (Smith et al., 1990; Castellanos et al., 1999). Other strategies reconstruct by analysis (e.g., silhouettes (Esteban and Schmitt, 2004)) or by

| Training Stage | SA-3DAO | | | | Preference set |
|-------------------------|--------------------------|---------------------|--------------------------|----------------------|---------------------------|
| | F1 @ 0.01 (\uparrow) | vIoU (\uparrow) | Chamfer (\downarrow) | EMD (\downarrow) | Texture WR (\uparrow) |
| Pre-training (Iso-3DO) | 0.1349 | 0.1202 | 0.1036 | 0.2396 | - |
| + Mid-training (RP-3DO) | 0.1705 | 0.1683 | 0.0760 | 0.1821 | 60.7 |
| + SFT (MITL-3DO) | 0.2027 | 0.2025 | 0.0578 | 0.1510 | 66.9 |
| + DPO (MITL-3DO) | 0.2156 | 0.2156 | 0.0498 | 0.1367 | 66.4 |
| + SFT (Art-3DO) | 0.2331 | 0.2337 | 0.0445 | 0.1257 | - |
| + DPO (Art-3DO) | 0.2344 | 0.2311 | 0.0400 | 0.1211 | - |

Table 4 Cascading improvements from multi-stage training on 3D shape and texture. For texture, we report win rates (WR) between each row and the row *above* it.

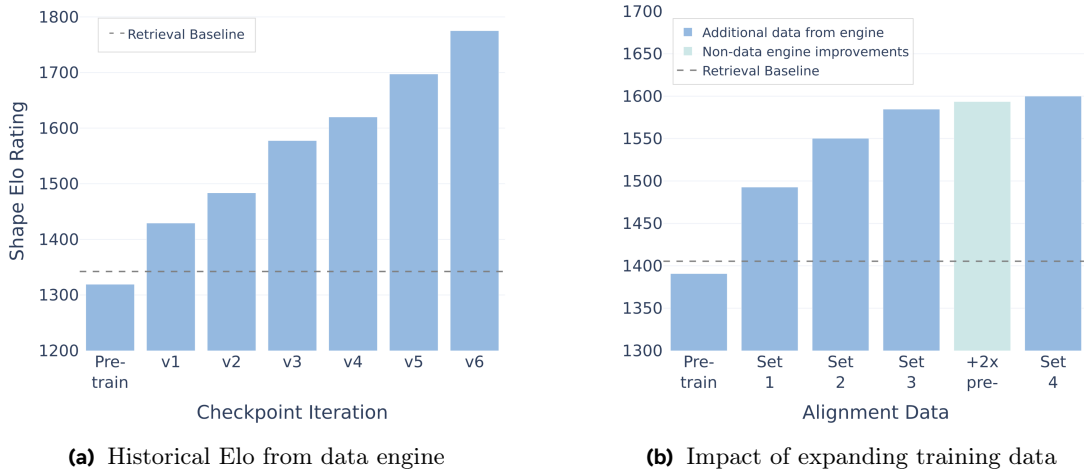


Figure 10 Data engine with additional iterations. The plots show Elo scores of different models; a 400 point Elo difference corresponds to 10:1 odds in a preference test. Models were (a) checkpoints around 3 weeks apart, indicating cumulative improvements as we scale and add different stages and (b) post-trained (SFT) using expanded training data.

synthesis via volume rendering (Kajiya and Von Herzen, 1984), using either implicit representations (Mildenhall et al., 2020) or explicit ones (Sitzmann et al., 2019; Liu et al., 2020). Supervised deep learning methods predict voxels (Xie et al., 2019; Wang et al., 2021), point clouds (Van Hoorick et al., 2022), or meshes (Worchel et al., 2022; Wen et al., 2019), or optimize implicit representations (Liu et al., 2024), e.g., signed distance functions (SDFs), often with high-quality output but requiring multiple views at inference. In contrast, we focus on the more restrictive setting of a single RGB image at test time.

Single-view 3D reconstruction is considerably more difficult. A large body of work trains models with direct 3D supervision, predicting meshes (Xu et al., 2019; Kulkarni et al., 2022), voxels (Girdhar et al., 2016; Wu et al., 2017), point clouds (Fan et al., 2017; Mescheder et al., 2019), or CAD-aligned geometry (Wang et al., 2018; Gkioxari et al., 2019). A recent line of work (Zhang et al., 2023; Xiang et al., 2025; Ren et al., 2024) supervises with VAE (Kingma and Welling, 2013) latent representations. However, these methods are typically evaluated on simplified synthetic single-object benchmarks such as ShapeNet (Chang et al., 2015), Pix3D (Sun et al., 2018) or Objaverse (Deitke et al., 2023).

Layout estimation. A large body of work estimates object poses from a single image, for object shapes (Labbé et al., 2022; Wen et al., 2024; Shi et al., 2025; Geng et al., 2025; Huang et al., 2025) or detections (Brazil et al., 2023), but is typically restricted to tabletop robotics, streets, or indoor scenes where objects rest on a supporting surface. In contrast, our approach estimates both pose for a broad range of object types across diverse scenes.

3D datasets. Sourcing 3D annotations is challenging: the modality itself is complex, and the specialized tools required are hard to master. Anecdotally, modeling a 3D mesh from a reference image can take an

experienced artist hours (Section D). Instead, existing 3D datasets (*e.g.*, ShapeNet (Chang et al., 2015), Objaverse-XL (Deitke et al., 2023)) primarily consist of single synthetic objects; without paired real-world images, models can only learn from rendered views. In the real-world domain, existing datasets are small and mostly indoors (Reizenstein et al., 2021; Khanna et al., 2024; Fu et al., 2021; Szot et al., 2021; Pan et al., 2023). Models trained on such constrained data struggle to generalize.

Post-training. While post-training began with a single supervised finetuning stage (Girshick et al., 2013; Wei et al., 2021), strong pretraining (Brown et al., 2020) made alignment much more data efficient (Hernandez et al., 2021), enabling iterative preference-based alignment like RLHF (Ouyang et al., 2022) and online DPO (Tang et al., 2024; Rafailov et al., 2023). When post-training must provide a strong steer, self-training methods offer denser supervision—leveraging the model itself to generate increasingly high-quality demonstrations, rather than relying solely on preference signals (Gulcehre et al., 2023; Anthony et al., 2017; Dong et al., 2023; Yuan et al., 2023). SAM 3D employs self-training to bridge the synthetic→real domain gap and break the data barrier for 3D perception; most closely resembling RAFT (Dong et al., 2023), but also incorporating preference tuning.

Multi-stage pretraining. Modern pretraining increasingly employs multiple training stages. Early work on curriculum learning (Bengio et al., 2009) provided a basis for staged data mixing in pretraining, with higher-quality data coming later (Grattafiori et al., 2024; OLMo et al., 2025). Li et al. (2023b); Abdin et al. (2024) show that mixing synthetic/web curricula can achieve strong performance at smaller scales. Increasingly, additional mid-training stages are used for capability injection, such as context extension (Grattafiori et al., 2024) or coding (Rozière et al., 2024), and recent work finds that mid-training significantly improves post-training effectiveness (Lambert, 2025; Wang et al., 2025b). SAM 3D introduces synthetic pretraining and mid-training that can generalize for 3D.

6 Conclusion

We share SAM 3D: a new foundation model for full reconstruction of 3D shape, texture, and layout of objects from natural images. SAM 3D’s robustness on in-the-wild images, made possible by an innovative data engine and modern training recipe, represents a step change for 3D and an advance towards real-world 3D perception. With the release of our model, we expect SAM 3D to unlock new capabilities across diverse applications, such as robotics, AR/VR, gaming, film, and interactive media.

Acknowledgements

We thank the following individuals for their contributions to this work:

For their contributions to SAM Playground Engineering we thank: Robbie Adkins, Rene de la Fuente, Facundo Figueroa, Alex He, Dex Honsa, Alex Lende, Jonny Li, Peter Park, Don Pinkus, Roman Radle, Phillip Thomas, and Meng Wang. We thank our excellent XFN team for leadership and support: Kris Kitani, Vivian Lee, Sasha Mitts, George Orlin, Nikhila Ravi, and Andrew Westbury. Thanks to Helen Klein, Mallika Malhotra, and Azita Shokrpour for support with Legal, Privacy, and Integrity. We thank Michelle Chan, Kei Koyama, William Ngan, Yael Yungster for all the design support throughout the project. Thanks to Arpit Kalla for work on model efficiency. We thank Faye Ma and Kehan Lyu for data engineering support and tooling, and Emmanuel Hernandez, Robert Kuo for pipeline development. We thank Nan Yang for support with egocentric video data efforts. Thanks to our two interns Cem Gokmen, Jasmine Shone for their work on 3D and Lea Wilken for feedback on the manuscript. Thanks to our fantastic data operations team: Paris Baptiste, Karen Bergan, Kai Brown, Ida Cheng, Khadijat Durojaiye, Patrick Edwards, Daniella Factor, Eva Galper, Leonna Jones, Zayida Suber, Tatum Turner, Joseph Walker, and Claudette Ward.

References

- Marah Abdin et al. Phi-3 technical report: A highly capable language model locally on your phone, 2024. <https://arxiv.org/abs/2404.14219>.
- Thomas Anthony, Zheng Tian, and David Barber. Thinking fast and slow with deep learning and tree search. In *Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS'17*, page 5366–5376, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.
- Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning, ICML '09*, page 41–48, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605585161. doi: 10.1145/1553374.1553380. <https://doi.org/10.1145/1553374.1553380>.
- Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In *European Conference on Computer Vision (ECCV)*, 2014.
- Garrrick Brazil, Abhinav Kumar, Julian Straub, Nikhila Ravi, Justin Johnson, and Georgia Gkioxari. Omni3d: A large benchmark and model for 3d object detection in the wild. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13154–13164, 2023.
- Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.
- Thomas J Cashman and Andrew W Fitzgibbon. What shape are dolphins? building 3d morphable models from 2d images. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):232–244, 2012.
- Jose A Castellanos, José MM Montiel, José Neira, and Juan D Tardós. The SPmap: A probabilistic framework for simultaneous localization and map building. *IEEE Transactions on robotics and Automation*, 1999.
- 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.
- Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, et al. The epic-kitchens dataset: Collection, challenges and baselines. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(11):4125–4141, 2020.
- Pieter-Tjerk de Boer, Dirk P. Kroese, Shie Mannor, and Reuven Y. Rubinstein. A tutorial on the cross-entropy method. *Annals of operations research*, 134(1):19–67, January 2005. ISSN 0254-5330. doi: 10.1007/s10479-005-5724-z.
- Paul E Debevec, Camillo J Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs: A hybrid geometry-and image-based approach. In *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, 2023.
- Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, et al. Objaverse-xl: A universe of 10m+ 3d objects. *Advances in Neural Information Processing Systems*, 36:35799–35813, 2023.
- Yuxuan Deng, Yujia Zhu, Jiahui Chen, Yuan Wang, Yifei Li, Haotian Li, Junnan Li, Jinsheng Zhang, Wenhui Liu, Yuzheng Zhang, et al. Emerging properties in unified multimodal pretraining. *arXiv preprint arXiv:2505.14683*, 2025.
- Kristin Diehl and Cait Poynor. Great expectations?! assortment size, expectations, and satisfaction. *Journal of Marketing Research*, 47(2):312–322, 2010. doi: 10.1509/jmkr.47.2.312. <https://doi.org/10.1509/jmkr.47.2.312>.
- Hanze Dong, Wei Xiong, Deepanshu Goyal, Yihan Zhang, Winnie Chow, Rui Pan, Shizhe Diao, Jipeng Zhang, KaShun SHUM, and Tong Zhang. RAFT: Reward ranked finetuning for generative foundation model alignment. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. <https://openreview.net/forum?id=m7p5O7zblY>.
- Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. FlowNet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2758–2766, 2015.
- Dylan Ebert. 3d arena: An open platform for generative 3d evaluation. *arXiv preprint arXiv:2506.18787*, 2025.

