StructBridge: Structure-Grounded 3D Indoor Object Generation via 3D Latent Diffusion Bridge and Normal Refinement

Nange Wang[1]†, Zhanxu Bai[1]†, Yuxuan Li[3], Cewu Lu[1,2], Jianhua Sun[1]§
[1]Shanghai Jiao Tong University, [2]Shanghai Innovation Institute, [3]Noematrix

Abstract

Generating 3D indoor objects under fine-grained structural control is crucial and has broad applications in robotics, gaming, and simulation. Such fine-grained structural control encompasses an object's spatial layout, geometric icons, and relative size, which together constitute the object's core structural properties. However, existing controllable 3D generation methods do not take structural control as a primary objective, and they commonly rely on suboptimal conditions in representing structure or inadequate algorithms for maintaining structural consistency. Therefore, they fail to achieve fine-grained structural control during generation. To address this, we present **structure-grounded 3D object generation**, a novel perspective on 3D object generation, which decomposes 3D shapes into structural configurations and geometric details, directly conditions on the object's structural configurations and synthesizes realistic geometric details to produce high-quality 3D objects. Building on this perspective, we introduce StructBridge, which first employs a diffusion bridge in the 3D latent space to transform comprehensive structural conditions into structurally accurate coarse meshes, followed by a structure-preserving normal refiner that enriches geometric details while preserving structure. We conduct extensive experiments and comprehensive comparisons with various conditional generation methods, demonstrating that StructBridge achieves state-of-the-art performance in both generation quality and structural control capabilities. The official implementation of StructBridge is available at `https://github.com/wngnbr/structbridge`.

1 Introduction

Recently, 3D object generation has witnessed rapid progress, giving rise to diverse paradigms such as SDS-based methods [1–3], multi-view based methods [4–6], and 3D native generation methods [7–9]. These advances have effectively supported text- and image-conditioned 3D generation. Meanwhile, controllability has become an important focus in 3D generation, and many studies introduce additional forms of conditioning, such as sketches [10], skeletons [11], or coarse geometries [12, 13], to control object properties like silhouette, pose or overall shape. Beyond these developments, generating 3D objects under fine-grained structural control, especially for indoor objects, is of critical importance, with broad practical value across diverse domains such as virtual content creation, industrial design, AR/VR applications, and robotics [14, 15]. This level of control is necessary, because an object's important properties such as its category, semantics, and functionality are fundamentally defined by its core structural properties [16, 17], including spatial layout, geometric icons [18], and relative size.

However, existing controllable 3D generation methods still have limitations in achieving fine-grained structural control. Specifically, the conditions commonly used in controllable 3D generation, such

[†]denote equal contribution, [§] denotes corresponding author

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

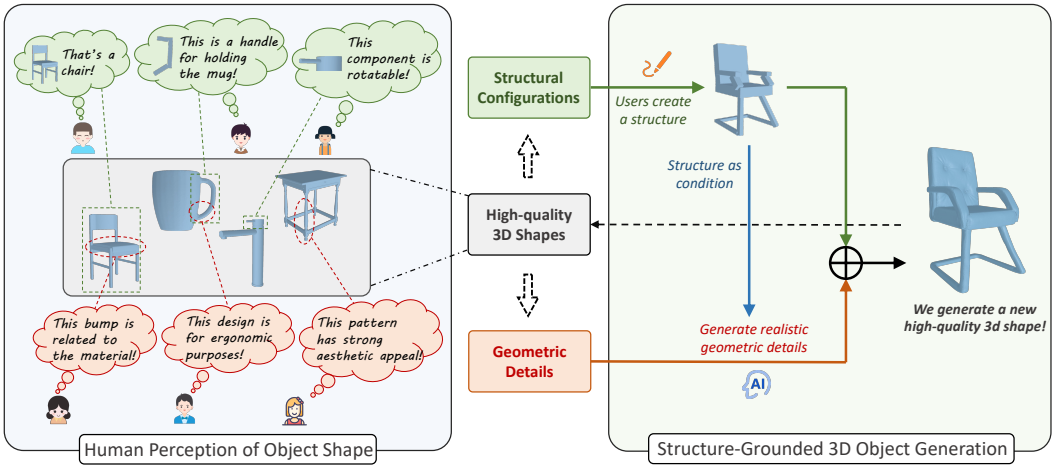

Figure 1: Design philosophy of structure-grounded 3D object generation. In the left side of the figure, we present several examples illustrating human perception of object shape, based on which we decompose 3D shapes into structural configurations and geometric details. Based on this decomposition, we design a structure-grounded 3D object generation pipeline, as shown on the right side of the figure.

as text [2, 3], images [6, 9], or sketches [19, 10], have limited expressive capability in representing the structure of 3D objects due to their modality limitations. As a result, these conditions inevitably lose critical structural information, and methods relying on them cannot achieve precise structural control. On the other hand, some methods [12, 20, 1] incorporate more expressive 3D shapes as conditions. However, these 3D shapes are only used as coarse geometric guidance rather than an accurate representation of object structure, and the algorithms employed in these methods do not explicitly focus on the structural information contained within the 3D shapes, resulting in limited ability to maintain structural consistency and poor performance on complex structures.

To address these limitations, we present **structure-grounded 3D object generation**, a controllable 3D object generation perspective that is grounded on and fully leverages the object's core structural properties. Structure-grounded 3D object generation decomposes 3D shapes into structural configurations and geometric details, conditions on the structural configurations and synthesizes realistic geometric details to produce high-quality 3D assets, as illustrated in Fig. 1. Following this idea, we first adopt an effective 3D representation, referred to as StructMap, which encodes an object's structural configurations through geometric primitives and corresponding parameters. Then we introduce **StructBridge**, a novel generative framework that takes StructMap as input and produces high-fidelity 3D objects that are both structurally consistent with StructMap and enriched with rich geometric details, consisting of two modules: 1) a 3D latent diffusion bridge that transforms StructMap into structurally accurate coarse meshes; 2) a structure-preserving normal refiner that incorporates fine-grained geometric details into the coarse mesh while preventing structural distortion.

We conduct extensive experiments to validate the effectiveness of our method by comparing it with other controllable 3D generation methods under different condition types and evaluating performance in terms of generation quality and structural consistency. Results demonstrate that our method significantly outperforms existing methods in terms of generation quality and structural consistency.

In summary, this work makes the following key contributions: 1) We explore **structure-grounded 3D object generation**, a novel controllable 3D generation perspective that is grounded on and fully leverages the object's core structural properties during generation, including spatial layout, geometric icons, and relative size. 2) We introduce **StructBridge** to generate high-quality 3D indoor objects under fine-grained structural control, consisting of two core modules: i) a **3D latent diffusion bridge** that establishes a smooth transition from 3D structural representations to structurally accurate coarse meshes, and ii) a **structure-preserving normal refiner** that enriches the coarse meshes with fine-grained, diverse, and semantically plausible geometric details while preventing structural distortions. 3) We conduct extensive experiments to validate the effectiveness of our framework, demonstrating significant improvements in generation quality and structural consistency.

2 Related Work

2.1 Conditional 3D Generation

3D generation has advanced rapidly in recent years, giving rise to a wide range of condition types for controlling the generation process. Most works are predominantly conditioned on text and images, upon which various 3D generation paradigms emerged, such as score-distillation sampling (SDS) based methods [1–3], multi-view (MV) based methods [4–6], and 3D native generation methods [7–9]. Building on this foundation, many works introduce additional forms of conditions to improve the controllability of 3D generation. Sketch-conditioned methods [10, 21] perform 3D object generation from sketches, and other approaches [11, 22, 23] incorporate skeleton or pose control signals. StrucADT [24] controls point-cloud generation through a graph specifying part existence and adjacency relationships. In addition, some methods [12, 20, 1] incorporate expressive 3D shapes as conditions, and CLAY [7] explores the integration of point cloud, bounding box, and voxel conditions into generative models. However, these condition types lack the ability to precisely characterize an object's structure, while more expressive conditions such as 3D shapes or voxels are typically used only as coarse geometric guidance rather than accurate representations of object structure. As a result, existing methods based on these conditions still face challenges in achieving fine-grained structural control during generation. Our focus is on explicitly specifying spatial layout, geometric icons, and relative size through a structural configuration that guides generation to preserve these attributes while synthesizing high-quality geometric details.

2.2 3D Native Generative Methods

3D native generative methods have shown strong capability by directly learning from 3D representations such as point clouds [25, 26], meshes [27] and implicit fields [28, 29], which benefit from the rapid progress of diffusion models [30, 31]. However, training diffusion models directly on raw 3D data is computationally expensive. To address this, recent works adopt latent diffusion frameworks [30] that operate in latent spaces. Michelangelo [32] and CLAY [7] train on latent set representations, while Direct3D [33] employs triplane features for scalability. CraftsMan [8] further enhances generalization through multi-view diffusion priors, and Hunyuan3D 2.1 [9] introduces vector-set representations of implicit functions to improve efficiency and shape fidelity. Beyond diffusion-based approaches, MeshArt [34] employs hierarchical autoregressive transformers to first generate articulation-aware object structures, represented by bounding primitives and articulation modes, and then synthesize part meshes conditioned on these structures. Our approach first integrates an explicit 3D structural representation into the diffusion bridge, directly modeling the transition from a prescribed structural configuration to detailed object geometry.

