Supplemental Material for Cupid: Generative 3D Reconstruction via Joint Object and Pose Modeling

Binbin Huang[†,1,2] Haobin Duan[†,1,2] Yiqun Zhao[1,2] Zibo Zhao[3] Yi Ma[1,2] Shenghua Gao[‡,1,2]

[1]The University of Hong Kong [2]Transcengram [3]Tencent

[†] Equal contribution [‡] Corresponding author

A. Implementation

Dataset. We train our model using several datasets, including ABO [6], HSSD [21], 3D-FUTURE [10], and a subset of Objaverse-XL [7], totaling approximately 260K 3D assets. These artist-curated datasets are predominantly aligned to a canonical frame where the ground plane corresponds to $z = 0$. Following TRELLIS [60], we encode each asset into occupancy grids and structured latents that are suitable for training the flow transformer. The structured latents can be decoded into high-quality triangle meshes or Gaussian splats using the SLat decoders. To enable diverse camera generation, we render 24 conditioning images from random viewpoints for each asset, with augmented focal lengths ranging from 24 mm to 200 mm.

Table 5. **Pose Reconstruction Fidelity.** We evaluate the accuracy of camera poses recovered from the decoded UV volumes via the PnP algorithm. We report the Reprojection Error in normalized pixels, RRE, RTE, and RFov in degrees.

Metric	Reproj. Err.	RRE	RTE	RFov
Mean	0.0004	0.36	0.35	0.10
95% Quantile	0.0006	0.94	0.95	0.19

Pose Encoding and Recovery. Following the methodology outlined in the main text, we represent the camera pose and coarse scene geometry as a voxelized 3D occupancy grid paired with a UV coordinate volume, denoted as $\{\mathbf{x}_i, \mathbf{u}_i(\boldsymbol{\theta})\}_{i=1}^{L}$. To facilitate efficient learning, we employ a 3D VAE to compress the input UV volume (resolution 64^3) into a compact latent representation (resolution 16^3). These pose latents are concatenated with occupancy embeddings to condition the training of the structural flow model, $\mathcal{G}_\mathbf{S}$. For pose recovery, the latents generated by $\mathcal{G}_\mathbf{S}$ are decoded back into the UV volume. We then estimate the camera extrinsics $[\mathbf{R}|\mathbf{t}]$ and intrinsics $\mathbf{K}$ using the Perspective-n-Point (PnP) algorithm in Algo. 1. We evaluate the fidelity of this reconstruction in Tab. 5, reporting the mean and 95th percentile for Reprojection Error in normalized pixel coordinates, Relative Translation Error (RTE), Relative Rotation Error (RRE), and Relative Field-of-View (RFov) error in degrees. With the solved camera pose, we finally train the latent flow model $\mathcal{G}_\mathbf{L}$ to synthesize the structured features $\{\mathbf{f}_i\}_{i=1}^{L}$ conditioned on the occupancy and the pose-aligned visual features.

Algorithm 1 Perspective-n-Point solver via Direct Linear Transformation (DLT)

$\mathcal{X}$: Sets of 3D coordinates $\{\mathbf{x}_i\}_{i=1}^{N}$
$\mathcal{U}$: 2D coordinates $\{\mathbf{u}_i\}_{i=1}^{N}$