- Carlos Hernández Esteban and Francis Schmitt. Silhouette and stereo fusion for 3D object modeling. *Computer Vision and Image Understanding*, 2004.
- Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3D object reconstruction from a single image. In *CVPR*, 2017.
- Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models. *arXiv preprint arXiv:2410.12557*, 2024.
- Huan Fu, Bowen Cai, Lin Gao, Ling-Xiao Zhang, Jiaming Wang, Cao Li, Qixun Zeng, Chengyue Sun, Rongfei Jia, Binqiang Zhao, et al. 3d-front: 3d furnished rooms with layouts and semantics. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10933–10942, 2021.
- Zheng Geng, Nan Wang, Shaocong Xu, Chongjie Ye, Bohan Li, Zhaoxi Chen, Sida Peng, and Hao Zhao. One view, many worlds: Single-image to 3d object meets generative domain randomization for one-shot 6d pose estimation. *arXiv preprint arXiv:2509.07978*, 2025.
- Rohit Girdhar, David F Fouhey, Mikel Rodriguez, and Abhinav Gupta. Learning a predictable and generative vector representation for objects. In *ECCV*, 2016.
- Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *CoRR*, abs/1311.2524, 2013. <http://arxiv.org/abs/1311.2524>.
- Georgia Gkioxari, Jitendra Malik, and Justin Johnson. Mesh r-cnn. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9785–9795, 2019.
- Aaron Grattafiori et al. The llama 3 herd of models, 2024. <https://arxiv.org/abs/2407.21783>.
- Kristen Grauman, Andrew Westbury, et al. Ego4d: Around the world in 3,000 hours of egocentric video. *International Journal of Computer Vision (IJCV)*, 2022.
- Kristen Grauman et al. Ego-exo4d: Understanding skilled human activity from first- and third-person perspectives. *arXiv preprint arXiv:2401.10889*, 2024.
- Caglar Gulcehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, Wolfgang Macherey, Arnaud Doucet, Orhan Firat, and Nando de Freitas. Reinforced self-training (rest) for language modeling, 2023. <https://arxiv.org/abs/2308.08998>.
- Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5356–5364, 2019.
- Richard Hartley and Andrew Zisserman. Multiple view geometry in computer vision, 2003.
- Danny Hernandez, Jared Kaplan, Tom Henighan, and Sam McCandlish. Scaling laws for transfer. *arXiv preprint arXiv:2102.01293*, 2021.
- Zehuan Huang, Yuan-Chen Guo, Xingqiao An, Yunhan Yang, Yangguang Li, Zi-Xin Zou, Ding Liang, Xihui Liu, Yan-Pei Cao, and Lu Sheng. Midi: Multi-instance diffusion for single image to 3d scene generation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 23646–23657, 2025.
- Team Hunyuan3D, Shuhui Yang, Mingxin Yang, Yifei Feng, Xin Huang, Sheng Zhang, Zebin He, Di Luo, Haolin Liu, Yunfei Zhao, et al. Hunyuan3d 2.1: From images to high-fidelity 3d assets with production-ready pbr material. *arXiv preprint arXiv:2506.15442*, 2025.
- James T Kajiya and Brian P Von Herzen. Ray tracing volume densities. *SIGGRAPH*, 1984.
- Abhishek Kar, Shubham Tulsiani, Joao Carreira, and Jitendra Malik. Category-specific object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1966–1974, 2015.
- Mukul Khanna, Yongsan Mao, Hanxiao Jiang, Sanjay Haresh, Brennan Shacklett, Dhruv Batra, Alexander Clegg, Eric Undersander, Angel X Chang, and Manolis Savva. 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*, pages 16384–16393, 2024.
- Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

- Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4015–4026, 2023.
- Jan J Koenderink, Andrea J Van Doorn, and Astrid ML Kappers. Surface perception in pictures. *Perception & psychophysics*, 52(5):487–496, 1992.
- Nilesh Kulkarni, Justin Johnson, and David F. Fouhey. What’s behind the couch? directed ray distance functions for 3D scene reconstruction. In *ECCV*, 2022.
- Yann Labbé, Lucas Manuelli, Arsalan Mousavian, Stephen Tyree, Stan Birchfield, Jonathan Tremblay, Justin Carpentier, Mathieu Aubry, Dieter Fox, and Josef Sivic. Megapose: 6d pose estimation of novel objects via render & compare. *arXiv preprint arXiv:2212.06870*, 2022.
- Nathan Lambert. *Reinforcement Learning from Human Feedback*. Online, 2025. <https://rlhfbbook.com>.
- Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, Noah A. Smith, and Hannaneh Hajishirzi. Rewardbench: Evaluating reward models for language modeling, 2024. <https://arxiv.org/abs/2403.13787>.
- Yanguang Li, Zi-Xin Zou, Zexiang Liu, Dehu Wang, Yuan Liang, Zhipeng Yu, Xingchao Liu, Yuan-Chen Guo, Ding Liang, Wanli Ouyang, et al. Triposg: High-fidelity 3d shape synthesis using large-scale rectified flow models. *arXiv preprint arXiv:2502.06608*, 2025.
- Yanhao Li, Haoqi Fan, Rohit Girdhar, and Alexander Kirillov. Segment anything in videos. *arXiv preprint arXiv:2305.06500*, 2023a.
- Yuanzhi Li, Sébastien Bubeck, Ronen Eldan, Allie Del Giorno, Suriya Gunasekar, and Yin Tat Lee. Textbooks are all you need ii: phi-1.5 technical report, 2023b. <https://arxiv.org/abs/2309.05463>.
- Weixin Liang, LILI YU, Liang Luo, Srini Iyer, Ning Dong, Chunting Zhou, Gargi Ghosh, Mike Lewis, Wen tau Yih, Luke Zettlemoyer, and Xi Victoria Lin. Mixture-of-transformers: A sparse and scalable architecture for multi-modal foundation models. *Transactions on Machine Learning Research*, 2025a. ISSN 2835-8856. <https://openreview.net/forum?id=Nu6N69i8SB>.
- Yixun Liang, Kunming Luo, Xiao Chen, Rui Chen, Hongyu Yan, Weiyu Li, Jiarui Liu, and Ping Tan. Unitex: Universal high fidelity generative texturing for 3d shapes. *arXiv preprint arXiv:2505.23253*, 2025b.
- Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *NeurIPS*, 2020.
- Minghua Liu, Ruoxi Shi, Linghao Chen, Zhuoyang Zhang, Chao Xu, Xinyue Wei, Hansheng Chen, Chong Zeng, Jiayuan Gu, and Hao Su. One-2-3-45++: Fast single image to 3d objects with consistent multi-view generation and 3d diffusion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10072–10083, 2024.
- Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.
- Zhaoyang Lv, Nicholas Charron, Pierre Moulon, Alexander Gamino, Cheng Peng, Chris Sweeney, Edward Miller, Huixuan Tang, Jeff Meissner, Jing Dong, et al. Aria everyday activities dataset. *arXiv preprint arXiv:2402.13349*, 2024.
- Lingni Ma, Yuting Ye, Fangzhou Hong, Vladimir Guzov, Yifeng Jiang, Rowan Postyeni, Luis Pesqueira, Alexander Gamino, Vijay Baiyya, Hyo Jin Kim, Kevin Bailey, David Soriano Fosas, C. Karen Liu, Ziwei Liu, Jakob Engel, Renzo De Nardi, and Richard Newcombe. Nymeria: A massive collection of multimodal egocentric daily motion in the wild. In *the 18th European Conference on Computer Vision (ECCV)*, 2024. <https://arxiv.org/abs/2406.09905>.
- Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4040–4048, 2016.
- Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3D reconstruction in function space. In *CVPR*, 2019.
- Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

- Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. Large language models: A survey, 2025. <https://arxiv.org/abs/2402.06196>.
- Kaixiang Mo, Yuxin Shi, Weiwei Weng, Zhiqiang Zhou, Shuman Liu, Haibo Zhang, and Anxiang Zeng. Mid-training of large language models: A survey, 2025. <https://arxiv.org/abs/2510.06826>.
- NVIDIA, :, Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi "Jim" Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu, Edith Llonet, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed, You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzen Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng, and Yuke Zhu. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv: 2503.14734*, 2025.
- Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, Nathan Lambert, Dustin Schwenk, Oyvind Tafjord, Taira Anderson, David Atkinson, Faeze Brahman, Christopher Clark, Pradeep Dasigi, Nouha Dziri, Allyson Ettinger, Michal Guerquin, David Heineman, Hamish Ivison, Pang Wei Koh, Jiacheng Liu, Saumya Malik, William Merrill, Lester James V. Miranda, Jacob Morrison, Tyler Murray, Crystal Nam, Jake Poznanski, Valentina Pyatkin, Aman Rangapur, Michael Schmitz, Sam Skjonsberg, David Wadden, Christopher Wilhelm, Michael Wilson, Luke Zettlemoyer, Ali Farhadi, Noah A. Smith, and Hannaneh Hajishirzi. 2 olmo 2 furious, 2025. <https://arxiv.org/abs/2501.00656>.
- Abby O'Neill et al. Open x-embodiment: Robotic learning datasets and rt-x models. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6892–6903. IEEE, 2024.
- Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.
- Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. *arXiv preprint arXiv: 2203.02155*, 2022.
- Xiaqing Pan, Nicholas Charron, Yongqian Yang, Scott Peters, Thomas Whelan, Chen Kong, Omkar Parkhi, Richard Newcombe, and Yuheng Carl Ren. Aria digital twin: A new benchmark dataset for egocentric 3d machine perception. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 20133–20143, 2023.
- William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4195–4205, 2023.
- Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.
- Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotny. Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10901–10911, 2021.
- Xuanchi Ren, Jiahui Huang, Xiaohui Zeng, Ken Museth, Sanja Fidler, and Francis Williams. Xcube: Large-scale 3d generative modeling using sparse voxel hierarchies. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4209–4219, 2024.
- Lawrence G Roberts. *Machine perception of three-dimensional solids*. PhD thesis, Massachusetts Institute of Technology, 1963.
- Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, Ivan Evtimov, Joanna Bitton, Manish Bhatt, Cristian Canton Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre Défossez, Jade Copet, Faisal Azhar, Hugo Touvron, Louis Martin, Nicolas Usunier, Thomas Scialom, and Gabriel Synnaeve. Code llama: Open foundation models for code, 2024. <https://arxiv.org/abs/2308.12950>.
- Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *IJCV*, 2002.
- Wubin Shi, Shaoyan Gai, Feipeng Da, and Zeyu Cai. Sampose: Generalizable model-free 6d object pose estimation via single-view prompt. *IEEE Robotics and Automation Letters*, 2025.

- Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhofer. DeepVoxels: Learning persistent 3D feature embeddings. In *CVPR*, 2019.
- Randall Smith, Matthew Self, and Peter Cheeseman. Estimating uncertain spatial relationships in robotics. In *Autonomous robot vehicles*, 1990.
- Julian Straub, Daniel DeTone, Tianwei Shen, Nan Yang, Chris Sweeney, and Richard Newcombe. Efm3d: A benchmark for measuring progress towards 3d egocentric foundation models. In *arXiv preprint arXiv:2406.10224*, 2024. <https://arxiv.org/abs/2406.10224>.
- Xingyuan Sun, Jiajun Wu, Xiuming Zhang, Zhoutong Zhang, Chengkai Zhang, Tianfan Xue, Joshua B Tenenbaum, and William T Freeman. Pix3d: Dataset and methods for single-image 3d shape modeling. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2974–2983, 2018.
- Richard Szeliski. *Computer vision: algorithms and applications*. Springer Nature, 2022.
- Andrew Szot, Alexander Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Singh Chaplot, Oleksandr Maksymets, et al. Habitat 2.0: Training home assistants to rearrange their habitat. *Advances in neural information processing systems*, 34:251–266, 2021.
- Yunhao Tang, Daniel Zhaohan Guo, Zeyu Zheng, Daniele Calandriello, Yuan Cao, Eugene Tarassov, Rémi Munos, Bernardo Ávila Pires, Michal Valko, Yong Cheng, and Will Dabney. Understanding the performance gap between online and offline alignment algorithms. *arXiv preprint arXiv: 2405.08448*, 2024.
- Tencent Hunyuan3D Team. Hunyuan3d 2.0: Scaling diffusion models for high resolution textured 3d assets generation, 2025.
- Carlo Tomasi and Takeo Kanade. Shape and motion from image streams under orthography: a factorization method. *IJCV*, 1992.
- Lorenzo Torresani, Aaron Hertzmann, and Chris Bregler. Nonrigid structure-from-motion: Estimating shape and motion with hierarchical priors. *PAMI*, 2008.
- Basile Van Hoorick, Purva Tendulkar, Dídac Surís, Dennis Park, Simon Stent, and Carl Vondrick. Revealing occlusions with 4d neural fields. In *CVPR*, 2022.
- Bram Wallace, Meihua Dang, Rafael Rafailov, Linqi Zhou, Aaron Lou, Senthil Purushwalkam, Stefano Ermon, Caiming Xiong, Shafiq Joty, and Nikhil Naik. Diffusion model alignment using direct preference optimization. In *CVPR*, 2024.
- Dan Wang, Xinrui Cui, Xun Chen, Zhengxia Zou, Tianyang Shi, Septimiu Salcudean, Z Jane Wang, and Rabab Ward. Multi-view 3d reconstruction with transformers. In *ICCV*, 2021.
- Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2Mesh: Generating 3D mesh models from single RGB images. In *ECCV*, 2018.
- Ruicheng Wang, Sicheng Xu, Cassie Dai, Jianfeng Xiang, Yu Deng, Xin Tong, and Jiaolong Yang. Moge: Unlocking accurate monocular geometry estimation for open-domain images with optimal training supervision. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 5261–5271, 2025a.
- Zengzhi Wang, Fan Zhou, Xuefeng Li, and Pengfei Liu. Octothinker: Mid-training incentivizes reinforcement learning scaling, 2025b. <https://arxiv.org/abs/2506.20512>.
- Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. Finetuned language models are zero-shot learners. *CoRR*, abs/2109.01652, 2021. <https://arxiv.org/abs/2109.01652>.
- Bowen Wen, Wei Yang, Jan Kautz, and Stan Birchfield. Foundationpose: Unified 6d pose estimation and tracking of novel objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17868–17879, 2024.
- Chao Wen, Yinda Zhang, Zhuwen Li, and Yanwei Fu. Pixel2Mesh++: Multi-view 3D mesh generation via deformation. In *ICCV*, 2019.
- Charles Wheatstone. Contributions to the physiology of vision.—part the first. on some remarkable, and hitherto unobserved, phenomena of binocular vision. *Philosophical transactions of the Royal Society of London*, 1838.
- Markus Worchel, Rodrigo Diaz, Weiwen Hu, Oliver Schreer, Ingo Feldmann, and Peter Eisert. Multi-view mesh reconstruction with neural deferred shading. In *CVPR*, 2022.