3 Structure-Grounded 3D Object Generation

Despite recent progress in controllable 3D generation, existing methods commonly fail to achieve fine-grained structural control during generation. To address this, we propose structure-grounded 3D object generation, a novel perspective for controllable 3D object generation.

Design Philosophy. Following cognitive science theories and prior work on 3D objects [35, 36, 18], we view a 3D object shape as comprising two components: structural configuration and geometric details. For static, predominantly rigid indoor objects, we characterize object structure through three core properties: the coarse geometry of individual parts (termed geometric icons), their spatial layout, and their relative sizes. Together, the specific instantiation of these properties define an object's structural configuration, while geometric details are fine-grained appearance characteristics built upon the structural configuration. Fig. 1 illustrates this decomposition. This view motivates us to use the structural configuration as an explicit generative condition and synthesize geometric details while preserving the specified structure.

Structural Reprensentation. For the structure-grounded 3D object generation, we first need a structural representation that can accurately and comprehensively describe the structural configuration of an object while remaining intuitive and easy for creation. We observe that the representation proposed by ConceptFactory [35], which conceptualizes objects through assemblies of parameterized and composable geometric primitives, can effectively characterize object structure. Therefore, we develop an effective structural condition based on this representation, and refer to it as StructMap in

our task. In general, a StructMap is represented as a primitive set $\{P_i(g_i, \theta_i^{\text{shape}}, \theta_i^{\text{pos}}, \theta_i^{\text{rot}})\}_{i=1}^{M}$, where $g_i \in \mathcal{G}$ is a type parameter selected from a predefined primitive library $\mathcal{G}$, and θ_i^* are parameters to control the shape and pose of geometric primitives. Under this definition, StructMap can precisely encode a wide range of structural properties. Specifically, the spatial layout of the object is defined by the pose parameters θ_i^{pos} and θ_i^{rot} of each part, and the geometric icons and relative size are determined by the geometric type parameter g_i and the shape parameters θ_i^{shape}. In addition, building upon ConceptFactory [35], we develop a StructMap modeling interface that enables users to intuitively and conveniently create StructMaps from scratch. *A detailed illustration of the StructMap modeling interface and statistics on the average time required to create StructMaps is provided in Appendix B.*

Generation Objective. After obtaining a StructMap, our goal is to enrich it with realistic geometric details while preserving the structural information it encodes, as illustrated in Fig. 2. We refer to the ability to explicitly specify such a configuration and guide generation to preserve it as explicit structural control. We observe that the geometric gap between a StructMap and a high-quality 3D object contains two different levels of variations. The first involves global geometric variations, such as surface completion and smooth transitions between adjacent primitives. These variations exhibit relatively regular distributions and are therefore suitable for generative modeling. The second involves localized fine-grained geometric details. Compared with global geometry variations, these details exhibit greater diversity and finer geometric granularity. Since they usually correspond to localized geometric modifications, such details are more suitable for direct appearance-guided geometric refinement.

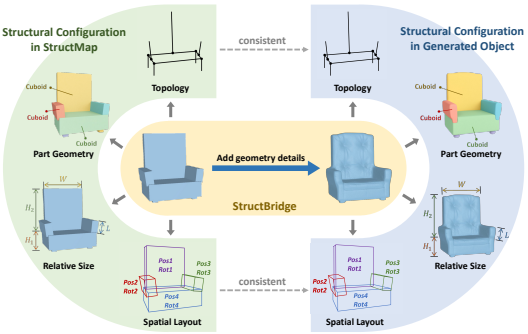

Figure 2: Illustration of the structural configuration during the generation process. The left side shows the structural configurations in StructMap, while the right side presents the corresponding configurations preserved in the generated 3D object. The structural configuration remains consistent throughout the generation process.

4 Method

4.1 Overview

To achieve our generation objective, we develop a novel generative framework, StructBridge, which fully leverages the structural information encoded in the StructMaps and transforms StructMaps into high-quality 3D objects with realistic geometric details. StructBridge consists of two core modules: a 3D latent diffusion bridge and a structure-preserving normal refiner. The 3D latent diffusion bridge establishes a smooth transition from 3D structural representations to structurally accurate coarse meshes. And the structure-preserving normal refiner further enriches the coarse meshes with fine-grained geometric details while preventing structural distortions.

Specifically, given a StructMap S, we first convert it into a normalized point cloud $\mathcal{P}_S$, and then encode it into a latent feature $z_T = \mathcal{E}_{\text{VAE}}(\mathcal{P}_S)$ using a 3D VAE [37]. Starting from z_T, we employ the 3D latent diffusion bridge to gradually inject geometric details, obtaining a latent feature z_0 that lies on the manifold of latent features corresponding to coarse shapes with basic geometric details, such as smoothing between components. Then, we use the 3D VAE decoder to decode z_0 into a coarse mesh $\hat{O} = \mathcal{D}_{\text{VAE}}(z_0)$. Given the coarse mesh $\hat{O}$ and an optional detail style image I_d, the structure-preserving normal refiner further enriches the mesh with fine-grained geometric details while preserving the overall structural configuration. The overall pipeline of StructBridge is illustrated in Fig. 3.

4.2 Data Preprocessing

Since a StructMap is composed of multiple geometric primitives, its raw mesh often contains redundant internal surfaces, which introduce unnecessary geometric cues during generation, and may

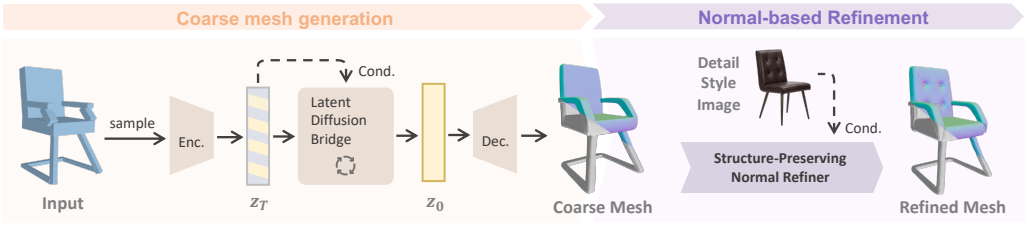

Figure 3: Illustration of the StructBridge pipeline. StructBridge first converts a StructMap into a coarse mesh using a 3D Latent Diffusion Bridge, and then refines the coarse mesh into a high-quality 3D object with rich geometric details through the Structure-Preserving Normal Refiner.

confuse the model. So we first apply boolean union operations to merge all primitives and remove unnecessary internal faces, preserving only the essential outer geometry and obtaining a cleaner structural representation. Then, as watertightness is crucial for 3D native generative models [8, 7], we enforce mesh watertightness following the preprocessing procedure of [8]. Finally, to ensure consistent scale across different StructMaps, all meshes are normalized to fit within a cubic space of $[-1, 1]^3$, and the corresponding parameters of each StructMap are adjusted accordingly.

4.3 3D Latent Diffusion Bridge

After obtaining the preprocessed StructMaps, we design a 3D latent diffusion bridge to transport the distribution of StructMaps toward that of 3D objects with geometric details. The overall pipeline of the 3D Latent Diffusion Bridge is illustrated in Fig. 4. Specifically, for a StructMap mesh S and its corresponding high-quality object mesh O, we sample input point clouds $\mathcal{P}_S$ and $\mathcal{P}_O$ and encode them into latent representations using a shared 3D-VAE [37]. Within its encoder, farthest-point sampling first selects N query points from each input's point groups, with coordinates denoted by $Q_S = \{q_S^i\}_{i=1}^N$ and $Q_O = \{q_O^i\}_{i=1}^N$. These queries aggregate information from the input point features through attention, producing latent representations z_T and z_0 in $\mathbb{R}^{N\times d}$, where N is the number of latent tokens and d is their feature dimension. Each latent token retains the index of its corresponding query, providing a spatial reference for the subsequent token alignment. The diffusion bridge process can be modeled as the solution to the following SDE [38]:

$$dz_t = f(z_t,t)\,dt + g(t)\,dw_t, z_0 = \mathcal{E}_{\text{VAE}}(\mathcal{P}_O), z_T = \mathcal{E}_{\text{VAE}}(\mathcal{P}_S) \tag{1}$$

Why Diffusion Bridge? Most 3D native generative methods have adopted diffusion models as their core generation algorithm. However, due to our specific conditioning form and generation objective, we consider the diffusion bridge to be more suitable for our task. First, a diffusion bridge explicitly models a stochastic path that is mathematically constrained by both endpoints, enabling the intermediate states z_t to follow a closed-form, tractable conditional distribution $q(z_t \mid z_0, z_T)$. This provides a theoretically grounded training objective and ensures that the generation process is fully consistent with the structural conditions. In contrast, heuristic conditioning used in conventional diffusion models does not model endpoint-constrained dynamics, lacks theoretical guaranties, and often leads to mismatches between the condition and the diffusion trajectory [38]. Second, since StructMap is represented in mesh form, using it as the starting point of the diffusion bridge reduces the difficulty of mesh generation and improves the resulting quality, while conventional diffusion models must generate a high-quality 3D shape starting from pure noise.