$N \leftarrow |\mathcal{X}|$ ▷ Number of correspondences
$\mathbf{A} \leftarrow \mathbf{0}_{2N\times 12}$ ▷ Initialize coefficient matrix
for $i = 1$ to N **do**
  $\tilde{\mathbf{x}}_i \leftarrow [\mathbf{x}_i^\top, 1]^\top$ ▷ Homogeneous coordinates
  $u_i, v_i \leftarrow \mathbf{u}_i$
  $\mathbf{A}[2i-1, :] \leftarrow [\tilde{\mathbf{x}}_i^\top, \mathbf{0}^\top, -u_i\tilde{\mathbf{x}}_i^\top]$
  $\mathbf{A}[2i, :] \leftarrow [\mathbf{0}^\top, \tilde{\mathbf{x}}_i^\top, -v_i\tilde{\mathbf{x}}_i^\top]$
end for
$[\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}^\top] \leftarrow \text{SVD}(\mathbf{A})$
$\mathbf{p} \leftarrow \mathbf{V}[:, -1]$ ▷ Solution is the last column of $\mathbf{V}$
$\mathbf{P} \leftarrow \text{reshape}(\mathbf{p}, 3, 4)$ ▷ Camera matrix $\mathbf{P} \in \mathbb{R}^{3\times 4}$
if $\det(\mathbf{P}_{1:3,1:3}) < 0$ **then** ▷ Fix scale/sign ambiguity
  $\mathbf{P} \leftarrow -\mathbf{P}$
end if
$[\mathbf{K}, \mathbf{R}, \mathbf{t}] \leftarrow \text{RQ}(\mathbf{P})$ ▷ Decompose camera matrix
return $\mathbf{K}, \mathbf{R}, \mathbf{t}$

Training Details. We initialize our models using the pre-trained weights of TRELLIS. During training, we apply classifier-free guidance [14] (CFG) with a drop rate of 0.1. Both $\mathcal{G}_\mathbf{S}$ and $\mathcal{G}_\mathbf{L}$ are trained using AdamW [30] at a fixed learning rate of 1×10^{-4} for 500k and 100k steps, respectively. Training completes in approximately one week on 32 GPUs. At inference, we use 25 sampling steps with classifier-free guidance strengths of 7.5 and 3.0 for $\mathcal{G}_\mathbf{S}$ and $\mathcal{G}_\mathbf{L}$, respectively.

Table 6. **NVS consistency with ICP alignment.** Evaluated on a subset of Toys4k.

Methods	PSNR↑	SSIM↑	LPIPS↓	mIOU↑	CD (avg)↓	F-score (0.05)↑
HY3D	18.81	84.71	0.2769	77.23	43.41	0.6141
TRELLIS	20.94	86.48	0.2479	81.42	30.95	0.6901
Ours	21.84	87.44	0.2387	83.75	21.18	0.7435

Table 7. **Input view metrics with external alignment.** Our full model with predicted poses outperforms baselines that rely on external alignment tools (VGGT or ICP).

Methods	PSNR↑	SSIM↑	LPIPS↓	mIOU↑	CD (avg)↓	F-score (0.05)↑
TRELLIS (VGGT)	17.43	86.96	0.1360	37.92	159.9	0.4618
HY3D (ICP)	22.17	92.63	0.0756	79.44	2.619	0.9173
TRELLIS (ICP)	24.42	93.81	0.0524	83.54	1.744	0.9541
Ours (ICP)	25.95	94.67	0.0431	86.71	0.792	0.9721
Ours (Full)	29.91	96.69	0.0252	92.46	0.651	0.9805

B. 3D Geometry and View Consistency

To validate the 3D fidelity of our generated assets, we conduct rigorous pixel-level evaluations (depth and appearance) on novel views using a subset of the Toys4k dataset, comparing against recent state-of-the-art 3D generators including TRELLIS [60] and HunYuan3D-2.0 (HY3D) [74], where we use its abedo as color. To ensure absolute fairness, all generated meshes are carefully aligned to the ground truth using ICP (4k points). As shown in Table 6, our method achieves the highest performance across all metrics on novel views. This confirms that our performance gains stem from fundamentally improved 3D geometry and fidelity, rather than merely overfitting to the input view.

Furthermore, a primary goal of our framework is to enable the seamless compositing of generated 3D objects back into input images. While global 3D metrics are important, pixel-level input view consistency is critical for this application. Existing 3D generative models often hallucinate details that deviate from the input pixels, making external registration (e.g., VGGT [49] via image registration, or ICP even with ground truth meshes) highly challenging.