- Jiajun Wu, Yifan Wang, Tianfan Xue, Xingyuan Sun, Bill Freeman, and Josh Tenenbaum. MarrNet: 3D shape reconstruction via 2.5D sketches. *NeurIPS*, 2017.
- Shuang Wu, Youtian Lin, Feihu Zhang, Yifei Zeng, Jingxi Xu, Philip Torr, Xun Cao, and Yao Yao. Direct3d: Scalable image-to-3d generation via 3d latent diffusion transformer. *Advances in Neural Information Processing Systems*, 37: 121859–121881, 2024.
- Shuang Wu, Youtian Lin, Feihu Zhang, Yifei Zeng, Yikang Yang, Yajie Bao, Jiachen Qian, Siyu Zhu, Philip Torr, Xun Cao, and Yao Yao. Direct3d-s2: Gigascale 3d generation made easy with spatial sparse attention. *arXiv preprint arXiv:2505.17412*, 2025.
- Jianfeng Xiang, Zelong Lv, Sicheng Xu, Yu Deng, Ruicheng Wang, Bowen Zhang, Dong Chen, Xin Tong, and Jiaolong Yang. Structured 3d latents for scalable and versatile 3d generation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 21469–21480, 2025.
- Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. In *Robotics: Science and Systems (RSS)*, 2018.
- Haozhe Xie, Hongxun Yao, Xiaoshuai Sun, Shangchen Zhou, and Shengping Zhang. Pix2vox: Context-aware 3D reconstruction from single and multi-view images. In *ICCV*, 2019.
- Hu Xu, Nikhila Goyal, Mitchell Wortsman, Gabriel Ilharco, Ozan Sener, Aniruddha Kembhavi, Ali Farhadi, and Rohit Girdhar. Metacclip: How to make clip efficiently. *arXiv preprint arXiv:2404.07143*, 2024.
- Qiangeng Xu, Weiyue Wang, Duygu Ceylan, Radomir Mech, and Ulrich Neumann. DISN: Deep implicit surface network for high-quality single-view 3D reconstruction. *NeurIPS*, 2019.
- Le Xue, Mingfei Gao, Chen Xing, Roberto Martín-Martín, Jiajun Wu, Caiming Xiong, Ran Xu, Juan Carlos Niebles, and Silvio Savarese. Ulip: Learning a unified representation of language, images, and point clouds for 3d understanding. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1179–1189, 2023.
- Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything: Unleashing the power of large-scale unlabeled data. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10371–10381, 2024a.
- Xianghui Yang, Huiwen Shi, Bowen Zhang, Fan Yang, Jiacheng Wang, Hongxu Zhao, Xinhai Liu, Xinzhou Wang, Qingxiang Lin, Jiaao Yu, et al. Hunyuan3d 1.0: A unified framework for text-to-3d and image-to-3d generation. *arXiv preprint arXiv:2411.02293*, 2024b.
- Chongjie Ye, Yushuang Wu, Ziteng Lu, Jiahao Chang, Xiaoyang Guo, Jiaqing Zhou, Hao Zhao, and Xiaoguang Han. Hi3dgen: High-fidelity 3d geometry generation from images via normal bridging. *arXiv preprint arXiv:2503.22236*, 3:2, 2025.
- Fisher Yu, Haofeng Chen, Xin Wang, Wei Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving video database with scalable annotation tooling. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.
- Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting Dong, Keming Lu, Chuanqi Tan, Chang Zhou, and Jingren Zhou. Scaling relationship on learning mathematical reasoning with large language models, 2023. <https://arxiv.org/abs/2308.01825>.
- Biao Zhang, Jiapeng Tang, Matthias Niessner, and Peter Wonka. 3dshape2vecset: A 3d shape representation for neural fields and generative diffusion models. *ACM Transactions On Graphics (TOG)*, 42(4):1–16, 2023.
- Junsheng Zhou, Jinsheng Wang, Baorui Ma, Yu-Shen Liu, Tiejun Huang, and Xinlong Wang. Uni3d: Exploring unified 3d representation at scale. *arXiv preprint arXiv:2310.06773*, 2023.
- Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5745–5753, 2019.

Appendix

Outline

The appendix provides additional context to the main paper; it contains additional details about the method and the implementation in SAM 3D, as well as ablations.

The structure of the appendix is as follows:

- (i) **Data Engine details:** A more detailed description the data collection used in the *collection step* in [Section 3.2.1](#).
- (ii) **Pretraining and Mid-Training Data:** How we collected and filtered the data used for pretraining and mid-training the Geometry and Texture & Refinement models
- (iii) **Training details:** Architectural details about MoT and the VAEs. Definitions used for objectives used in each stage. Details on the Geometry and Texture & Refinement models.
- (iv) **Evaluations:** Introducing the details of the new **SA-3DAO** benchmark, and evaluation protocols of preference tests and quantitative metrics
- (v) **Additional experiments and qualitative examples:** Providing additional analysis and insights into the model’s performance.
- (vi) **Limitations;** An analysis of common failure modes, and future work

A Data Annotation Engine Details

A.1 Stage 1: Image and Object Candidate Sourcing

Image sources. To promote generalization across diverse real-world scenes, we expanded our domain coverage by sourcing images from multiple datasets. These include large-scale web-sourced imagery (SA-1B (Kirillov et al., 2023), MetaCLIP (Xu et al., 2024)), video data capturing everyday environments (SA-VI (Li et al., 2023a)), egocentric video datasets (Ego4D (Grauman et al., 2022), Ego-Exo4D (Grauman et al., 2024)), AEA (Lv et al., 2024), AEO (Straub et al., 2024), Nymeria (Ma et al., 2024)), and domain-specific collections such as food (Food Recognition (Bossard et al., 2014)) and driving scenes (BDD100k (Yu et al., 2020)).

We first filter out images with low resolution, severe blurriness, low contrast, or noticeable artifacts to ensure high-quality visual inputs that are representative of real-world scenarios. Next, we employ visual-language models for object recognition to generate object-level annotations for each image. Images containing only uninformative backgrounds (*e.g.*, ground, sky, ocean) without salient 3D objects are subsequently removed from the dataset.

For each object description, we employ a referral segmentation model to visually ground the object, followed by human annotator verification or refinement of object masks. We discard low-quality masks, masks covering multiple objects, or partial masks that do not capture a distinct object part. This ensures that each retained mask corresponds to a clearly indexable single object instance with sufficient granularity.

2D object selection In addition to the objects manually selected and masked by annotators, we also supplement our object mask inputs with segmentation masks sampled from pre-existing datasets. Besides saving annotation time, this strategy gives us more fine-grained control over the object distribution of the input masks, as object distributions are difficult to enforce on a per-image or per-annotator basis. To ensure a broad coverage of object categories, we adopt two complementary sampling strategies. First, we construct a 3D-oriented taxonomy by carefully merging and modifying the LVIS (Gupta et al., 2019) 1,200 object categories, emphasizing representations of 3D geometry. For example, different dog breeds are grouped together due to their similar underlying 3D structures, regardless of color, texture, or size. Second, we incorporate human annotator input to identify additional salient objects that may fall outside the taxonomy or are difficult to describe using text alone.

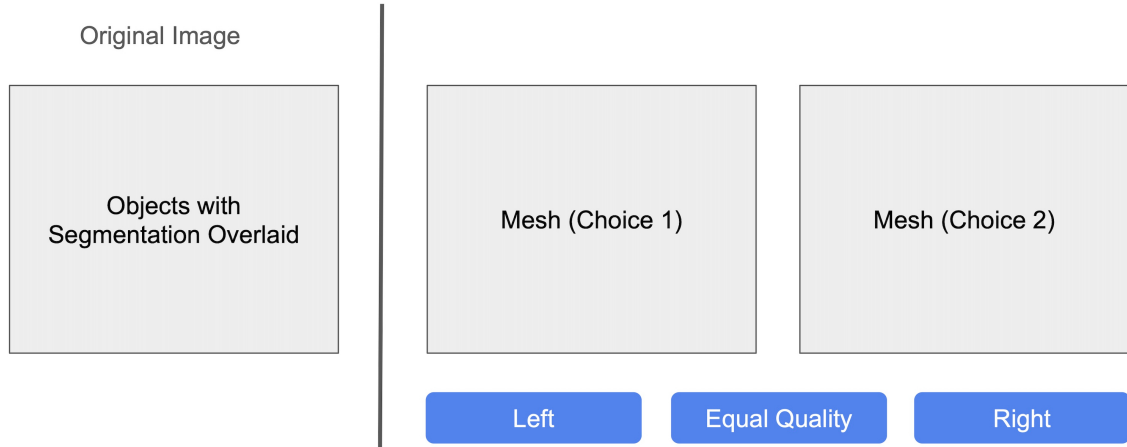


Figure 12 Stage 2 UI sketch. Annotators can only choose between options; they cannot directly edit the meshes or textures.

occlusion, but human recognition can still identify the object.

- **Image-to-3D generation:** Image-to-3D methods, including our own SAM 3D checkpoint, generate 3D objects in the form of point clouds or meshes, conditioned on the image input. When successful, this tends to produce examples that go beyond semantic matches and better respect the object’s physical appearance in the image. However, lack of robustness to occlusion or clutter can negatively impact the results.

3D texture model suite. For texture generation, we utilize image-to-3D models, multi-view texture generation models, and our own SAM 3D checkpoints. All texture candidates are generated using shapes produced by the SAM 3D Geometry model, ensuring that texture models have the best chance of success, even in cases of heavy occlusion.

Stage 2 Selection procedure. The annotators select the best-of- N candidates by making a series of pairwise comparisons (see Figure 12). For each object, the annotator is initially presented with two candidates to compare and is asked to pick from the following three choices: “Left” (is better), “Right” (is better), or “Equal Quality”. Because the options are in 3D, we by default automatically rotate the objects on a turntable, but annotators are free to rotate the objects as they wish, or zoom the camera. After making a selection, the non-selected option is replaced by a new candidate; if “Equal Quality”, we randomly choose which candidate to keep. The selection procedure continues until all candidates have been shown. We randomize the order in which candidates are presented to the annotator, to prevent biases due to order from affecting the selection process.

After the best candidate is identified in the selection process, annotators are asked to rate the mesh against a predefined quality bar α . Examples meeting the bar will become candidates to enter Stage 3 for alignment, while examples under the bar will become negative examples for preference alignment or considered as candidates for manual mesh generation in Stage 2.5.

A.3 Stage 2.5: 3D Artist Mesh Details

When the 3D model-in-the-loop suite fails to generate an acceptable mesh for a particular sample, the aforementioned preference-based annotation approach is unable to provide the data needed to improve the model for such objects. To overcome this data distribution blind spot, we work with a team of 3D artists to build meshes for such hard meshes. Given the high cost of specialized 3D artists, we seek to maximize their value by ensuring each object sent to the 3D artists represents a genuine failure case that cannot be resolved by the data engine alone. To maximize the value of this investment, we develop a refined labeling framework that categorizes failures into common types: *e.g.*, complex geometry, occlusion, transparency, and small object size. We balance sampling across these categories. In addition, we employ clustering techniques over images,

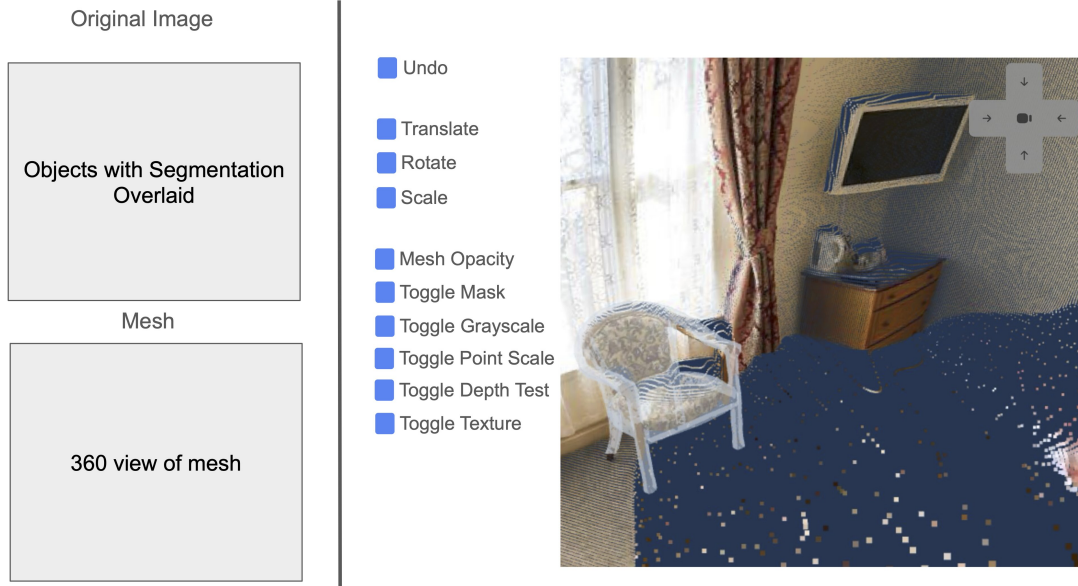


Figure 13 Stage 3 UI sketch. The UI supports annotators in directly placing the object in the 2.5D pointcloud.

3D latents, and object semantics to deduplicate candidates, ensuring that one or a few representative samples per group suffices for effective coverage in data sampling.

Additional details on the data collection process for meshes created by 3D artists can be found in [Section D.1](#), which employed a similar mesh creation process by the artists, but with more intentional curation of inputs.

A.4 Stage 3: 3D Mesh Alignment

We collect object pose annotations by aligning meshes from prior stages to a scene point cloud derived from the input image. To make this accessible to generalist annotators, we designed and implemented an annotation tool which allows the annotator to manipulate 3D meshes to align to a 2.5D point cloud pre-computed by an off-the-shelf depth estimator. Annotators can use either keyboard or mouse to rotate, translate, and scale the meshes so that the mesh is accurately anchored to the 2.5D point cloud. We also provide additional functions including (a) mesh visibility toggle, (b) target indicator toggle, (c) point cloud size adjustment, (d) control visibility toggle, (e) undo, (f) camera view reset and pre-defined view, and (g) mesh IOU indicator as shown in [Figure 13](#).

A.5 Annotation Statistics

- Stage 1: Annotators on average spend 10 seconds to segment a single interesting object. We utilize SAM ([Kirillov et al., 2023](#)) as a tool to assist in segmentation.
- Stage 2: Annotators on average spend 80 seconds to select the best candidate shape/texture from 6-10 candidate meshes from variable sources.
- Stage 3: Annotators on average spend 150 seconds to anchor and orient the matched 3D shape to the 2.5D point cloud.
- Over the lifetime of the project (including development), our MITL data engine yields 3.14 million trainable shapes, 1.23 million samples of layout data, 100K trainable textures, and over 7 million pairwise preferences.