Latent Alignment Mechanism. Diffusion bridges have been primarily applied in the image domain, where the diffusion processes operate in pixel space. However, when extended to 3D latent space, several easily overlooked yet important issues arise. Specifically, the latent tokens produced by the 3D-VAE [37, 7, 39] are indexed by their query points, but independently encoded shapes do not share a canonical query ordering. Consequently, the i-th token in z_0 may not correspond spatially to the i-th token in z_T, making direct token-wise comparison between the endpoints problematic.

To address this problem, we introduce a latent alignment mechanism based on Earth Mover's Distance [40] (EMD). Instead of establishing correspondences independently at each time step, we compute a global alignment between the source and target query sets Q_S and Q_O. Specifically, we

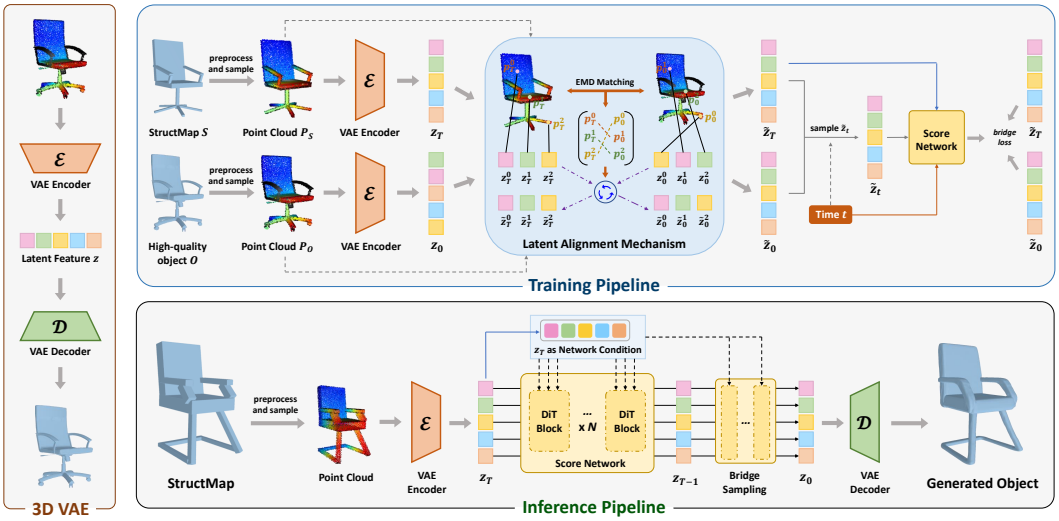

Figure 4: Illustration of the 3D Latent Diffusion Bridge. It first employs a 3D-VAE to encode point clouds sampled from meshes into a latent space, based on which we develop an inference pipeline.

solve for an optimal transport plan $\pi^* \in \Pi(Q_S, Q_O)$ as

$$\pi^* = \arg \min_{\pi \in \Pi(Q_S, Q_O)} \sum_{i,j} \pi_{ij} \left\| q_S^i - q_O^j \right\|_2 \tag{2}$$

Here, $\Pi(Q_S, Q_O)$ denotes the set of binary permutation matrices with exactly one nonzero entry in each row and column. An entry $\pi_{ij} = 1$ matches StructMap query i to target-object query j.

Taking the StructMap token order as the reference, we reorder only the target-object tokens, yielding $\tilde{z}_0 = \pi^* z_0$ and $\tilde{z}_T = z_T$. Intermediate bridge states $\tilde{z}_t$ are then sampled between these aligned endpoints, retaining the same token indexing without further matching at individual time steps.

Training. The networks in StructBridge are trained on paired data (O, S), where O represents a 3D object and S is its corresponding StructMap that encodes the object's structural configuration, and we directly utilize the 3D objects and their conceptual annotations from ConceptFactory [35] as our training pairs. Each mesh in our training pairs is also preprocessed and sampled to a point cloud, and notably, to avoid subtle misalignments introduced by geometric details during normalization, both the StructMap and the corresponding 3D object are normalized using the translation and scaling parameters of the StructMap. During training, both $\mathcal{P}_O$ and $\mathcal{P}_S$ are encoded by the 3D-VAE encoder to obtain their latent representations $(z_0, z_T) = (\mathcal{E}_{\text{VAE}}(\mathcal{P}_O), \mathcal{E}_{\text{VAE}}(\mathcal{P}_S))$. Next, we apply the alignment mechanism introduced above to obtain $\tilde{z}_0 = \pi^* z_0$ and $\tilde{z}_T = z_T$. Finally, we optimize the diffusion bridge model s_θ using the bridge loss [38]:

$$\mathcal{L}(\theta) = \mathbb{E}\big[\|s_\theta(\tilde{z}_t, t, z_T) - \nabla_{\tilde{z}_t} \log q(\tilde{z}_t | \tilde{z}_0, \tilde{z}_T)\|^2\big], \tag{3}$$

4.4 Structure-Preserving Normal Refiner

To further enrich the generated meshes with semantically meaningful fine-grained details, we introduce a Structure-Preserving Normal Refiner, as illustrated in Fig. 5.

Reference Normal Map Generation. Given a coarse mesh, we first render multi-view normal maps and use a multimodal large language model (MLLM) to enhance them with category-appropriate geometric details. An optional detail style image provides guidance for decorative patterns. The rendered views are arranged into a single atlas and processed jointly, with prompts requesting preservation of the original silhouettes, part layout, and view ordering, as well as consistency of details across views.

However, the generated RGB values may not correspond to unit-length normals. We therefore apply *Unit Sphere Projection* after decoding the normal-map colors. Specifically, for a foreground pixel with RGB value $\mathbf{c} \in [0, 1]^3$, the decoded and normalized camera-space normal is

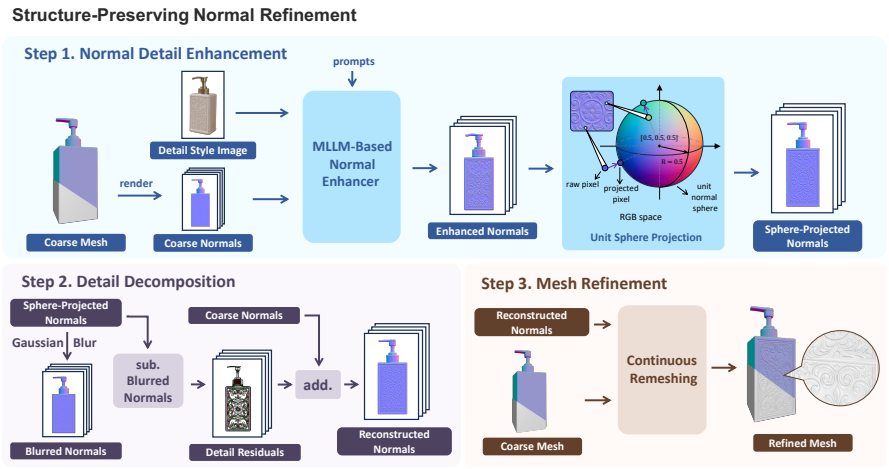

Figure 5: Illustration of the Structure-Preserving Normal Refiner.

$$\hat{\mathbf{n}}_{\text{cam}} = \mathcal{N}(2\mathbf{c} - \mathbf{1}), \qquad \mathcal{N}(\mathbf{v}) = \frac{\mathbf{v}}{\max(\|\mathbf{v}\|_2, \epsilon)}, \tag{4}$$

where ϵ provides numerical stability near zero. The normals are then transformed into a shared world coordinate system using the known camera rotations.

Frequency-aware Normal Extraction. Although the generated normal maps contain rich geometric details, the stochastic generation process of the MLLM may also introduce undesired structural variations and local noise. We therefore extract local high-frequency residuals from the enhanced maps and combine them with the original rendered normals, retaining the coarse mesh as the geometric reference. After foreground alignment and masking, the normalized world-space normals are encoded as RGB and bilateral-filtered to suppress local noise. Let $\mathbf{C}_b^v$ denote the filtered enhanced map for view v, and $\mathbf{C}_0^v$ the original rendered normal map in the same world-space encoding. We compute

$$\mathbf{R}^v = \mathbf{M}^v \odot \left(\mathbf{C}_b^v - G_\sigma(\mathbf{C}_b^v)\right), \tag{5}$$

where G_σ is a Gaussian low-pass filter and $\mathbf{M}^v$ is an eroded foreground mask that suppresses residuals near silhouette boundaries. Within the valid foreground overlap, the target normal map is obtained by residual fusion and renormalization:

$$\mathbf{C}_{\text{tar}}^v = \frac{1}{2}\left[\mathcal{N}\left(2\left(\mathbf{C}_0^v + \alpha\mathbf{R}^v\right) - \mathbf{1}\right) + \mathbf{1}\right], \tag{6}$$

where α controls the strength of the transferred details and $\mathcal{N}$ is applied pixel-wise. The original normal maps anchor the fusion to the coarse geometry, while the residuals introduce local surface variations.