As shown in Table 7, our joint modeling approach significantly bypasses these difficulties, ensuring precise pixel-level consistency. Even when our pose advantage is removed (by aligning all methods via ICP), our method still generates superior and more consistent geometry.

C. Additional Ablations on Pose Modeling and Conditioning

Impact of Decoupled Camera Poses. To demonstrate the necessity of explicitly modeling camera poses, we implemented a view-aligned baseline (fixed camera, rotated objects) trained on the ABO dataset. Note that this baseline inherently necessitates a fixed Field of View (FoV). As shown in Table 8 and Figure 8, our method (also trained with the ABO fixed FoV) achieves vastly superior geometric alignment. View-centric baselines suffer from degraded consistency due to the increased complexity of the latent space, as the model is forced to represent the same object in infinite rotated configurations. Explicitly predicting camera poses decouples geometry from viewpoint, ensuring higher fidelity without assuming fixed intrinsics.

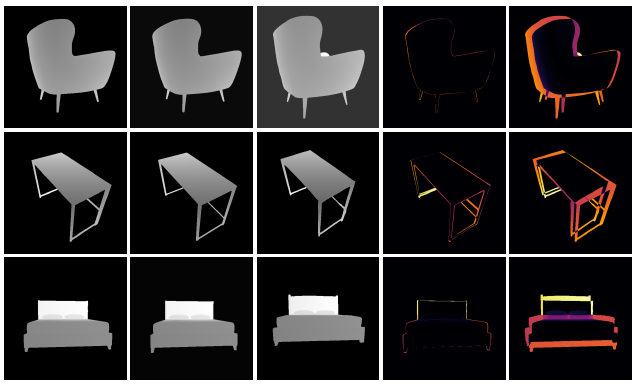

Figure 8. **Comparison of our decouple camera poses vs. fixed camera.** Explicitly predicting camera poses yields significantly higher geometric fidelity.

Table 8. **Effectiveness of predicting decoupled camera poses.** Evaluated on the ABO dataset.

Method	CD (avg)↓	CD (med)↓	F-score (0.05)↑
Fixed Camera	6.050	1.745	0.8371
Ours	0.864	0.102	0.9824

Pose-Aware Conditioning. Our Pose-Aware Conditioning (PAC) is the prerequisite for effectively utilizing low-level features. To validate that visual features are ineffective without spatial anchoring, we conduct two experiments (Table 9): 1) **Broken alignment:** We randomly perturb the pose used in PAC by adding noise $(3°, 6°, 9°)$. This causes a sharp performance drop proportional to the noise level. 2) **Features without alignment:** We retrain the model with low-level features injected as global (pose-agnostic) tokens, which yields negligible improvement over the baseline. These results prove that PAC provides the necessary spatial correspondence to inject features correctly.

Table 9. **Effectiveness of PAC.** Perturbing the pose used in PAC degrades performance, proving spatial anchoring is required.

	GT			Sampled		
Model Name	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Ours (Full)	34.86	98.24	0.017	30.05	96.81	0.025
- Pose Noise 3°	33.29	97.83	0.019	29.27	96.47	0.027
- Pose Noise 6°	31.60	97.27	0.023	28.43	96.09	0.031
- Pose Noise 9°	30.19	96.76	0.029	27.63	95.72	0.035
w/o PAC	32.81	97.67	0.020	27.84	95.74	0.032

D. Occlusion-aware conditioning

To handle occlusion in complex scene reconstruction, our model leverages partial 3D object observations as conditions to generate complete objects. Our model takes a 2D occlusion mask $\mathbf{M}^{\text{occ}} \in \{0,1\}^{H\times W}$ as input alongside the visible object observation $\mathbf{I}^{\text{cond}}$. The mask $\mathbf{M}^{\text{occ}}$ identifies pixels that may belong to the object if occluders were removed, with values set to zero when no occlusion is present. Together, $\mathbf{M}^{\text{occ}}$ and the alpha channel of $\mathbf{I}^{\text{cond}}$ identify three pixel classes: (a) directly observed object pixels, (b) background pixels that must not contain the object, and (c) occluded pixels that may or may not contain the object.