Algorithm 1 SAM 3D Basic Alignment (Texture, Shape)

Require: Base model π_0 , quality threshold curriculum α_k , ensemble size N

Ensure: Aligned model π_K

```
1: // Let  $d = (I, M, S, T, R, t, s)$  denote a demonstration (i.e., a training sample)
2: for  $k = 1$  to  $K$  do
3:   // Collection Step: Generate demonstrations via expert policy
4:   Initialize  $\mathcal{C}_k \leftarrow \emptyset$  ▷ The dataset collected during iteration  $k$ 
5:   for  $(I, M) \sim p(\mathbf{I}, \mathbf{M})$  do
6:      $\tilde{\pi}_k \leftarrow \text{Amplify}(\pi_{k-1})$  ▷ Amplify current policy via model ensemble and best-of- $N$  search
7:     Sample  $\{d_i\}_{i=1}^N \sim \tilde{\pi}_k(I, M)$  ▷ Generate  $N$  candidate demonstrations from expert policy
8:      $d^*, r \leftarrow \text{HumanRank}(\{d_i\}_{i=1}^N)$  ▷ Humans select best candidate via pairwise comparisons
9:      $\mathcal{R} \leftarrow \{d_i : i \neq \arg \max\}$  ▷ Store rejected candidates for preference learning
10:     $\mathcal{C}_k \leftarrow \mathcal{C}_k \cup \{(d^*, r, \mathcal{R})\}$  ▷ Collect chosen demonstration with rating and rejections
11:   end for
12:   // Update Step: Train on aggregated high-quality demonstrations and preferences
13:    $\mathcal{C} \leftarrow \{(d^+, \mathcal{R}) : (d^+, r, \mathcal{R}) \in \bigcup_{i=1}^k \mathcal{C}_i, r \geq \alpha_k\}$  ▷ Aggregate and filter by quality
14:    $\mathcal{D} \leftarrow \{(d^+, d^-) : (d^+, \mathcal{R}) \in \mathcal{C}, d^- \in \mathcal{R}\}$  ▷ Create preference pairs for DPO training
15:    $\pi_k^{\text{SFT}} \leftarrow \arg \min_{\pi} \mathbb{E}_{(d^+, d^-) \sim \mathcal{D}} [\mathcal{L}_{\text{CFM}}(\pi; d^+)]$  ▷ Supervised finetuning
16:    $\pi_k \leftarrow \arg \min_{\pi} \mathbb{E}_{(d^+, d^-) \sim \mathcal{D}} [\mathcal{L}_{\text{DPO}}(\pi, \pi_k^{\text{SFT}}; d^+, d^-)]$  ▷ Align with preferences
17: end for
18: return  $\pi_K$ 
```

A.6 Core Alignment Algorithm

A.6.1 Basic Algorithm

Algorithm 1 shows the core alignment algorithm, used for all texture annotations and most shape annotations (MITL-3DO). During each collection step, we generate a set of predictions from the current model, and ask annotators to rank and verify these predictions. Generalist annotators can only choose between model outputs and accept/reject; they cannot edit. We maximize the probability of a successful annotation at each iteration by ensembling multiple models and combining multiple models with human preferences into an *expert* annotator.

The learning efficiency of the alignment, or the “speed of the data flywheel”, is controlled by two factors:

- **Amplification factor:** The size of the performance gap between current model and the expert annotations at each iteration
- **Stepwise efficiency:** How closely the new model approximates the previous expert from the previous iteration

The former induces an upper bound on the new policy’s performance at each iteration, while the latter describes how closely we approach that upper bound – similar to Expert Iteration (Anthony et al., 2017).

A.6.2 Training Intuition

Our goal in post-training (see Algorithm 1) is to align the model to match human preference on the distribution of *all* possible real-world objects.

The core algorithm in our data engine generates samples by asking humans to select viable samples from a set of candidate generations. Challenging inputs often result in no viable candidate generations and thus never get selected by humans. However, at any current time our model is usually good on some parts of the data distribution, but not on other parts, as shown in the cartoon below:

The intuition behind the data engine in SAM 3D is that these green islands of reliably good performance correspond to high-density parts of the training data (O’Neill et al., 2024; NVIDIA et al., 2025), and the approach in SAM 3D is that we want to push out from these islands of reliably good generations into the “tail”

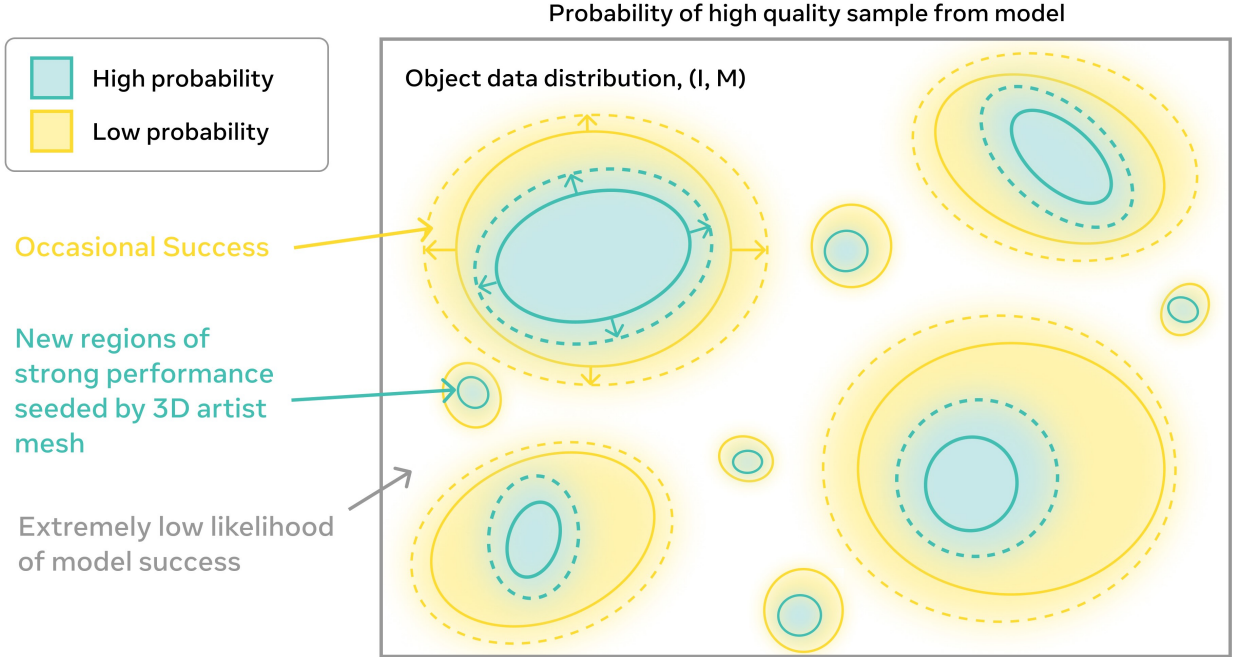


Figure 14 Simplified cartoon depiction of data engine improvement. The diagram depicts model sample quality (color) across the real-world distribution of images and masks. During training, the model begins by doing well (teal) on common categories and simple objects (chairs, bottles, signs, cars). Our goal is to both improve accuracy and robustness on these easy examples (teal), and then push the model to improve performance on less common objects (yellow) in the tail of the probability distribution. While the amplification stage of MITL generally leads to the slow expansion of existing regions of success, using 3D artists to create data for the hardest samples allows us to shortcut the process by “seeding” new regions of the data distribution, which may have taken us longer to reach through MITL verification alone.

of the distribution, demarcated by yellow and white background in the cartoon above. The yellow parts of the distribution are challenging for the model, but near enough to the blue islands, that we can *occasionally* generate satisfactory annotations, but it requires humans to go through many samples.

This can create a chicken-and-egg problem where, for the model to become good, it must already be capable of produce a good generation; at least some of the time. For examples that are so challenging that the probability of success is extremely low (white), the model has no hope and we ask human 3D artists (Section A.3) to provide supervision in this part of the data distribution, in order to seed new islands.

A.7 Increasing Amplification Factor with Search

A.7.1 Best-of- N Search with Reward Models

Qualitatively, however, we observe that re-visiting some (yellow) inputs with a large number of seeds, our model can sometimes still yield a few good generations (*e.g.*, food can take around ~ 50 seeds to reliably generate a successful mesh in Table 12). This suggests that increasing N in the best-of- N rejection sampling can potentially allow us to obtain annotations for challenging inputs, which would be difficult to source otherwise. Doing so would allow us to rapidly “push into the tail”, increasing the convergence speed of the alignment algorithm in Algorithm 1. However, the primary impediment to increasing N is that, at some point, there are too many choices for a human to compare. This linearly scales the annotation time of preference data collection, and the selections themselves become noisier and more random due to choice overload (Diehl and Poynor, 2010).

To address this challenge, we explored using learned reward models to perform a first pass in order to surface a smaller number of candidates for humans to then choose between. Figure 15 shows a pipeline to perform

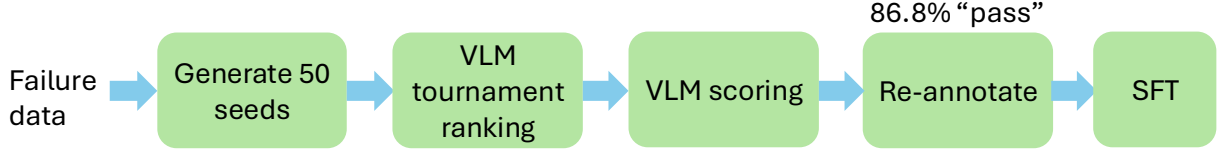


Figure 15 Reward model data recovery pipeline. The diagram shows how we use reward models to increase N in best-of- N search to improve the chance of a successful annotation on challenging tail inputs. We use both a VLM and also DPO implicit reward as reward models.

reward-ranked best-of- N search that increases the yield of successful annotations on challenging inputs. We first run 50 generations with different initial noise, and use the reward model to perform tournament-style ranking, and then pass the winning candidate to human annotators for ranking and verification (as in Stage 2).

We find that this approach indeed helps to recover some of this otherwise difficult data. For example, by scaling the best-of- N from $N = 2$ to $N = 50$ to recover samples that were originally discarded, improving the yield from 0% (since these were originally failures) to 86.8%. In particular, we observe significant increase in the proportion of successful annotations coming from challenging categories. The *food* category improves $9\times$ from 4% in the original annotated distribution to 36%. We show the experiments with resulting model performance, as well as ablations using VLMs instead of DPO implicit reward models, in [Section E.7](#).

A.8 Relationship to Self-Training Approaches

The data engine in SAM 3D can alternatively be viewed as an online alignment algorithm similar to RLHF ([Ouyang et al., 2022](#)) or related self-training methods. Under this interpretation, the generative model q is a policy and the data collection step is a policy evaluation; collecting demonstrations \mathcal{D}^+ and preferences $\mathcal{D}^+/\mathcal{D}^-$ through the interaction with the environment (annotators). The model improvement step simply updates the current policy using both finetuning and DPO.

This reframing helps make the relationship to existing work more clear. The most similar learning algorithm to our data engine is Expert Iteration (ExIt) ([Anthony et al., 2017](#)). As in ExIt, each iteration starts with a current policy, that we amplify using additional information into an expert policy, and we use this expert policy to generate supervision for imitation learning. Unlike ExIt, which uses purely imitation learning, we use humans-as-verifiers to select which samples to train on, and we make use of additional preference signals as reward signal ([Section 3.2.2](#)). However, there are also notable differences in type of supervision that can be used and the amplification steps. Our expert policy amplification step uses a model ensemble instead of only tree search with a value function, and we use preferences in the update step to better align to the task.

[Algorithm 1](#) uses reward ranking, similar to RAFT ([Dong et al., 2023](#)) and RFT ([Yuan et al., 2023](#)), although the alignment algorithm in SAM 3D adds explicit expert policies/ensembles and leverages preference supervision.

B Pretraining and Mid-Training Data Details

B.1 Iso-3DO Data Filtering

For the Iso-3DO data used for pretraining, the quality of the 3D meshes can vary substantially, and not all samples exhibit high-fidelity geometry. Such examples can ultimately prove harmful to model pretraining, even at scale. One way to filter data is by an aesthetic score filter, as employed by [Xiang et al. \(2025\)](#), which primarily emphasizes visual and textural appeal. We employ a similar filter process for the Texture & Refinement model.

However, this filter does not necessarily capture the geometric quality of a training data. Therefore, we develop a rule-based filtering strategy on shape to curate the pretraining data for the Geometry model, removing data with following characteristics:

- **Overly simplistic geometry**, characterized by extremely small volumes (*e.g.*, near-degenerate point-like structures) or minimal normal direction variation (*e.g.*, flat, sheet-like surfaces).
- **Structural outliers**, which includes meshes containing spatial outliers: isolated points or fragments that deviate significantly from the primary 3D structure.

B.2 Render-and-Paste Data Pipeline

We define the Render-Paste approach as follows: Given a natural image and an object instance defined by its mask segment, we replace the object in the image with a synthetic 3D object drawn from the same synthetic sources used in Iso-3DO. The size and position of the 3D object are determined using the 2D object mask together with a pointmap produced by a single-image depth estimator, which also guides the object’s visibility and occlusion to obtain a natural appearance in the final rendering. By nature of starting from a synthetic 3D object first, the resulting data (which we refer to as *RP-3DO*) has excellent 3D ground truth precision and pixel-alignment compared to subsequent data sources in our training pipeline, which much try to reconstruct 3D from partial 2D information as part of the data annoation process.

In the following sections, we introduce three variants of Render-Paste that differ along two axes: pose information and semantic relevance.

- **Section B.2.1: *Flying Occlusions (FO)*** inserts randomly oriented synthetic objects without pose information, resulting in pose-unaware but semantically loose composites.
- **Section B.2.2: *Object Swap – Random (OS-R)*** determines object scale and translation from masks and pointmaps, while using a random rotation and object. Beyond simple replacement, the incorporation of depth ordering provides meaningful visual cues for object size and spatial placement, yielding pose-aware but not fully aligned insertions with moderate semantic relevance, higher than in the Flying Occlusions setting.
- **Section B.2.3: *Object Swap – Annotated (OS-A)*** replaces the original object using the annotator-provided ground-truth scale, translation, and rotation, producing fully pose-aligned and semantically matched renderings.