Mesh Refinement. Finally, the extracted geometric details are incorporated into the mesh geometry. We differentiably render the current mesh from the same viewpoints and optimize its vertex positions by minimizing the mean squared difference between rendered and target normal RGB values in the shared world-space encoding. We use continuous remeshing [41] during optimization to adapt the mesh tessellation, together with Laplacian regularization in the vertex updates to promote local smoothness.

5 Experiment

5.1 Experiment Setup

To comprehensively evaluate the effectiveness of StructBridge, we conduct two sets of experiments. First, we evaluate **3D generation under freely designed structural conditions** (denoted as **Exp 1**).

Table 1: Quantitative results of generation based on freely designed structure(Exp 1).

Method	Condition	MMD ↓	COV ↑	1-NNA	FID ↓	KID ↓	MOS-Q ↑	MOS-S ↑
Direct3D-s2	Fine-grained Sketch	0.01828	0.4077	**0.7416**	26.29	0.3417	4.11	4.11
Hunyuan3D-2.1	Fine-grained Sketch	0.0197	0.3974	0.7447	28.19	0.4615	4.02	3.18
LAS-Diffusion	Sketch	0.0223	0.3846	0.7996	54.59	1.6249	1.56	1.93
Coin3d	StructMap	0.0594	0.2885	0.9517	56.08	1.7902	2.67	3.76
StructBridge	StructMap	**0.0167**	**0.4423**	0.7691	**25.38**	**0.3245**	**4.31**	**4.57**

Table 2: Quantitative results of generation based on reference structure (Exp 2).

Method	Condition	MMD ↓	COV ↑	1-NNA	FID ↓	KID ↓	CD ↓	IoU ↑	F-score ↑	MOS-Q ↑	MOS-S ↑
Direct3D-s2	F.g. Sketch	**0.0231**	0.5171	0.6108	25.66	0.4002	0.1374	0.3440	0.6211	4.12	4.15
Hunyuan3D-2.1	F.g. Sketch	0.0261	0.5011	0.5965	37.6	0.92	0.1569	0.3123	0.5700	4.08	3.20
LAS-Diffusion	Sketch	0.0287	0.4323	0.6834	54.24	1.6525	0.1701	0.2746	0.5053	1.65	1.83
Coin3d	StructMap	0.0447	0.3494	0.8680	51.04	1.6193	0.1494	0.3374	0.5903	2.64	3.81
StructBridge	StructMap	0.0251	**0.5323**	**0.5675**	**21.68**	**0.3312**	**0.0330**	**0.5883**	**0.9409**	**4.39**	**4.61**

Specifically, we manually create a set of StructMaps and derive other forms of conditions based on the structural information encoded in these StructMaps. To prevent appearance differences under different conditions from affecting experimental fairness, we further provide a detailed style image for each case, ensuring that different methods generate geometric details with similar styles. Based on the StructMaps and the corresponding detail style images, annotators further create fine-grained sketches for sketch-based baselines. Then we use StructBridge and other baselines to generate 3D objects based on these structure-aligned conditions, and compare the quality of the generated objects. However, in Exp 1, we are unable to compare the structural fidelity of the generated objects since the baseline methods takes conditions of various modalities as input, making it impossible to find a unified metric to evaluate the structural consistency between the baselines' conditions and their respective generated objects. To address this, we introduce the second experiment: **Generation Conditioned on Reference Structures** (referred to as **Exp 2**). Specifically, we take off-the-shelf objects from [42] as reference and create structural conditions of all modalities based on the reference objects. In this way, we are able to use the reference 3D object as ground truth and evaluate the structral consistency between generated objects and the reference object using 3D-to-3D metrics. We use 500 test samples covering 20 categories of common 3D indoor objects in each experiment, and *the detailed data creation process is provided in Appendix C.1.*

Baseline methods. We consider all methods that have the potential to achieve structure-controllable 3D generation, and select four representative baselines: Coin3D [12], LAS-Diffusion [21], Direct3D-s2 [43], and Hunyuan3D-2.1 [9]. Specifically, we use 1) the StructMaps as coarse shape conditions for Coin3D, 2) the sketches aligned with the StructMaps as the condition for LAS-Diffusion, 3) the fine-grained sketches aligned with the StructMaps as the condition for Direct3D-s2 and Hunyuan3D-2.1. Notably, the original training data of Coin3D and LAS-Diffusion do not fully cover the categories used in our experiments, so we retrain their models using the same dataset as described in Sec. 4.3. *The results generated using the official models of Coin3D and LAS-Diffusion are included in Appendix D.3.*

Evaluation Metrics. We evaluate the generated objects from two aspects: mesh quality and structural consistency. Specifically, following [8, 34], we evaluate mesh quality using MMD (↓), COV (↑), and 1-NNA (optimal at 50%), and further compute FID and KID on the rendered images for feature-level assessment. To evaluate the structural consistency between the generated 3D models and their reference objects, we use Chamfer Distance (CD), Volume Intersection over Union (IoU), and F-score. In addition, we conduct a two-part Mean Opinion Score (MOS) study, where human participants evaluate 1) the quality of the generated 3D models (**MOS-Q**) and 2) the structural consistency between the generated models and their input conditions (**MOS-S**), using a 1–5 Likert scale.

5.2 Results & Analysis

Quantitative Analysis. In Tab. 1 and Tab. 2, we present the quantitative results of Exp 1 and Exp 2, which show that StructBridge achieves the best performance in most metrics. Specifically, StructBridge achieves the best performance in COV, FID, and KID, and achieves performance comparable to Direct3D-s2 w/ Fine-grained Sketch in MMD and 1-NNA. We note that our method is trained on a dataset that is significantly smaller than the one used for Direct3D-s2, yet achieves comparable or superior performance, demonstrating the high data efficiency of StructBridge. Moreover, our method achieves the lowest CD and the highest IoU and F-score, indicating that StructBridge preserves the

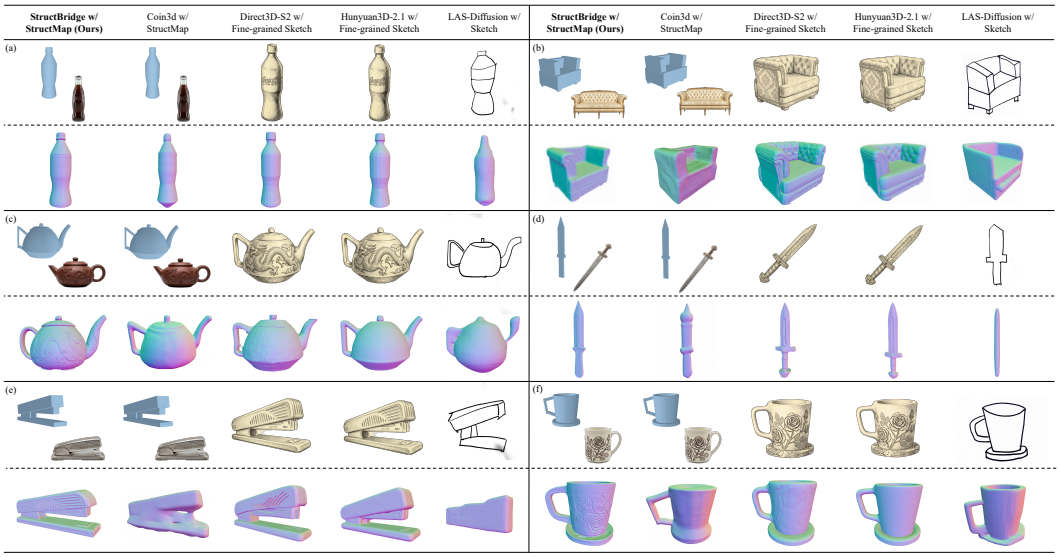

Figure 6: Qualitative results of Exp 1. For each sub-panel, the top row shows the input conditions for each method, and the bottom row shows the 3D object generated by different method.

intended structural configuration more faithfully than other methods, which stems from the structurally grounded design of StructBridge. In addition, for human evaluations, our method achieves the highest MOS-Q and MOS-S scores, indicating that our results are also perceived by humans as having the best mesh quality and structural consistency.

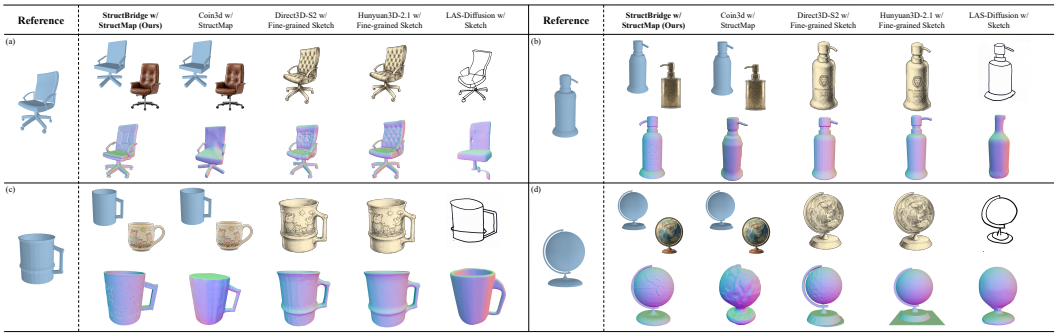

Figure 7: Qualitative results of Exp 2 across different methods. For each sub-panel, the leftmost column presents the reference object, the top row shows the input conditions for each method, and the bottom row shows the object generated by different method.