We apply two modifications to both flow transformers to incorporate the mask. First, inspired by Amodal3R [59], we modulate the attention weight matrix during global condition injection via cross-attention using the mask. Specifically, the attention weight for each input token is computed by patching the mask to match the DINOv2 tokens and calculating the ratio of unmasked pixels in each patch. The logarithm of the weight values is added to the attention logits before applying the softmax operation. Second, for the geometry and appearance flow model that takes additional pose-aligned features, we concatenate the input image with the mask as an additional channel before feeding it into the convolution layer in the second stage.

During training, we randomly generate occlusion masks $\mathbf{M}^{\text{occ}}$ following Amodal3R and zero out the corresponding regions in $\mathbf{I}^{\text{cond}}$, preventing information leakage on occlusion regions and encouraging the model to reconstruct complete 3D objects from partial observations. At inference time, occlusion masks can be obtained heuristically or manually for scene reconstruction.

E. Generative reconstruction diversity.

Reconstruction methods like LRM [16] typically generate a single 3D object from one image. However, our approach not only surpasses LRM in terms of reconstruction quality but also produces multiple plausible interpretations—what we refer to as generative reconstruction. Since standard metrics do not effectively capture this diversity, we offer qualitative comparisons in Figure 9. The results demonstrate that our framework creates diverse 3D models with visible regions that consistently align with the input image. In contrast, conventional 3D generation methods [60] often struggle to maintain consistency in the visible regions of the objects they create. Furthermore, while directly processing an image with multiple objects may still yield a scene, the generation quality is often degraded as such inputs are out-of-distribution. In contrast, our method allows for component-aligned compositional reconstruction, as demonstrated by the additional examples in Figure 11.

Qualitative comparison with generation method. Generation method like TRELLIS [60] does not provide an explicit object-centric camera pose for the input image and post-hoc alignment is unreliable. Here we focus on qualitative comparisons in Figure 10. For both methods, we render a canonical object from front and back views. As shown, TRELLIS struggles to maintain texture consistency with the input image, exhibiting notable color drifting on the candle statue, teddy bear, and the lamb. In contrast, our method closely preserves the appearance of the input. Importantly, although we incorporate pixel-level cues via back-projection, the model does not simply copy pixels: the generated back views remain coherent with the front, indicating learned view-consistent geometry and appearance. These results demonstrate that our mechanism effectively mitigates color drifting and texture inconsistency. We hypothesize that this limitation is common among 3D generators that lack localized, pixel-attended conditioning, including follow-up works of TRELLIS. We hope these findings can inspire future designs of 3D generators.

Figure 9. **Diversity of our *Generative Reconstruction*.** We visualize canonical front and back views of generated 3D objects using random seeds (1-4). Given a single input image, our model synthesizes diverse hypotheses for unobserved regions while remaining highly consistent with visible regions. In contrast, the base 3D generator [60] struggles to conform to the input image, or produce less diverse unobserved regions.