B.2.1 Flying Occlusions (FO)

The aim of this dataset is to build invariance to occlusion and size variations that commonly occur in real-world scenarios—and to enable the model to leverage full image context instead of only object-centered crops—we construct a dataset of natural images with blended synthetic 3D objects. Inspired by Flying Chairs (Dosovitskiy et al., 2015) and FlyingThings3D (Mayer et al., 2016), we name our first variant Flying Occlusions, reflecting its use of freely inserted synthetic objects.

Each training example consists of a natural image onto which we composite two rendered 3D objects: an *occluder* and an *occludee*. For each pair, we also compute the final visible mask of the occludee after occlusion. To generate each training sample, we randomly pair a selected object with an occluder object. Given the mask of the selected object M_{obj} and the mask of the random occluder M_{occluder} , each corresponding to the full mask of the respective object, the visible mask is defined as $M_{\text{vis}} = M_{\text{obj}} \odot (1 - M_{\text{occluder}})$, where \odot denotes the element-wise (Hadamard) product. To ensure a reasonable degree of occlusion, we enforce $0.1 \leq |M_{\text{vis}}|/|M_{\text{obj}}| \leq 0.9$. In addition, samples with insufficient visibility are filtered out by requiring $|M_{\text{vis}}|/|I| \geq 0.2\%$, where $|I|$ is the total number of pixels in the image. Here, $|M|$ denotes the sum of all elements in M (*i.e.*, the total number of pixels with value 1 if M is a binary mask).

Finally, to prevent the model from always predicting the occluded object, in one third of the samples, we treat the selected mesh as the occluder. In these cases, the mask of the selected mesh is complete. In total, we have 55.1M sample with 2.87M unique meshes and 11.17M unique images.

B.2.2 Object Swap – Random (OS-R)

To enhance robustness to variations in object location and scale, we propose Object Swap – Random (OS-R), a depth-aware render-paste strategy that replaces an object in a natural image with a randomly selected synthetic mesh.

Given a natural image I , mask M , and random object mesh S , we synthesize a new training tuple $(I', M_{\text{vis}}, S, R, t, s)$. We first predict the 2.5D scene pointmap and identify the 3D centroid and bounding box of the target object. The original object is removed via inpainting, and we then insert a random synthetic mesh S at the computed centroid t with a random 3D rotation R . The mesh scale s is determined by fitting the mesh to the original object’s 3D bounding box.

We complete the process by re-rendering the new image I' with a z-buffer check. We render the new mesh into the inpainted image such that only pixels not occluded by existing scene geometry are visible, forming the visible mask M_{vis} . We filter samples with insufficient visibility ($< 20\%$ visibility) and update the pointmap P by projecting the unoccluded surface points of the new mesh, using M_{vis} .

To ensure the dataset provides sufficient visual cues for estimating translation and scale, we use heuristics to replace only objects that are partially occluded or supported along the bottom, which provides depth ordering and T-junction cues, respectively. We verify these cues by trying to find occlusion boundaries: we sample points on opposite sides of the mask border, and if the outer pixel is significantly closer to the camera than the inner pixel, we consider this part of the boundary occluded. A sample is retained if it meets one of two conditions: (1) *physical support*, where the background is closer to the camera along the bottom 10% of the object (indicating it rests on a surface), or (2) *partial occlusion*, where foreground elements occlude at least 10% of the total object perimeter. This process yields 5.95M training samples composed of 2.38M unique meshes and 1.20M unique images.

B.2.3 Object Swap – Annotated (OS-A)

In addition to the *Object Swap – Random* variant, we construct a complementary render-and-paste setting, which we refer to as *Object Swap – Annotated* (OS-A), which performs an in-place replacement of a real image with a rendered human-annotated object. The motivation for this dataset is to enable Texture & Refinement training that faithfully preserves pixel-aligned correspondence between the rendered mesh and the visual appearance of the target object in the image.

This approach closely follows the *OS-R* pipeline, with key distinctions arising from the use of human-annotated data in MITL-3DO. Specifically, each training sample is generated using an image from a curated MITL-3DO subset, where the initial object mask, selected mesh S , object placement (translation t , rotation R , and scale s), and target pose are all sourced from human annotations provided in the MITL-3DO dataset. The selected mesh for each object is chosen by annotators as the best available match to the object’s appearance in the image. During rendering, lighting conditions used for rendering are carefully matched to those in the training data preparation, ensuring consistent brightness and appearance across the dataset. We used a subset of the MITL-3DO shape-preference annotations, yielding 0.4 million training samples from this render-paste process.

B.3 Lighting for Texture Data

For Iso-3DO and RP-3DO (FO), we randomize the lighting (*i.e.*, direction and intensity) applied on the input images, and use ambient lighting when rendering views used to computing the target latents. Qualitatively, such data processing encourages the model to predict “de-lighted” textures without baking in strong directional shading or specular highlights from the input render. We verify that this is preferred by humans through preference tests (see “Lighting Aug” preference rate in Figure 17).

C Details on Model Training

The following sections outline the details of the Geometry and Texture & Refinement models, including architecture, training objective, and training hyperparameters.

C.1 Architecture Details on Geometry Model

We employ a latent flow matching model. For shape, it denoises the 64^3 voxels in the latent space of a coarser $16^3 \times 8$ representation, following [Xiang et al. \(2025\)](#). For layout, we perform denoising directly in the parameter space (R, t, s) , as their dimensionality is small. Additionally, we introduce modality-specific input and output projection layers to map both the shape and layout parameters into a shared feature space of dimension 1024, and subsequently project them back to their respective parameter spaces. This results in a total of 4096 tokens for the shape and 1 token for R, t, s , respectively, as input to the Mixture of Transformers (MoT). The MoT architecture comprises two transformers: one dedicated to the shape tokens, and a second whose parameters are shared for the layout parameters (R, t, s) , as shown in [Figure 2](#).

The MoT design allows independently training of some modalities while maintaining performance on others (*e.g.*, fine-tune shape or layout only), thanks to the structured attention mask illustrated in [Figure 2](#). This proves helpful when training on datasets that contain labels for only one modality (*e.g.* shape-only), and when freezing shape capabilities and finetuning just for layout. At the same time, MoT still allows for information sharing during the forward pass, through the joint self-attention layers for cross-modal interaction. This shared context is critical for self-consistency: notably, rotation is only meaningful when anchored to the predicted shape.

C.2 Pretraining & SFT Objective: Conditional Rectified Flow Matching

The is trained to jointly generate multiple 3D modalities using rectified conditional flow matching ([Liu et al., 2022](#)). Given an input image I and mask M , the Geometry model optimizes the following multi-modal flow matching objective:

$$\mathcal{L}_{\text{CFM}} = \sum_{m \in \mathcal{M}} \lambda_m \mathbb{E}_{\tau, \mathbf{x}_0^m} [\|\mathbf{v}^m - \mathbf{v}_\theta^m(\mathbf{x}_\tau^m, c, \tau)\|^2] \quad (1)$$

where $\mathcal{M} = \{S, R, t, s\}$ denotes the set of prediction modalities (shape, rotation, translation, scale), $c = (I, M)$ contains the conditioning modalities (image, mask), and \mathbf{v}_θ^m is the learned velocity field for modality m at the partially noised state, \mathbf{x}_τ^m .

We want to learn to generate clean states $\{\mathbf{x}_1^m\}_{m \in \mathcal{M}} \sim p(\mathcal{M}|c)$, and during training these are the ground-truth 3D annotations for each modality. Then, the target probability path at time $\tau \in [0, 1]$ is a linear interpolation $\mathbf{x}_\tau^m = \tau \mathbf{x}_1^m + (1 - \tau) \mathbf{x}_0^m$ between the target state \mathbf{x}_1^m and initial noise state $\mathbf{x}_0^m \sim \mathcal{N}(0, \mathbf{I})$. As a result, the target velocity field is the gradient of this linear interpolation $\mathbf{v}^m = \dot{\mathbf{x}}_\tau^m = (\mathbf{x}_1^m - \mathbf{x}_0^m)$. λ_m is simply a per-modality weighting coefficient.

The Texture & Refinement model optimizes analogous flow-matching objectives using SLAT features. We train both models using AdamW (without weight decay), and training hyperparameters such as sampling and learning rate schedules, EMA weights are in [Section C.7](#).

C.3 Preference Alignment Objective: DPO

For preference alignment in [Section 3.2.2](#), we follow Diffusion-DPO ([Wallace et al., 2024](#)) and adapt to flow matching as follows: given the same input image and mask c , we sample a pair of 3D output (x_0^w, x_0^l) based on human preference, where x_0^w is the preferred option and x_0^l is the less preferred. Our training objective is:

$$\mathcal{L}_{\text{DPO}} = -\mathbb{E} \begin{array}{l} I \sim \mathcal{I}, \\ (x_0^w, x_0^l) \sim \mathcal{X}_I^2 \\ \tau \sim \mathcal{U}(0, T) \\ x_\tau^w \sim q(x_\tau^w | x_0^w) \\ x_\tau^l \sim q(x_\tau^l | x_0^l) \end{array} [\log \sigma(-\beta T w(\tau) \cdot \Delta)] \quad (2)$$

$$\begin{aligned} \text{where } \Delta = & \|\mathbf{v}^w - \mathbf{v}_\theta(x_\tau^w, c, \tau)\|_2^2 - \|\mathbf{v}^w - \mathbf{v}_{\text{ref}}(x_\tau^w, c, \tau)\|_2^2 \\ & - (\|\mathbf{v}^l - \mathbf{v}_\theta(x_\tau^l, c, \tau)\|_2^2 - \|\mathbf{v}^l - \mathbf{v}_{\text{ref}}(x_\tau^l, c, \tau)\|_2^2) \end{aligned} \quad (3)$$

where \mathbf{v}^w and \mathbf{v}^l are the target flow-matching velocities for x_τ^w and x_τ^l , and \mathbf{v}_θ , \mathbf{v}_{ref} are the learned and frozen reference velocity fields, respectively.

Implementation details. We apply DPO on shape prediction in the Geometry model and the predictions of the Texture & Refinement model. We use the preference data collected in Stage 2, where we remove the negatives from non-SAM 3D generations (*e.g.* retrieval-based methods or multi-view diffusion texture generations), since they are out of the distribution for SAM 3D.

C.4 Model Distillation Objective: Shortcut Models

For applications needing online 3D perception capabilities (*e.g.* robotics), model inference time is an essential consideration. In diffusion and flow matching models, the most straightforward way to improve inference speed is by reducing the number of function evaluations (NFE). However, naïvely decreasing the number of steps can significantly degrade performance. Instead, we employ flow matching distillation techniques to reduce the number of inference steps while minimizing impact to quality. Specifically, we adopt the diffusion shortcut formulation from Frans et al. (2024), which offers several advantages over previous consistency distillation approaches: (1) it is simple, avoiding multi-stage training and instability; and (2) the model supports two modes, allowing seamless switching back to the original flow matching inference, so a single model can serve both purposes. Unlike the original formulation, we do not train shortcut models from scratch. Instead, we initialize from fully trained checkpoints and further finetune them with the shortcut objective.

$$\mathcal{L}_S(\theta) = \mathbb{E}_{\substack{x_0 \sim \mathcal{N}(0, I), \\ x_1 \sim q(x), \\ \tau, d \sim p(\tau, d)}} \left[\underbrace{\|\mathbf{v} - \mathbf{v}_\theta(x_\tau, c, \tau, d=0)\|^2}_{\text{Flow-Matching}} + \underbrace{\|\mathbf{v}_{\text{consistency}} - \mathbf{v}_\theta(x_\tau, c, \tau, 2d)\|^2}_{\text{Self-Consistency}} \right]. \quad (4)$$

where:

- $x_0 \sim \mathcal{N}(0, I)$: a Gaussian noise sample drawn from the standard normal distribution.
- $x_1 \sim p(x)$: a real data sample from the data distribution.
- x_τ : an interpolated sample between x_0 and x_1 at time step τ . (Defined earlier in the paper through the diffusion / flow matching path.)
- τ : the diffusion time (or noise level) at which the model predicts a local velocity or update step.
- d : the step size specifying how large a step the shortcut model should predict. $d = 0$ corresponds to flow-matching, $d > 0$ corresponds to consistency training.
- c : conditioning tokens.
- $p(\tau, d)$: the joint sampling distribution over diffusion times and step sizes used during training.
- $\mathbf{v}_\theta(x_\tau, c, \tau, d)$: the shortcut model parameterized by θ , taking as input the current sample x_τ , conditioning c , time τ , and desired step size d .
- \mathbf{v} : the empirical instantaneous velocity of the data flow used as the target for the flow-matching objective (corresponds to $d = 0$).
- $\mathbf{v}_{\text{consistency}}$: the self-consistency target, constructed by composing two steps of size d to form a reference for a single jump of size $2d$. Algorithm 2 describes how to construct $\mathbf{v}_{\text{consistency}}$.

We also distill CFG into the shortcut mode by using a fixed CFG strength of $w_{\text{CFG}} = 2$ for Stage 1 and CFG strength of $w_{\text{CFG}} = 1$ for Stage 2. The final model is fine-tuned for approximately 4K iterations using the same objective as in Frans et al. (2024): 75% flow matching and 25% shortcut. When shortcut mode is disabled, the model behaves identically to the original flow matching model. We initialize the step size embedder by setting the weights and bias of its final linear layer to zero, since, unlike the other parameters that have already been trained, these are new parameters introduced at the distillation stage. Figure 18 shows quantitative results and Figure 21 shows examples.