Qualitative Analysis. In Fig. 6 and Fig. 7, we present the generated objects and the corresponding input conditions in Exp 1 and Exp 2, which show that our method produces objects whose structures closely align with the input conditions. In contrast, objects generated by Coin3D w/ StructMap exhibits noticeable geometric distortions (Fig. 6 (d), (e)), while objects generated by Direct3D-s2 w/ Fine-grained Sketch and Hunyuan3D-2.1 w/ Fine-grained Sketch exhibit spatial layouts and relative sizes that differ from those specified in the input conditions (Fig. 6 (d); Fig. 7 (a)). And due to the limited structural cues in sketch conditions, the results from LAS-Diffusion w/ Sketch deviate from the intended overall structure (Fig. 7 (a); Fig. 6 (b)). In terms of mesh quality, our method produces meshes with smoother surfaces, richer geometric details, and more natural connections between primitives. By comparison, Coin3D w/ StructMap and LAS-Diffusion w/ Sketch often produce meshes with surface noise (Fig. 6 (e) Coin3D w/ StructMap) or unnatural shapes (Fig. 7 (c) LAS-Diffusion w/ Sketch). For Direct3D-s2 w/ Fine-grained Sketch and Hunyuan3D-2.1 w/ Fine-grained Sketch, the generated meshes lack realistic geometric details (Fig. 6 (c); Fig. 7 (a)). This is because they are trained on 3D datasets with limited semantic richness, which restricts their ability to generate fine-grained and semantically rich geometric details.

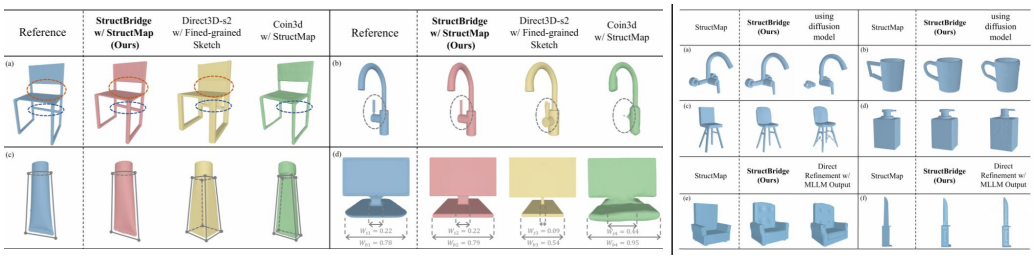

Figure 8: **[Left]**Detailed comparison of structural control capabilities across different methods. **[Right]** Qualitative results of the ablation studies.

5.3 Analysis of Structural Control Capabilities

To more intuitively demonstrate the ability of StructBridge to control spatial layout, geometric icons, and relative size, we present additional representative examples in Fig. 8-[Left], where we compare StructBridge with the two baselines that achieve the best structural consistency in main experiment. **1) Spatial layout.** In example (a), Direct3D-s2 w/ Fine-grained Sketch and Coin3D w/ StructMap produce chairs that either omit the lower crossbar or add spurious components, while in example (b), the faucet handle is either rotated incorrectly or entirely absent. In contrast, our method accurately preserves the correct spatial layout. **2) Geometric icons.** In example (c), the baselines generate a cuboidal shampoo bottle, whereas our method accurately captures the curved geometry. **3) Relative size.** In example (d), the display's support and base show notable scale discrepancies in the baselines, but our method maintains the correct proportions between parts.

5.4 Ablation Study

In Sec. 4, we introduce StructBridge and describe its two core components: the 3D latent diffusion bridge and the structure-preserving normal refiner. To validate the design of each component, we conduct two ablation studies to evaluate the effectiveness of their core designs. Qualitative and quantitative comparisons are summarized in Tab. 3 and Fig. 8-[Right].

Table 3: Quantitative results of the ablation study.

Method	MMD ↓	COV ↑	1-NNA	FID ↓	KID ↓
Ours	**0.0167**	**0.4423**	**0.7691**	**25.38**	**0.3245**
using diffusion model	0.0188	0.4119	0.7896	28.15	0.3872
direct refinement w/ MLLM output	0.0182	0.4201	0.7961	28.07	0.3711

Effect of the diffusion bridge. We replace the diffusion bridge with a standard diffusion model while keeping all other settings unchanged. As shown in Tab. 3, removing the diffusion bridge leads to a consistent drop on all five evaluated metrics, indicating a clear degradation in mesh quality. Results in Fig. 8-[Right] further validate these findings: the standard diffusion model produces meshes with noticeably poorer geometric quality (examples (a) and (c)) and structural inconsistencies (examples (a) and (d)). These effects arise because a standard diffusion model generate a 3D shape from pure noise, making it difficult to produce a high-quality mesh while preserving structural consistency.

Effect of the unit sphere projection and the frequency-aware normal extraction. We remove the unit sphere projection and frequency-aware normal extraction modules from the normal refiner, and directly use the normal maps generated by the MLLM for mesh refinement. As shown in Fig. 8-[Right], removing these modules results in significant surface noise (examples (e) and (f)). Results in Tab. 3 further show that removing these two modules leads to degraded mesh quality, indicating that they are crucial for improving mesh quality.

6 Conclusion

In this work, we introduced StructBridge, a novel generative framework that enables fine-grained structural control. Driven by the structure-grounded 3D object generation perspective, StructBridge directly conditions on the object's structural configuration and synthesizes coarse meshes with basic geometric details through a 3D latent diffusion bridge. StructBridge also integrates a structure-preserving normal refiner that enriches geometric details while preserving structure. Extensive experiments demonstrate that StructBridge outperforms existing 3D generation methods in both mesh quality and structural consistency.

Acknowledgments and Disclosure of Funding

This work was supported by the National Natural Science Foundation of China (No. 62595774 and No. 62606317), the Science and Technology Major Project of Jiangsu Province (No. BG2025035), Shanghai Artificial Intelligence Laboratory, and the XPLORER PRIZE grants.

References

[1] Rui Chen, Yongwei Chen, Ningxin Jiao, and Kui Jia. Fantasia3d: Disentangling geometry and appearance for high-quality text-to-3d content creation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 22246–22256, October 2023.

[2] Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. Magic3d: High-resolution text-to-3d content creation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[3] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv*, 2022.

[4] Ming Li, Pan Zhou, Jia-Wei Liu, Jussi Keppo, Min Lin, Shuicheng Yan, and Xiangyu Xu. Instant3d: Instant text-to-3d generation. *arxiv: 2311.08403*, 2023.

[5] Yuan Liu, Cheng Lin, Zijiao Zeng, Xiaoxiao Long, Lingjie Liu, Taku Komura, and Wenping Wang. Syncdreamer: Generating multiview-consistent images from a single-view image. *arXiv preprint arXiv:2309.03453*, 2023.

[6] Xiaoxiao Long, Yuan-Chen Guo, Cheng Lin, Yuan Liu, Zhiyang Dou, Lingjie Liu, Yuexin Ma, Song-Hai Zhang, Marc Habermann, Christian Theobalt, et al. Wonder3d: Single image to 3d using cross-domain diffusion. *arXiv preprint arXiv:2310.15008*, 2023.

[7] Longwen Zhang, Ziyu Wang, Qixuan Zhang, Qiwei Qiu, Anqi Pang, Haoran Jiang, Wei Yang, Lan Xu, and Jingyi Yu. Clay: A controllable large-scale generative model for creating high-quality 3d assets. *arXiv preprint arXiv:2406.13897*, 2024.

[8] Weiyu Li, Jiarui Liu, Hongyu Yan, Rui Chen, Yixun Liang, Xuelin Chen, Ping Tan, and Xiaoxiao Long. Craftsman3d: High-fidelity mesh generation with 3d native generation and interactive geometry refiner, 2024.

[9] Tencent Hunyuan3D Team. Hunyuan3d 2.1: From images to high-fidelity 3d assets with production-ready pbr material, 2025.

[10] Seunggwan Lee, Hwanhee Jung, Byoungsoo Koh, Qixing Huang, Sangho Yoon, and Sangpil Kim. Pasta: Part-aware sketch-to-3d shape generation with text-aligned prior. *arXiv preprint arXiv:2503.12834*, 2025.

[11] Yuanyou Xu, Zongxin Yang, and Yi Yang. SKDream: Controllable multi-view and 3d generation with arbitrary skeletons, 2024. URL `https://openreview.net/forum?id=6r1nbspMUl`.

[12] Wenqi Dong, Bangbang Yang, Lin Ma, Xiao Liu, Liyuan Cui, Hujun Bao, Yuewen Ma, and Zhaopeng Cui. Coin3d: Controllable and interactive 3d assets generation with proxy-guided conditioning, 2024.