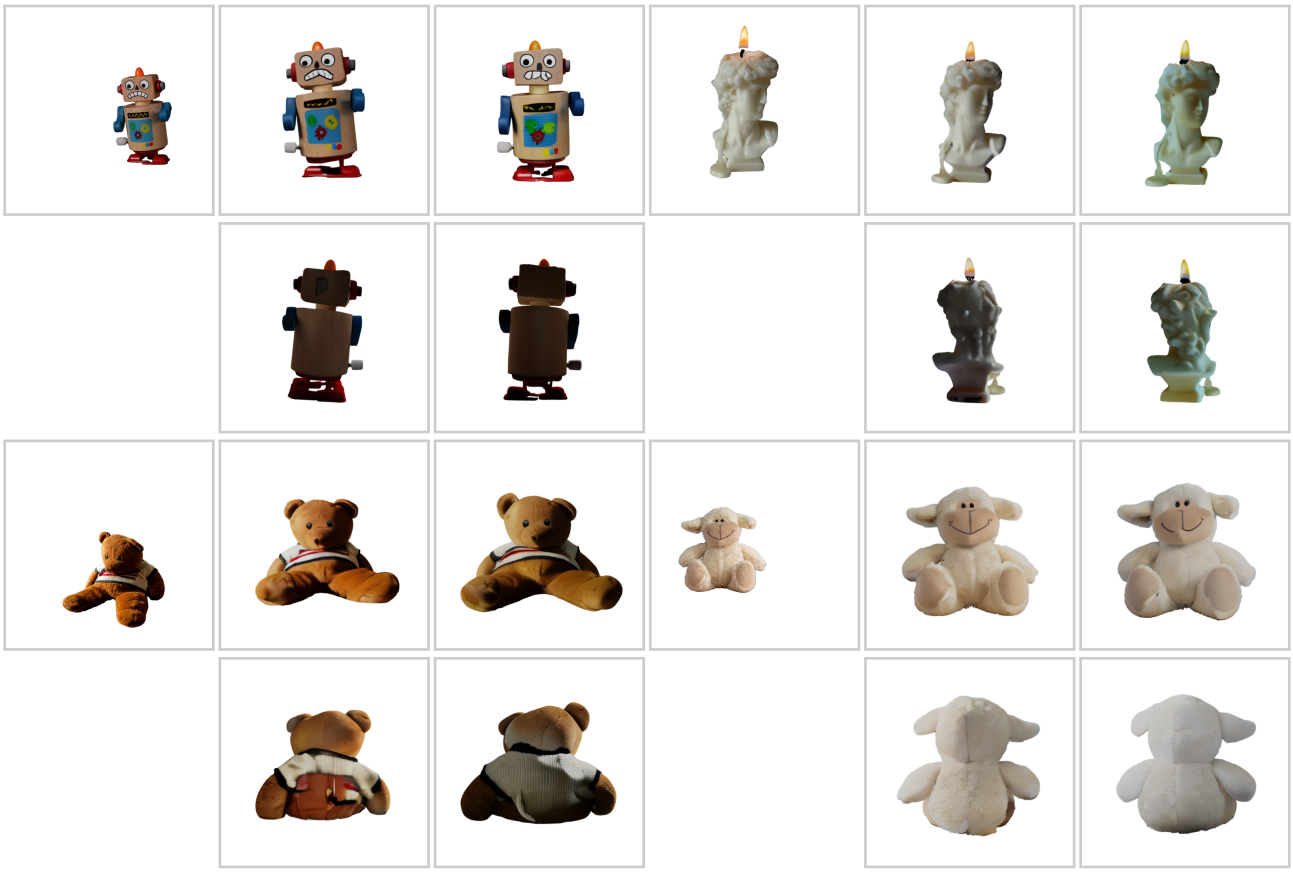

Figure 10. **Qualitative comparison: *Generative Reconstruction* vs. *3D Generation*.** Unlike standard 3D generation, which aims to create novel objects from images, our generative reconstruction is specifically designed to accurately recreate a particular object that replicates the visible regions of the input image while maintaining diversity in the invisible regions. This difference in objectives is crucial, as prior 3D generators [60] are not optimized for this task and often produce artifacts such as *color drift* and *texture inconsistencies*. As demonstrated in the canonical front and back views, our method's pose-aligned image conditioning effectively reduces these issues, resulting in a reconstruction that remains true to the input.

Figure 11. **Additional examples of component-aligned scene reconstruction.** For each example shown, the panels display: (top left) the input image, (top right or bottom left) the final rendered output, and (bottom) the reconstructed individual components, color-coded for clarity.