Algorithm 2 Consistency Target Construction with CFG Guidance for Shortcut Model Distillation

Require: Current state x_τ , conditioning c , step size d , CFG weight w_{CFG} , teacher model \mathbf{v}_θ

Ensure: Consistency target $\mathbf{v}_{\text{consistency}}$

- 1: *//* First shortcut step with CFG guidance
 - 2: $\mathbf{v}_\tau \leftarrow \mathbf{v}_\theta(x_\tau, \emptyset, \tau, d) + w_{\text{CFG}}(\mathbf{v}_\theta(x_\tau, c, \tau, d) - \mathbf{v}_\theta(x_\tau, \emptyset, \tau, d))$ ▷ Apply CFG to get guided velocity
 - 3: $\tilde{x}_{\tau+d} \leftarrow x_\tau + d \cdot \mathbf{v}_\tau$ ▷ Take first step of size d
 - 4: *//* Second shortcut step with CFG guidance
 - 5: $\mathbf{v}_{\tau+d} \leftarrow \mathbf{v}_\theta(\tilde{x}_{\tau+d}, \emptyset, \tau + d, d) + w_{\text{CFG}}(\mathbf{v}_\theta(\tilde{x}_{\tau+d}, c, \tau + d, d) - \mathbf{v}_\theta(\tilde{x}_{\tau+d}, \emptyset, \tau + d, d))$ ▷ Apply CFG at new state
 - 6: $\tilde{x}_{\tau+2d} \leftarrow \tilde{x}_{\tau+d} + d \cdot \mathbf{v}_{\tau+d}$ ▷ Take second step of size d
 - 7: *//* Compute consistency target from two-step trajectory
 - 8: $\mathbf{v}_{\text{consistency}} \leftarrow \text{stopgrad}\left(\frac{\tilde{x}_{\tau+2d} - x_\tau}{2d}\right)$ ▷ Average velocity over combined $2d$ step
 - 9: **return** $\mathbf{v}_{\text{consistency}}$
-

C.5 Texture & Refinement Training Details

We train the Texture & Refinement model following a multi-stage training paradigm analogous to that of the Geometry model (described in [Section 3](#)). Below we provide implementation details for the texture training stages.

VAE Training. Learning the inverse problem of image to texture map requires a strong alignment in training data between them. However, in previous work ([Xiang et al., 2025](#)), renderings of meshes create artifacts like reflections and shadows; objects consistently having extremely dark bottoms is one such common example. To curate a cleaner set of ground truth data, we create our latent SLAT with a “de-lighted rendering” by using ambient lighting to minimize such artifacts. We use this to process all stages of data. We select a uniform lighting strength, based on computing the similarities between SAM 3D predictions and the original image using RP-3DO (FO).

Pretraining. We start with pretraining the model on Iso-3DO-500K, a partition of Iso-3DO data with high aesthetics. Training on such data allows the model to predict plausible, high-quality texture that is characteristic of many 3D asset generation models. To ensure robustness of the texture model on real-world, complex images, we must further train on increasingly challenging data, via additional training stages described below.

For pretraining data in Iso-3DO, we render conditional images from the 3D mesh. We introduce a random lighting augmentation, where for each view, we apply a random lighting setting in the rendering engine. We hope the model can learn to remove the lighting effects when generating textures.

Mid-training. During mid-training, we train on RP-3DO (FO and OS-A). It is starting at this stage where we additionally provide full image conditioning to the model, as we believe contextual cues can help the model predict plausible textures, especially when the object is heavily occluded. We show the effect of training on RP-3DO (FO), as well as the effect of further adding RP-3DO (OS-A) training data to this stage, in [Figure 17](#).

We also introduce image data augmentation for RP-3DO (FO): **Mask** augmentation and **Blur** augmentation. The Mask augmentation randomly erode or dilute the input mask. This is designed to handle noise in mask at inference time (*e.g.* segmentation predictions from a model). The Blur augmentation applies a downsample operation on the image followed by an upsample operation. This is especially important to handle cases for motion blur and small objects in an image. These augmentation is carefully studied in [Section E.2](#).

Supervised fine-tuning (SFT). In the SFT stage, the model is trained on MITL-3DO texture annotations, which includes the “aesthetic” samples described in [Section A.1](#). We show the effect of scaling the MITL-3DO texture annotations by 2x in [Figure 17](#), which improves human preference rate by 14.2%.

Preference optimization. Like the Geometry model, we run a final DPO stage to align the model with human preferences collected from the texture data engine. The effect of DPO on texture performance is shown in

| Training stage | Datasets | Condition input | Learning rate | Modality weights | # Meshes | # Images | # Samples | # Tokens |
|----------------|-------------------------|----------------------|----------------------|------------------------------|----------|----------|-----------|----------|
| Pre-training | Iso-3DO | object-centric crop | 10^{-4} | $S=1.0, R=0.1$ | 2.7M | 64.8M | 64.8M | 2.5T |
| Mid-training | RP-3DO (FO) | full image | 10^{-4} | $S=1.0, R=0.1$ | 2.87M | 11.17M | 55.1M | 2.4T |
| | RP-3DO (OS-R), ProcThor | full image, pointmap | 10^{-4} | $S=1.0, R=0.1, t=1.0, s=0.1$ | 2.38M | 1.20M | 5.95M | 0.3T |
| SFT | MITL-3DO, Art-3DO | full image, pointmap | 10^{-5} | $S=1.0, R=0.1, t=1.0, s=0.1$ | 0.6M | 0.5M | 0.6M | 0.9T |
| Alignment | MITL preference | full image, pointmap | 2.5×10^{-6} | $S=1.0$ | 88K | 31K | 44K | - |

Table 5 Detailed training hyperparameters for SAM 3D training stages (Geometry Model). This table extends Table 1 from the main paper with additional hyperparameter details.

both Table 4 and Figure 17. We follow the same training objective described in Equation (4).

C.6 Texture & Refinement VAE

We make improvements over the original SLAT VAE design in Xiang et al. (2025), where features are back-projected to all voxels, including those that are not visible (*i.e.*, occluded) from the current image. This original design choice leads to reduced sharpness in the reconstructed images. To address this issue, we back-project features only to voxels that are visible from each image, utilizing the depth information from that specific view. We call this VAE variant Depth-VAE. During training, we normalize the Kullback–Leibler (KL) regularization term by the active voxel count to prevent large objects from dominating the training loss. We also fine-tune the decoder for downstream needs, such as reducing the number of decoded Gaussians for faster inference.

C.6.1 Depth-VAE: Depth-Aware Feature Aggregation

To integrate depth information into patch-level features, we propose a depth-guided projection algorithm. Given a feature map $\mathbf{F} \in \mathbb{R}^{B \times C \times H \times W}$ and normalized 2D coordinates $\mathbf{U} \in [-1, 1]^{B \times N \times 2}$, we sample features and aggregate them based on visibility, handling occlusions via a depth buffer.

Feature Sampling. For each coordinate $\mathbf{u}_i \in \mathbf{U}$, we extract the corresponding feature vector \mathbf{f}_i from the DINO-V2 map \mathbf{F} using differentiable bilinear interpolation (denoted as `GridSample`).

Occlusion Handling (Depth Filtering). To identify visible points, we construct a temporary depth buffer. We map the coordinates \mathbf{U} to a discrete grid (P_x, P_y) and retain the minimum predicted depth \hat{d} at each location to form a surface depth map \mathbf{D}_{surf} :

$$\mathbf{D}_{\text{surf}}(x, y) = \min_{i: (x_i, y_i) = (x, y)} \hat{d}_i. \quad (5)$$

We then resample this map at the original coordinates \mathbf{U} to obtain the reference surface depth \mathbf{d}_{ref} . Note that if ground-truth depth is available, it is used in place of \mathbf{D}_{surf} .

Visibility Masking. We compute a binary mask \mathbf{M} to discard occluded points. A point is considered visible if its depth \hat{d}_i is within a tolerance τ of the reference surface $\mathbf{d}_{\text{ref}, i}$:

$$\mathbf{M}_i = \mathbb{I} \left[\mathbf{d}_{\text{ref}, i} > \hat{d}_i - \tau \right]. \quad (6)$$

We normalize this mask across the batch dimension (or views) to obtain weights $\tilde{\mathbf{M}}$.

Weighted Aggregation. The final depth-aware representation is the weighted sum of visible patch features:

$$\mathbf{F}_{\text{depth}} = \sum_b \tilde{\mathbf{M}}_b \odot \mathbf{f}_b. \quad (7)$$

C.7 Training Hyperparameters

We summarize training parameters in details in Table 5 for Geometry model and Table 6 for Texture & Refinement model.

| Training stage | Datasets | Condition input | Learning rate | EMA | # Meshes | # Images | # Samples | # Tokens |
|----------------|------------------|---------------------|---------------|--------|----------|----------|-----------|----------|
| Pre-training | Trellis500K | object-centric crop | 10^{-4} | 0.9999 | 350K | 9M | 10M | 1.1T |
| Mid-training | RP-3DO (FO,OS-A) | full image | 10^{-4} | 0.9999 | 800K | 2.4M | 2.4M | 1T |
| SFT | MITL | full image | 10^{-5} | 0.999 | ~100K | 100K | 100K | 115B |
| Alignment | MITL preference | full image | 10^{-6} | 0.99 | 146K | 73K | 73K | - |

Table 6 Detailed training hyperparameters for SAM 3D training stages (Texture & Refinement Model). This table extends Table 1 from the main paper with additional hyperparameter details.

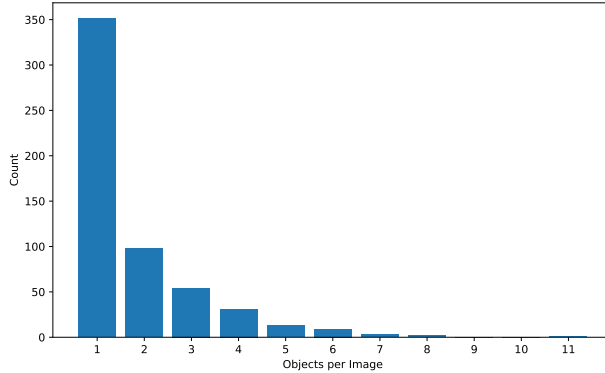


Figure 16 Distribution of number of objects per image in SA-3DAO. The number of objects follows a roughly power-law distribution.

We use a batch size of 6 per GPU for all training stages of the Geometry model, and epochs iterate over meshes. Pretraining is conducted on 512 A100 GPUs for 200 epochs. Mid-training on FO utilizes 320 A100 GPUs for 50 epochs, followed by further FO mid-training on 128 A100 GPUs for an additional 50 epochs, followed by OS-R midtraining on 256 A100s for 12 epochs. SFT is performed on 128 H200 GPUs for 100 epochs, training on data from our data engine as it becomes available. As this data leads to model improvements (and thus also improving the quality of data produced by the data engine), we raise our quality threshold α_k for keeping samples in our SFT training set; the final run uses a quality cutoff α_K that keeps 500K samples. DPO is performed on 128 A100 GPUs for 1 epoch.

For the Texture & Refinement model, we perform pretraining on 256 A100s for 245 epochs with a batch size of 4, followed by mid-training on 256 A100s for 80 epochs with a batch size of 4. For SFT, we use 192 A100s for 89 epochs and batch size of 4. Finally, DPO is conducted on 128 A100s for 2 epochs with a batch size of 3.

D Evaluation

Current evaluation benchmarks for visually grounded 3D object reconstruction fall short of capturing the complexity of the real world. Many rely on synthetic datasets (Deitke et al., 2023; Chang et al., 2015) where single objects are rendered in isolation, centered against a white background. This introduces a large visual gap with real-world evaluation conditions and the rich variation of real-world imagery. Efforts to move to real data mostly focus on indoor environments (Khanna et al., 2024; Sun et al., 2018; Pan et al., 2023), but these benchmarks heavily skew toward furniture categories such as chairs and tables, limiting the diversity of objects that models must handle in practice. As a result, evaluations on these datasets do not reflect the challenges of natural environments, where objects are occluded, scenes are cluttered, scales vary, lighting conditions complicate appearance, and domains span far beyond indoor or synthetic scenes.

D.1 SA-3DAO: A New Benchmark for Real-World 3D Object Reconstruction

We introduce SAM 3D Artist Objects (SA-3DAO), a new benchmark designed to capture the diversity and complexity of real-world 3D perception. SA-3DAO consists of 1,000 untextured 3D objects, created from and carefully aligned to selected natural images capturing scenes spanning both indoor and outdoor environments, including parks, ski resorts, flea markets, parades and more. The benchmark covers a wide spectrum of object types: objects range from large, structured entities such as ski lifts and escalators, to everyday items like

clothing, to rare and culturally specific objects such as tribal face masks. Crucially, the 3D ground truth are of high-fidelity created by professional 3D artists, who are tasked with producing accurate 3D shapes for the objects depicted in the input images. This combination of visual diversity, real-world context, and professionally crafted 3D ground truth makes SA-3DAO a comprehensive testbed for evaluating 3D object reconstruction models.

Collection details. We task professional 3D artists with recovering the shape of a target object from a single image, mirroring our model’s goal of reversing the photographic transform (as described in [Section 2.1](#)). In other words, the 3D artists must create a whole 3D mesh that precisely aligns with the object’s visible pixels in the image. Even under ideal settings, this requires contending with only partial information as the back side of the object is typically unseen, but many of the objects in SA-3DAO additionally have natural occlusions, or are small in size within the image; disentangling depth versus scale is often also challenging for a single image. To fill these information gaps, artists rely on recognition and context, using common-sense priors, physical plausibility, and assumptions of symmetry (when appropriate) to complete the meshes. The requirements imposed by this task are atypical to the normal 3D asset creation process that artists are more accustomed to, and efficient annotation requires learning a different mode of operation. After acclimation to the task, completion time per object mesh can vary considerably, ranging from up to 5 minutes for obvious objects with simple geometries to over 5 hours for more challenging cases; the median mesh in our dataset has 4751 vertices. Many of the images provided multiple objects with meshes from the 3D artists; we show the frequency distribution in [Figure 16](#).