[13] Jiayi Su, Youhe Feng, Zheng Li, Jinhua Song, Yangfan He, Botao Ren, and Botian Xu. Artformer: Controllable generation of diverse 3d articulated objects, 2025. URL `https://arxiv.org/abs/2412.07237`.

[14] Pushkal Katara, Zhou Xian, and Katerina Fragkiadaki. Gen2sim: Scaling up robot learning in simulation with generative models, 2023.

[15] Tencent Hunyuan3D Team. Hunyuan3d-omni: A unified framework for controllable generation of 3d assets, 2025. URL `https://arxiv.org/abs/2509.21245`.

[16] Yu-Wei Chao, Zhan Wang, Rada Mihalcea, and Jia Deng. Mining semantic affordances of visual object categories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[17] David Marr. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. MIT Press, 07 2010. ISBN 9780262514620. doi: 10.7551/mitpress/9780262514620.001.0001. URL `https://doi.org/10.7551/mitpress/9780262514620.001.0001`.

[18] Irving Biederman. Recognition-by-components: a theory of human image understanding. *Psychological review*, 94(2):115, 1987.

[19] Alexandre Binninger, Amir Hertz, Olga Sorkine-Hornung, Daniel Cohen-Or, and Raja Giryes. SENS: Part-aware sketch-based implicit neural shape modeling. *Computer Graphics Forum (Proceedings of EUROGRAPHICS 2024)*, 43(2), 2024.

[20] Gal Metzer, Elad Richardson, Or Patashnik, Raja Giryes, and Daniel Cohen-Or. Latent-nerf for shape-guided generation of 3d shapes and textures. *arXiv preprint arXiv:2211.07600*, 2022.

[21] Xin-Yang Zheng, Hao Pan, Peng-Shuai Wang, Xin Tong, Yang Liu, and Heung-Yeung Shum. Locally attentional sdf diffusion for controllable 3d shape generation. *ACM Transactions on Graphics (SIGGRAPH)*, 42(4), 2023.

[22] Hao-Yang Peng, Jia-Peng Zhang, Meng-Hao Guo, Yan-Pei Cao, and Shi-Min Hu. Charactergen: Efficient 3d character generation from single images with multi-view pose canonicalization. *ACM Transactions on Graphics (TOG)*, 43(4), 2024. doi: 10.1145/3658217.

[23] Anbang Wang, Yuzhuo Ao, Shangzhe Wu, and Chi-Keung Tang. Sk-adapter: Skeleton-based structural control for native 3d generation. *arXiv preprint arXiv:2603.14152*, 2026.

[24] Zhenyu Shu, Jiajun Shen, Zhongui Chen, Xiaoguang Han, and Shiqing Xin. Strucadt: Generating structure-controlled 3d point clouds with adjacency diffusion transformer. *IEEE Transactions on Visualization and Computer Graphics*, 31(12):10403–10420, December 2025. ISSN 2160-9306. doi: 10.1109/tvcg.2025.3600392. URL `http://dx.doi.org/10.1109/TVCG.2025.3600392`.

[25] Shitong Luo and Wei Hu. Diffusion probabilistic models for 3d point cloud generation, 2021. URL `https://arxiv.org/abs/2103.01458`.

[26] Linqi Zhou, Yilun Du, and Jiajun Wu. 3d shape generation and completion through point-voxel diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5826–5835, October 2021.

[27] Zhen Liu, Yao Feng, Michael J. Black, Derek Nowrouzezahrai, Liam Paull, and Weiyang Liu. Meshdiffusion: Score-based generative 3d mesh modeling. In *International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=0cpM2ApF9p6`.

[28] Gene Chou, Yuval Bahat, and Felix Heide. Diffusion-sdf: Conditional generative modeling of signed distance functions. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2023.

[29] J. Ryan Shue, Eric Ryan Chan, Ryan Po, Zachary Ankner, Jiajun Wu, and Gordon Wetzstein. 3d neural field generation using triplane diffusion. In *CVPR*, pages 20875–20886, 2023. URL `https://doi.org/10.1109/CVPR52729.2023.02000`.

[30] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2021.

[31] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper_files/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf`.

[32] Zibo Zhao, Wen Liu, Xin Chen, Xianfang Zeng, Rui Wang, Pei Cheng, BIN FU, Tao Chen, Gang YU, and Shenghua Gao. Michelangelo: Conditional 3d shape generation based on shape-image-text aligned latent representation. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL `https://openreview.net/forum?id=xmxgMij3LY`.

[33] 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. *arXiv preprint arXiv:2405.14832*, 2024.

[34] Daoyi Gao, Yawar Siddiqui, Lei Li, and Angela Dai. Meshart: Generating articulated meshes with structure-guided transformers. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025.

[35] Jianhua Sun, Yuxuan Li, Longfei Xu, Nange Wang, Jiude Wei, Yining Zhang, and Cewu Lu. Conceptfactory: Facilitate 3d object knowledge annotation with object conceptualization. *arXiv preprint arXiv:2411.00448*, 2024.

[36] Jianhua Sun, Yuxuan Li, Jiude Wei, Longfei Xu, Nange Wang, Yining Zhang, and Cewu Lu. Arti-pg: A toolbox for procedurally synthesizing large-scale and diverse articulated objects with rich annotations, 2024. URL `https://arxiv.org/abs/2412.14974`.

[37] Rui Chen, Jianfeng Zhang, Yixun Liang, Guan Luo, Weiyu Li, Jiarui Liu, Xiu Li, Xiaoxiao Long, Jiashi Feng, and Ping Tan. Dora: Sampling and benchmarking for 3d shape variational auto-encoders. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, pages 16251–16261, June 2025.

[38] Linqi Zhou, Aaron Lou, Samar Khanna, and Stefano Ermon. Denoising diffusion bridge models. *arXiv preprint arXiv:2309.16948*, 2023.

[39] 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. *arXiv preprint arXiv:2412.01506*, 2024.

[40] Chi Zhang, Yujun Cai, Guosheng Lin, and Chunhua Shen. Deepemd: Differentiable earth mover's distance for few-shot learning, 2020.

[41] Werner Palfinger. Continuous remeshing for inverse rendering. *Computer Animation and Virtual Worlds*, 33, 07 2022. doi: 10.1002/cav.2101.

[42] Kaichun Mo, Shilin Zhu, Angel X. Chang, Li Yi, Subarna Tripathi, Leonidas J. Guibas, and Hao Su. PartNet: A large-scale benchmark for fine-grained and hierarchical part-level 3D object understanding. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[43] 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.

[44] Kaiwen Zheng, Guande He, Jianfei Chen, Fan Bao, and Jun Zhu. Diffusion bridge implicit models. *arXiv preprint arXiv:2405.15885*, 2024.

[45] Quan Huynh-Thu, Marie-Neige Garcia, Filippo Speranza, Philip Corriveau, and Alexander Raake. Study of rating scales for subjective quality assessment of high-definition video. *IEEE Transactions on Broadcasting*, 57(1):1–14, 2011. doi: 10.1109/TBC.2010.2086750.

[46] Rensis Likert. A technique for the measurement of attitudes. *Archives of Psychology*, 22(140):1–55, 1932.

A Discussion

A.1 StructMap vs. Sketch

While both StructMap and sketches are able to describe an object's structure, they differ in dimensionality and expressiveness, which leads to different performance in structure-grounded 3D object generation. (1) Sketches are inherently 2D and manually drawn, requiring high professional skills yet still being difficult to faithfully capture 3D structural properties such as depth, angle of view, and symmetry. In contrast, StructMap is natively 3D, and thus naturally encodes such 3D attributes. Moreover, sketches struggle to maintain multi-view consistency, that is, creating coherent sketches of the same object from different viewpoints is nontrivial, as each new view requires redrawing and careful recalibration. In comparison, once a StructMap is constructed, rendering from different viewpoints becomes trivial and consistent, without requiring additional effort. (2) StructMap offers a more accessible structural input condition. Unlike sketches that demand drawing skills, StructMap creation only requires assembling geometric primitives via interpretable parameters. This significantly reduces the creation burden and skill barrier for users. Alternatively, attempts to simplify sketching, like using scribbles, often result in significant information loss, leading to incomplete or ambiguous structural specification.

A.2 Query-Based Latent Alignment

We adopt Dora VAE [37] because it provides compact, high-fidelity continuous latent representations suitable for DiT-based diffusion, with sharp-edge sampling and dual cross-attention capturing both global shape and salient local geometry. This set-valued representation introduces a token-ordering issue when constructing a diffusion bridge between independently encoded shapes. Spatially ordered alternatives, such as triplanes or spatial VQ-VAEs, avoid this particular permutation ambiguity through predefined spatial indexing, but involve different trade-offs in spatial discretization, reconstruction fidelity, and diffusion efficiency. To align the bridge endpoints, we use EMD to resolve token indexing rather than to enforce latent-feature similarity. In Dora VAE, sampled surface points serve as cross-attention queries and determine the latent-token indices, while each token aggregates broader contextual information. We compute a minimum-cost spatial assignment between the paired StructMap and object query sets and use it to reorder the target latent tokens along the token dimension, without changing their feature values or requiring matched tokens to have identical features. Because each pair shares a structural configuration and a common normalization, spatial proximity provides a correspondence prior for endpoint alignment. The resulting correspondence is specific to each paired query set and serves to define the endpoint ordering for the bridge loss; it does not establish semantic part correspondence or a shared token identity across different objects.