D.2 Human Preference Set

We further expand our evaluation suite to support more rigorous and domain-targeted assessments. While SA-3DAO provides a general and standardized way to measure progress, we want to also capture the challenges of settings where 3D perception is most critical, such as robotic manipulation and egocentric vision. To address this, we design a human preference set composed of images drawn from these domains of interest. This set enables evaluation through direct human judgment, providing insights that go beyond numerical metrics and capturing aspects of 3D perception that are important in embodied and real-world applications.

Domains. We design four human preference test datasets to comprehensively evaluate model capabilities across different scenarios.

- **SA-1B** ([Kirillov et al., 2023](#)): We uniformly sample 1,000 image and object mask pairs, covering a diverse range of object categories. This set is intended to assess the model’s generalization ability across varied object distributions, with particular emphasis on long-tail categories.
- **MetaCLIP** ([Xu et al., 2024](#)): We select 1,000 samples where the object masks are of median or heavily occluded. This set evaluates the model’s performance in reconstructing occluded objects, a common challenge in cluttered scenes.
- **LVIS** ([Gupta et al., 2019](#)): We densely sample 1,000 images containing between 10 and 30 objects per scene, and is designed to evaluate the model’s transferability to out-of-domain data and to demonstrate its ability to capture physical properties within dense scene layouts.
- **Aria Digital Twin** ([Pan et al., 2023](#)): We sample a smaller set of 40 video frames, with around 30 objects per scene. This dataset is intended to compare against baselines on scenes with highly accurate pointmaps, similar to those on which the baselines were trained.

Setup. Human preference evaluations are conducted through a structured sequence of pairwise comparisons. For each image and a masked object, annotators are first presented with two reconstructions (model “A” vs. “B”) and asked to select the one that most accurately matches the object in the image. The chosen reconstruction is then compared against the output of a third model (model “C”), and this process continues iteratively until all candidate models have been compared. Through this series of binary decisions, the most accurate reconstruction is identified as the preference for that particular image. To ensure fairness and avoid bias, the order of comparisons is randomized and the identity of models are anonymized.

D.3 Evaluation Metrics

D.3.1 Shape Metrics Definitions

For shape evaluation on SA-3DAO, we first normalize the artist-created ground-truth mesh and the generated mesh independently into the range $[-1, 1]$. We then apply ICP alignment for each mesh pair before computing metrics. We uniformly sample 1M points from both meshes and report four complementary metrics that capture different aspects of geometric fidelity:

- **F-score @ 0.01:** Measures correspondence accuracy between the reconstructed and ground-truth points under a 0.01 threshold. We compute precision and recall between the two point clouds and report their harmonic mean. F1 evaluates how many points lie close to the ground truth and how completely the reconstruction covers the target shape.
- **Voxel-IoU:** Provides a coarse volumetric agreement score and is sensitive to gross errors in volume, silhouette, and topology. We voxelize both point clouds to 64^3 resolution and compute intersection-over-union over occupied voxels.
- **Chamfer Distance (CD):** Measures bidirectional nearest-neighbor distance between reconstructed and ground-truth point sets, highlighting fine-grained geometric deviation and penalizing missing or distorted regions.
- **Earth Mover’s Distance (EMD):** Quantifies the minimal cost required to transport one point distribution to match the other. EMD is more stringent than CD, capturing global structural differences and enforcing bijective correspondence between distributions.

Together, these metrics provide a comprehensive view of reconstruction fidelity, from local accuracy to global shape consistency.

Moreover, to evaluate shape quality on the ISO3D dataset (Ebert, 2025)—which consists of 101 in-the-wild synthetic images without 3D ground truth—we measure perceptual similarity between the generated shape and the input image using ULIP (Xue et al., 2023) and Uni3D (Zhou et al., 2023). For each generated mesh, we uniformly sample 8,192 surface points to form a point cloud representation, and compute cross-modal similarity between the point cloud features and image features.

D.3.2 Layout Metrics Definitions

To evaluate single-object pose and compare with existing methods, we employ standard 6D pose estimation metrics, and then define ICP rotation error below.

- **3D IoU:** Measures the overlap of 3D axis-aligned bounding boxes between predicted and ground-truth bounding boxes, using the intersection-over-union. Values range from 0 (no overlap) to 1 (perfect overlap).
- **ICP-Rot:** *ICP Rotation Error* is the residual rotation error (in degrees) after ICP alignment. Given predicted rotation R_{pred} and ground-truth rotation R_{gt} , the meshes are first posed, then ICP finds optimal alignment R_{ICP} , and **ICP-Rot** is the angle of this rotation in degrees.
- **ADD-S (Average Distance with Symmetry):** ADD-S (Xiang et al., 2018) is the symmetrized average of the minimum point-to-point distances between predicted and ground-truth posed objects, normalized by object diameter:

$$\text{ADD}(\mathcal{A}, \mathcal{B}) = \frac{1}{|\mathcal{A}|} \sum_{\mathbf{x} \in \mathcal{A}} \min_{\mathbf{y} \in \mathcal{B}} \|\mathbf{x} - \mathbf{y}\|_2 \quad (8)$$

$$\text{ADD-S} = \frac{\text{ADD}(\mathcal{M}, \mathcal{M}_{\text{gt}}) + \text{ADD}(\mathcal{M}_{\text{gt}}, \mathcal{M})}{2d} \quad (9)$$

where \mathcal{M} and \mathcal{M}_{gt} are the predicted and ground-truth point clouds for the posed shape, and $d = \max_{\mathbf{x}, \mathbf{y} \in \mathcal{M}_{\text{gt}}} \|\mathbf{x} - \mathbf{y}\|_2$ is the diameter of the ground-truth point cloud. The symmetrized formulation averages distances in both directions: from predicted to ground-truth and from ground-truth to predicted. Lower values indicate better pose accuracy.

The original ADD-S metric definition (Xiang et al., 2018) was designed for 6DoF pose estimation using a ground truth CAD model. In this case, when the predicted and ground truth shape are the same, the asymmetric and symmetric versions of ADD-S coincide. In SAM 3D we jointly estimate shape and pose, so generalize the metric to the symmetric version.

- **ADD-S @ 0.1:** A binary value per-sample indicating whether the ADD-S distance is less than 10% of the object’s diameter.

E Additional Ablations

E.1 Intermediate Training Stage Knockout

While Table 4 in the main paper shows the cumulative effect of adding different stages during training, Table 7 shows the impact of real-world data as intermediate stages. Knocking out any of these stages results in a notable drop in shape performance.

| Model | Training Setup | SA-3DAO | | | |
|--------|--------------------------|--------------------------|---------------------|--------------------------|----------------------|
| | | F1 @ 0.01 (\uparrow) | vIoU (\uparrow) | Chamfer (\downarrow) | EMD (\downarrow) |
| SAM 3D | Full | 0.2344 | 0.2311 | 0.0400 | 0.1211 |
| | w/o training on MITL-3DO | 0.2211 | 0.2220 | 0.0486 | 0.1338 |
| | w/o training on Art-3DO | 0.2027 | 0.2025 | 0.0578 | 0.1510 |
| | w/o DPO on MITL-3DO | 0.2156 | 0.2156 | 0.0498 | 0.1367 |

Table 7 Training stage knockout. The impact of training on MITL and 3D artist-generated data.

E.2 Texture Evaluations

Comparison to SOTA. We compare SAM 3D with existing methods on a holistic level (combined geometry and texture prediction) in Table 8. We compare against existing image-to-3D methods that predict Gaussians or textured meshes, including Trellis (Xiang et al., 2025) and Hunyuan3D-2.1 (Hunyuan3D et al., 2025). We also conduct a texture-only comparison by providing SAM 3D geometry as input to the texture modules of the aforementioned baselines, with the addition of Unitex (Liang et al., 2025b), a model that performs texture prediction given paired image and shape input.

| Model | SAM 3D WR over baselines, SAM 3D shape | | | |
|---------------|--|----------------|---------|------|
| | iso3d | Preference Set | SA-3DAO | LVIS |
| Trellis | 81.1 | 87.0 | 86.2 | 89.1 |
| Hunyuan3D-2.1 | 63.8 | 87.0 | 86.2 | 89.1 |
| Hunyuan3D-2.0 | 70.1 | 77.5 | 77.4 | 85.7 |
| Unitex | 83.3 | 84.7 | 84.5 | 88.3 |

Table 8 3D texture. Preference results comparing SAM 3D to competing image-to-3D methods on ISO3D (Ebert, 2025), Preference Set, and SA-3DAO. We compare to the recent Trellis (Xiang et al., 2025), Hunyuan3D-2.1 (Hunyuan3D et al., 2025), and Unitex (Liang et al., 2025b), with the same shape of SAM 3D. The human preference rates represent preference for SAM 3D over each baseline approach.

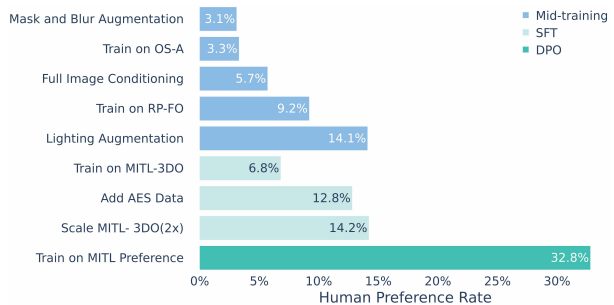


Figure 17 Ablations for the Texture & Refinement model, grouped by training stage. Percentages denote human preference rate for each ablation, over the model *without* the ablation.

We report human preference for SAM 3D over each baseline method on multiple datasets: ISO3D (Ebert, 2025), Preference Set, SA-3DAO, and LVIS (Gupta et al., 2019). Results indicate that SAM 3D outperforms existing methods on the holistic image-to-3D task as well as texture estimation – this is due to the fact that the evaluated datasets often contain occlusion and clutter, which is a setting prior works struggle on.

Ablations for Texture Training. We conduct comprehensive studies on design choices for Textue & Refinement model (Figure 17) using annotator preferences on the Pref Set. We benchmark each component to the alternative model without the change. We remark a few themes here:

- Augmentation is very important, with lighting augmentation to be the most critical here. This is expected, given the Mask and Blur augmentations primarily focus on specific challenging cases (poor mask quality and low resolution inputs), so their effects get diluted in a holistic evaluation.
- RP-3DO data are critical and helps the model adapt to real world.
- Post-training data are critical, with significant gains coming from it. It demonstrates the effectiveness of our Data Engine, and DPO further amplifies the gains. In addition, sourcing specific type of data (AES) and scaling the data both show significant improvements.

E.3 Layout Test-Time Optimization

Render-and-compare is a longstanding popular approach for pose estimation (Labbé et al., 2022; Wen et al., 2024): iteratively render the object’s shape according to the most recently predicted pose, directly compare with the input pixels, and then adjust the pose prediction accordingly. This heuristic-based search can lead to fairly accurate results, even with weak initial pose samples (“proposals” from a base model). By contrast, SAM 3D operates in a feedforward manner, directly diffusing the object pose (rotation, translation, and scale) conditioned on the image features and, optionally, the scene pointmap. Notably, we do not include any sort of pixel-based loss objective in this process. However, SAM 3D’s pose estimation can very naturally be used as a proposal for render-and-compare optimization. We include experiments showing the impact of this post-optimization process in Table 9, demonstrating that we can achieve further gains in pose metrics on ADT (Pan et al., 2023).

| Model | 3D IoU (\uparrow) | ICP-Rot. (\downarrow) | ADD-S (\downarrow) | ADD-S @ 0.1 (\uparrow) | 2D IoU (\uparrow) |
|---------------------|-----------------------|---------------------------|------------------------|----------------------------|-----------------------|
| SAM 3D | 0.4837 | 14.4702 | 0.08265 | 0.7545 | 0.5143 |
| SAM 3D (post-optim) | 0.5258 | 14.1460 | 0.07932 | 0.7617 | 0.6487 |

Table 9 Test-time optimization for layout. Quantitative comparison of SAM 3D layout test-time optimization on Aria Digital Twin (Pan et al., 2023).

Specifically, we further optimize the layout proposals from SAM 3D by applying the layout to the generated objects, rendering and comparing for both masks and pixels, and backpropagating the gradients to refine the layout. We finally apply an automatic proposal-checking step, where the optimized layout is accepted only if its mask IoU exceeds that of the initial layout. As shown in Table 9, for the 554 accepted optimized samples out of the 1027 ADT instances, both the 3D layout metrics and the 2D mask IoU metric improve substantially, demonstrating the effectiveness of layout test-time optimization.

E.4 Rotation Representation

We compare different rotation representations while keeping all other architecture and training settings identical. Each model is trained on the pretraining dataset and evaluated on a held-out Objaverse test split of 216 samples. As shown in Table 10, switching from a quaternion representation to the 6D continuous rotation parameterization (Zhou et al., 2019) yields a notable reduction in oriented rotation error, confirming that the 6D formulation provides a smoother optimization landscape more suitable for generative modeling. Further applying normalization to the 6D rotation vectors using the statistics over the training datasets leads to an additional improvement when training the flow matching models.

| Representation | Chamfer (\downarrow) | ICP-Rot. (\downarrow) |
|------------------------|--------------------------|---------------------------|
| Quaternion | 0.0061 | 17.9585 |
| 6D Rotation | 0.0074 | 15.5399 |
| Normalized 6D Rotation | 0.0049 | 14.5946 |

Table 10 Rotation representation. Ablation on the representation used during pretraining. We report Chamfer distances and ICP rotation error.

E.5 Pointmap Minimally Affects Shape

SAM 3D can condition on a 2.5D pointmap derived from sensor measurements (see Section 2.2) or from an off-the-shelf monocular depth estimation derived from the image itself, as is primarily used throughout this work. The latter is notable, as it also means that the current model can continue to benefit from future improvements of depth estimation methods. We observe that the pointmap minimally affects the shape performance: in a head-to-head preference test for shape on LVIS, the version of SAM 3D conditioned on pointmaps and the version without pointmaps are each selected 48% of the time.