B Details about StructMap

B.1 StructMap Modeling Interface

To make it convenient for users to create StructMaps, we develop a StructMap modeling interface based on the conceptualization interface from ConceptFactory [35]. Specifically, We first provide a set of geometric primitive templates for users to choose. Users can instantiate the desired primitives and adjust their parameters to modify their shapes and poses. Then, we also incorporate the STL-C templates from ConceptFactory into our interface, which we refer to as high order configurations. Users can directly select these higher-level templates to further improve the efficiency of StructMap creation. We present a video demonstrating the StructMap creation process in modeling_interface.mp4.

B.2 Efficiency of obtaining StructMaps

ConceptFactory reports an average annotation time of about 7 min for object conceptualization. In contrast, our StructMap modeling interface does not require precise alignment of object sizes, making the modeling process relatively faster. We measured the average time for creating StructMaps in our test data, which is approximately 5 min 30 s.

Additionally, we present the recorded creation times and creators' retrospective difficulty ratings for coarse sketches, StructMaps, and fine-grained sketches in Table 4. In this comparison, StructMap

costs moderately more than a coarse sketch but is substantially faster and easier than a fine-grained sketch.

Table 4: Creation time and retrospective difficulty ratings for different input conditions.

Condition	Time (min)	Difficulty (1–5 scale)
Coarse sketch	1.36	2.1
StructMap	5.50	2.7
Fine-grained sketch	20.76	4.4

B.3 Scalability and Complexity of StructMaps

The STL-C in ConceptFactory consists of 263 concept templates that comprehensively covers the essential structure of daily objects [35], demonstrating the strong scalability of the representation proposed in ConceptFactory. Moreover, our StructMap modeling interface allows users to create StructMaps using geometric primitives, which gives StructMaps greater flexibility, scalability, and capacity to represent complex structures compared to the representation proposed in ConceptFactory.

C Detailed Experiment Settings

C.1 Test Data Construction

In our experiments, we used two sets of test data, corresponding to Exp 1 and Exp 2 respectively.

For the data in Exp 1, we first used our StructMap Modeling Interface to create a set of StructMaps. These StructMaps describe the structures of 20 categories of common indoor objects, and we intentionally selected diverse combinations of geometric primitives and high-order configurations to ensure diversity in the StructMaps. Next, based on this set of StructMaps, we created a set of other structure-aligned condition types, including: 1) textual descriptions of the StructMaps' structural configuration; 2) sketches created based on the StructMaps' structure; 3) fine-grained sketches created based on the StructMaps' structure. The textual descriptions, sketches and fine-grained sketches were created by 10 volunteers. Notably, to reduce the technical expertise and time required from creators while ensuring the quality of the fine-grained sketches, we leverage MLLM-based optimization techniques to enhance the fine-grained sketches.

For the data in Exp 2, we take off-the-shelf objects from [42] as references, and create a set of StructMaps based on the structures of these reference objects, and then create other types of conditions. Different from Exp1, the textual descriptions, sketches and fine-grained sketches conditions are created directly from the reference data.

C.2 Implementation Details

Bridge training and sampling. We train the diffusion bridge on 3,472 paired examples using 4 H200 GPUs with mixed precision for 300k steps, using AdamW with a learning rate of 10^{-4} and a batch size of 128. The released training configuration uses zero weight decay, an exponential moving average with decay 0.9999, no learning-rate annealing, and seed 42. The implementation adopts the `i2sb_cond` noise schedule with 1,000 discrete entries and parameters $\beta_{\min} = 0.1$ and $\beta_{\max} = 1.0$. Training times are sampled uniformly over $[10^{-4}, 1)$ and mapped to the schedule by nearest-index rounding. Intermediate latent states are sampled from the conditional Gaussian bridge distribution between the aligned target-object latent $\tilde{z}_0$ and the StructMap latent $\tilde{z}_T$, with the training objective given in the main text. At inference, we use 30 DBIM [44] sampling steps with $\eta = 0.9$, a uniform time grid with endpoint cutoffs 10^{-4} and $1 - 10^{-3}$, and denoised clipping disabled. The released sampling configuration uses seed 42 and full-precision model execution. For SDF mesh extraction, we use resolution 512, corresponding to an octree depth of 9.

Network architecture. We use the Dora VAE [37], and keep both its encoder and decoder frozen. The encoder selects 384 coarse queries and 384 sharp queries by farthest-point sampling, producing 768 latent tokens of dimension 64. Query features attend to the input point features, so a token is

indexed by its query location but is not restricted to information from that single point. The VAE has hidden width 768, 12 attention heads, 8 encoder layers, and 16 decoder layers. Following the DiT architecture used in Craftsman3D [8], the bridge network uses 32 transformer blocks, hidden width 1024, and 16 attention heads. The StructMap latent z_T is also supplied to the DiT as conditioning input through cross-attention.

Normal Refiner. We use Gemini 3 Pro Image for normal-map enhancement. We render four views at 90° azimuth intervals and zero elevation, each at 1024×1024 resolution, and assemble them into a 2×2 atlas for joint processing. For frequency-aware extraction, bilateral filtering uses diameter 15, color standard deviation 10, and spatial standard deviation 75 on 8-bit normal RGB maps, followed by Gaussian filtering with a 121×121 kernel and standard deviation 20 pixels. We set the detail strength to $\alpha = 1$. Mesh fitting uses nvdiffrast at 2048×2048 resolution per view for 101 optimization updates, with a base step-size parameter of 0.3 and Laplacian regularization coefficient 0.01. Adaptive remeshing uses local reference edge lengths in $[0.0005, 0.002]$ in normalized mesh coordinates.

C.3 Dataset Splits

The training, validation, and test splits contain 3,472, 500, and 500 instances, respectively. The training and test sets share the same 20 indoor-object categories but contain no overlapping object instances. Table 5 lists these categories.

Table 5: The 20 indoor-object categories shared by the training and test sets.

Bottle	Box	Bucket	Chair	Clip
Dispenser	Display	Eyeglasses	Faucet	Globe
Kettle	Kitchenpot	Knife	Laptop	Microwave
Mug	Scissors	Shampoo	Stapler	Table

C.4 Computational Cost

Training takes approximately 48 hours on four H200 GPUs, each with 141 GB of memory. Inference requires approximately 15 GB of GPU memory and takes about 10 minutes per object on H200.

C.5 Human Evaluation

We conduct a two-part Mean Opinion Score (MOS) study [45], where human participants evaluate (1) the quality of the generated 3D model (**MOS-Q**) and (2) the structural consistency between the generated models and their input conditions (**MOS-S**) on a 1–5 Likert scale [46].

The detailed scoring criteria are defined as follows:

MOS-Q (3D Model Quality): This metric evaluates the overall quality, realism, and geometric plausibility of the generated 3D model.

- **1 – Very Poor:** The 3D model is severely flawed, containing broken geometry, missing parts, or extreme distortions. The shape is clearly unrealistic or unusable.
- **2 – Poor:** The model has substantial artifacts or implausible geometry. Many components exhibit incorrect proportions or noticeable defects.
- **3 – Fair:** The model is generally recognizable but contains visible issues such as uneven surfaces, inaccurate proportions, or moderate noise/distortions.
- **4 – Good:** The model is mostly clean, realistic, and well-formed, with only minor geometric imperfections.
- **5 – Excellent:** The model is highly realistic, clean, and structurally coherent, with high-quality geometry suitable for downstream use.

MOS-S (Structural Consistency): This metric evaluates how well the generated 3D model matches the structural configuration specified by the input condition.

- **1 – Very Poor:** The generated model does not reflect the input structure at all; major components are missing, misplaced, or incorrectly shaped.
- **2 – Poor:** Only a small portion of the input structure is preserved; many structural configurations are inaccurate or distorted.
- **3 – Fair:** The overall alignment with the input structure is partially maintained, but several components deviate noticeably in spatial layout, geometric icons, and relative size.
- **4 – Good:** The model largely follows the input structural configuration, with only minor deviations in details or proportions.
- **5 – Excellent:** The model closely matches the input structural specification, with an accurate correspondence in spatial layout, geometric icons, and relative size.

Specifically, we collected 40 human evaluation responses, including 20 participants with experience in computer vision or graphics and 20 participants without relevant domain expertise. Each participant was asked to rate a randomly sampled subset of generated objects using the scoring criteria described above.

To reduce potential bias, the order of samples was randomized for each participant, and participants were not informed of the underlying generation method used to generate the objects. Each object was evaluated by multiple participants, and the final MOS-Q and MOS-S scores were computed by averaging the ratings across all collected responses.

All participants were fairly compensated for their time at a standard hourly rate.

We identified no foreseeable risks to participants arising from the evaluation of generated 3D indoor objects.