E.6 Texture & Refinement Depth-VAE Comparison

Table 11 shows the result of the improvements made to the VAE used in the Texture & Refinement model; see Section C.6. We found that the depth feature significantly improves the perceptual quality of reconstruction, while scaling the training data further improves the reconstruction performance. We also notice that the enhancement primarily arises from the difficult scenarios for the non-depth VAE (when it has a bad performance).

| Method | PSNR (\uparrow) | SSIM (\uparrow) | LPIPS (\downarrow) |
|---------------------|---------------------|---------------------|------------------------|
| Non-Depth VAE | 30.65 | 0.9470 | 0.04776 |
| Depth-VAE | 30.87 | 0.9500 | 0.04579 |
| Depth-VAE + scaling | 31.60 | 0.9547 | 0.04093 |

Table 11 Depth-VAE ablations. Effectiveness the depth-feature modification to the SLAT VAEs used in the Texture & refinement model. Results are evaluated on the entire GSO dataset.

E.7 Data Engine: Increasing Best-of- N Search with Reward Models

| | Tail Holdout | | Epic Kitchens | | SA-3DAO | |
|----------------------------|----------------------|---------------|----------------------|---------------|----------------------|---------------|
| | Chamfer \downarrow | F1 \uparrow | Chamfer \downarrow | F1 \uparrow | Chamfer \downarrow | F1 \uparrow |
| SFT with $N = 2$ | 0.0059 | 0.39 | 0.0094 | 0.30 | 0.0083 | 0.26 |
| SFT with $N = 50$ recovery | 0.0053 | 0.41 | 0.0090 | 0.32 | 0.0081 | 0.26 |

Table 12 Including reward-model-recovered data during SFT, from the pipeline in Figure 15, improves model performance on challenging inputs, as seen in both Chamfer Distance and F1 score on the tail holdout set, Epic Kitchens (Damen et al., 2020), and SA-3DAO.

Finetuning on data recovered using the reward-model-best-of- N pipeline improves model performance on various challenging inputs, such as the artist evaluation set, tail holdout set, as well as Epic Kitchens (Damen et al., 2020), as shown in Table 12. This demonstrates that further amplifying the expert policy in Algorithm 1 by increasing N in the best-of- N search can improve the robustness of the model in challenging categories, and suggests that improved test-time search can increase the alignment convergence speed of the data engine.

We found that both vision-language models (VLMs), and also the implicit reward models from our DPO stage (Lambert et al., 2024) performed similarly in our case. In our testing, the VLM-as-reward model had 68.9% binary choice agreement with humans rater preferences, DPO agreed $\sim 65\%$, and two human annotators agreed $< 75\%$ of the time. Around 80% of the recovery data came from the DPO-as-reward model.

E.8 Model Distillation Results

Figure 18 illustrates the performance of SAM 3D with and without distillation, plotted against the number of flow matching steps for the Geometry model. Specifically, using 1-step and 4-step methods yields a $38\times$ and $10\times$ inference speed improvement, respectively, compared to 25 steps with flow matching (w/ CFG). In flow matching mode, CFG is applied during the first half of the steps, resulting in a total NFE that is 1.5 times the number of steps. In contrast, shortcut mode achieves an NFE equal to the number of steps, as CFG is distilled directly into the model. For the Texture & Refinement model, we opted not to apply distillation for the final release, as the final model already performs well with fewer steps out of the box. This is because the overall geometry is primarily determined by the Geometry model’s voxel output, and increasing the number

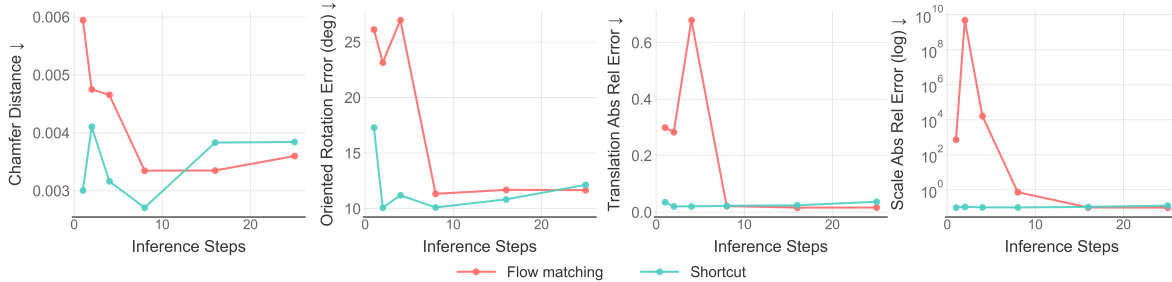


Figure 18 Model distillation. Geometry model shortcut versus flow matching. Flow matching distillation enables the model to perform significantly better during the early iterations, nearly on par with 25 steps performance.

of steps does not significantly alter the geometry. However, as illustrated in Figure 21, where we present a visualization of the model outputs using different numbers of inference steps with shortcut mode enabled for both the Geometry model and the Texture & Refinement model, shortcut model distillation improves texture quality when using fewer steps.

F Limitations

There are limits to our model’s resolution based on the architectural hyperparameters we used. The geometry model uses a coarse shape resolution of $O \in \mathbb{R}^{64^3}$; we trained multiple 3D Gaussian splat decoders, with up to 32 splats per occupied voxel. This is sufficient for many types of objects, but for more complex shapes or where human perception is especially attuned, these limits to resolution can lead to noticeable distortions or loss of details. For example, as part of a whole human body, the number of voxels/splats our chosen resolution is able to devote to hands or faces is inherently limited by the overall body’s scale, and due to human visual system’s acuity to such features, this can lead to perceptible artifacts to these body parts. By contrast, when focused on just a single hand or the head, the higher relative resolution available means SAM 3D can reconstruct these significantly better. For these kinds of objects and others, a natural next step for SAM 3D would be to increase the output resolution via architectural changes, a superresolution model, parts-based generation, or switching to an implicit 3D representation.

Object layouts are another area where improvements can be made. SAM 3D predicts objects one at a time, and isn’t trained to reason about physical interactions, such as contact, physical stability, interpenetration, or co-alignment (*e.g.* on the same ground plane). Multi-object prediction combined with appropriate losses would allow joint reasoning about multiple objects in a scene. Further, SAM 3D’s texture predictions are made without knowledge of the predicted object’s pose. As a result, for objects with rotational symmetry, it can occasionally predict textures that effectively rotate the object to the incorrect orientation.



Figure 19 Additional qualitative shape and texture results of our model on the SA-3DAO eval set. For models that include texture, we show the untextured mesh (left) and textured mesh (right) separately.

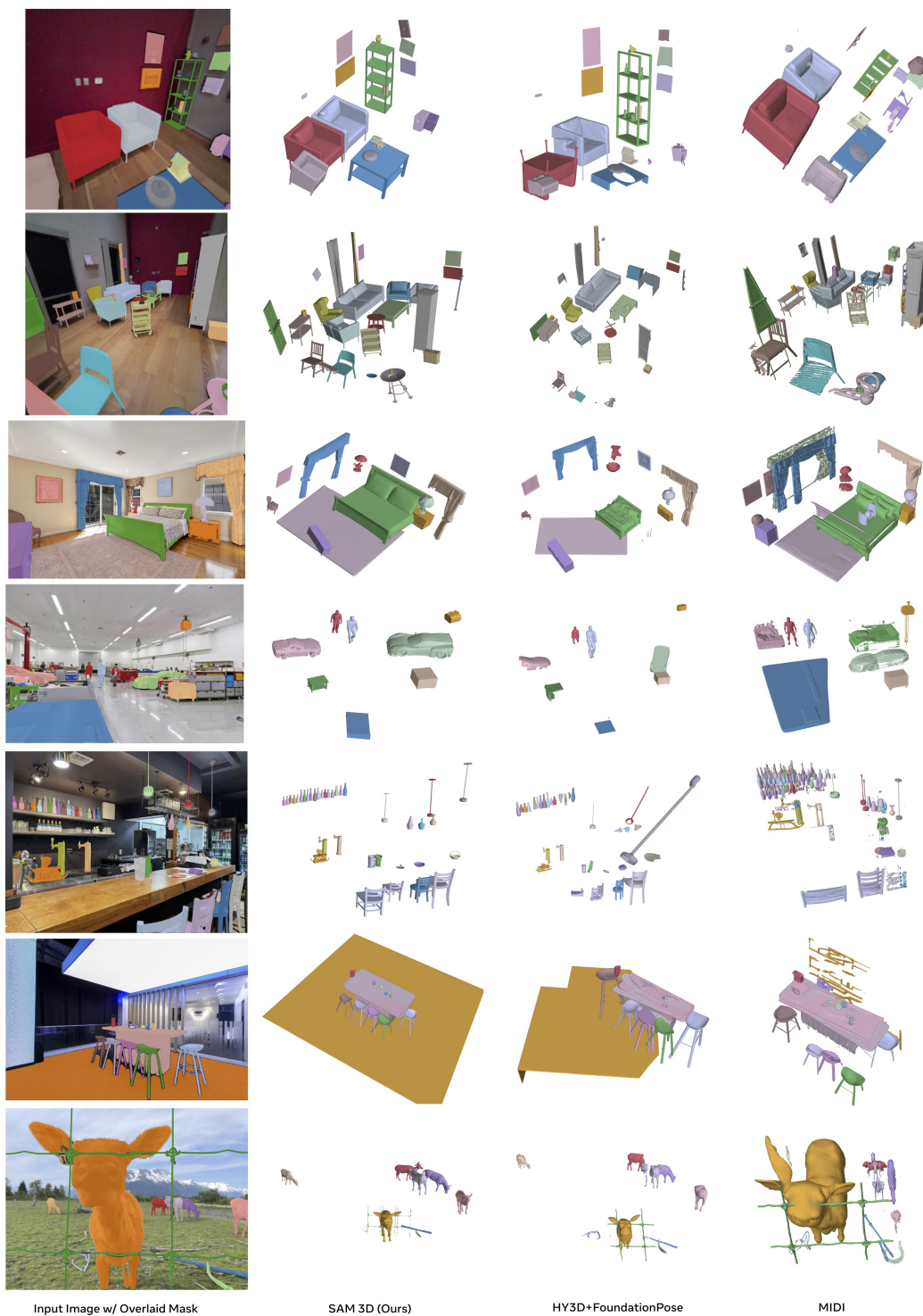


Figure 20 Qualitative examples for scene reconstruction. Showing examples of SAM 3D and alternative scene reconstruction methods.

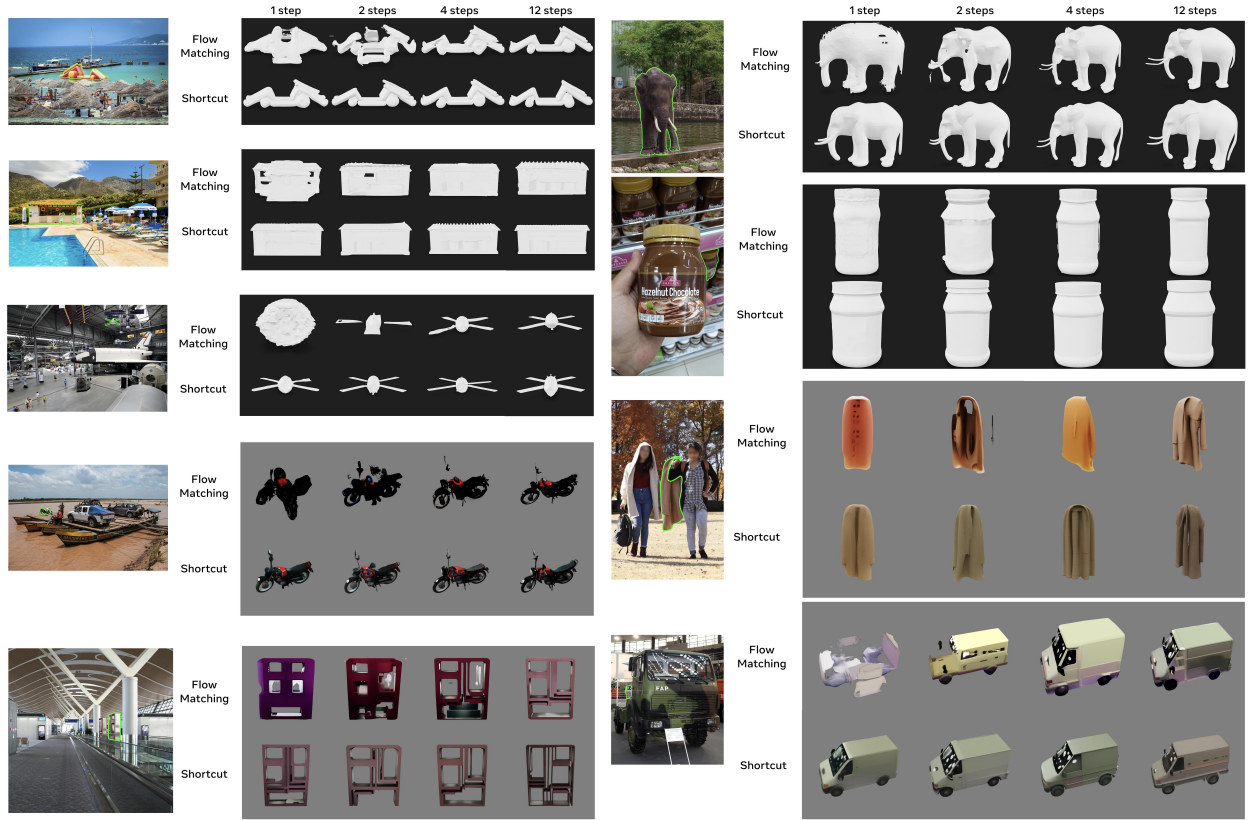


Figure 21 Qualitative examples after distillation. Visualization using difference number of steps in flow matching mode and shortcut mode. The black background displays the mesh rendering without texture, while the grey background shows the rendering of the Gaussian splatting.