D Additional Experimental Results

D.1 More Qualitative Results

To further demonstrate the generality and effectiveness of StructBridge across diverse object categories, we provide additional qualitative results in Fig. 9 and Fig. 10. These additional examples show that StructBridge generates objects whose structures closely follow the input structural conditions, while maintaining high mesh quality with smooth surfaces (Fig. 10 (c), (e)), rich geometric details (Fig. 9 (c), (h)), and natural connections between primitives (Fig. 9 (a), (c)). In contrast, competing methods often exhibit noticeable structural deviations, such as incorrect spatial layouts (Fig. 9 (e)-Direct3D-S2 w/ Fine-grained Sketch), distorted part proportions (Fig. 9 (f)-Hunyuan3D-2.1 w/ Fine-grained Sketch), or missing structural components (Fig. 9 (a)-Coin3d w/ StructMap). Moreover, methods relying on sketch conditions frequently fail to preserve the intended structure due to the limited structural cues provided by these inputs (Fig. 9 (e)-LAS-Diffusion w/ Sketch).

D.2 Effect of Structure-preserving Normal Refiner

To illustrate the effects of our 3D Latent Diffusion Bridge and the Structure-Preserving Normal Refiner, we present the two-stage generated meshes for several examples in Fig. 11. From these examples, we observe that the 3D Latent Diffusion Bridge primarily transforms a structure representation composed of geometric primitives into a geometrically plausible real-world object. Specifically, it introduces basic geometric details, such as smoothing the edges of primitives and creating natural connections between them. The Structure-Preserving Normal Refiner subsequently enriches the object surface with finer and more diverse geometric details.

To further quantify the contribution of the entire refinement stage, we compare the full pipeline with a variant that directly outputs the coarse mesh generated by the diffusion bridge, without applying the Normal Refiner. As shown in Table 6, adding the Normal Refiner improves all five reported generation-quality metrics. Together with the mesh and normal-map comparisons in Fig. 11, these results support the contribution of the refinement stage to geometric detail and overall generation quality.

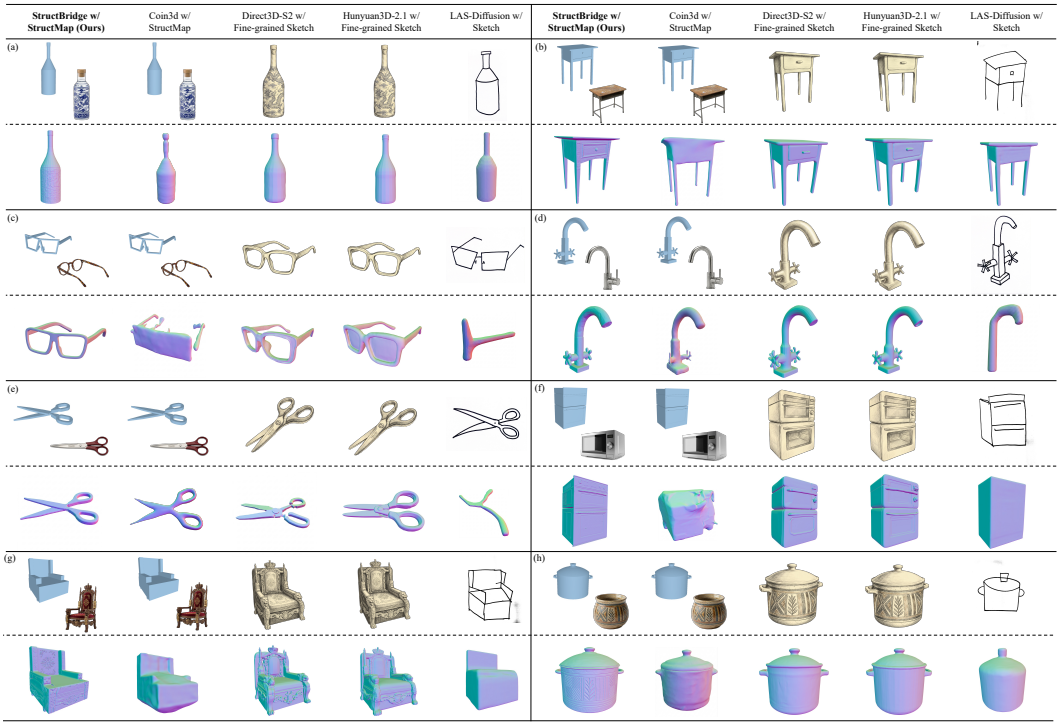

Figure 9: More qualitative results of Exp 1 across different methods. For each sub-panel, the top row shows the input conditions for each method, and the bottom row shows the 3D object generated by different method. To more clearly illustrate the fine-grained geometric details added to the object surface, we visualize the generated objects using normal maps.

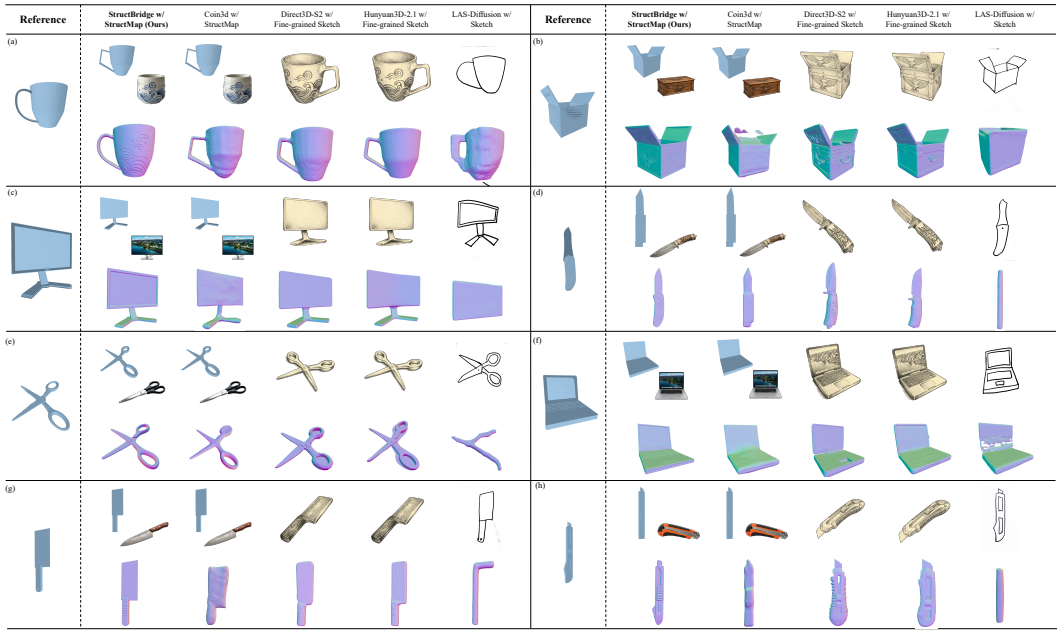

Figure 10: More qualitative results of Exp 2 across different methods. For each sub-panel, the leftmost column presents the reference object, the top row shows the input conditions for each method, and the bottom row shows the 3D object generated by different method. To more clearly illustrate the fine-grained geometric details added to the object surface, we visualize the generated objects using normal maps.

Table 6: Ablation of the entire Structure-Preserving Normal Refiner.

Configuration	MMD ↓	COV ↑	1-NNA	FID ↓	KID ↓
w/o Normal Refiner	0.0191	0.4025	0.8037	28.96	0.3922
Full Pipeline	**0.0167**	**0.4423**	**0.7691**	**25.38**	**0.3245**

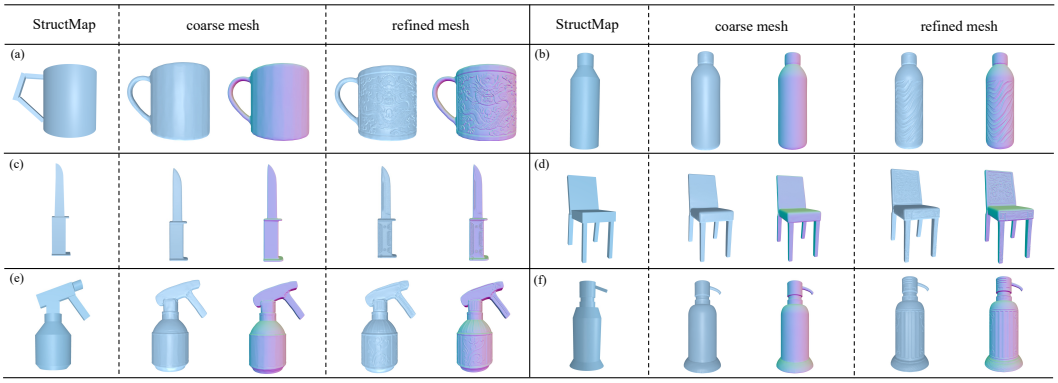

Figure 11: Examples of the two-stage mesh generation results. To better visualize the geometric details, for each case we present both the mesh (left) and the corresponding normal map (right).

D.3 Generating using the official pretrained models of Coin3D and LAS-Diffusion

As mentioned in Sec. 5.1 of the main paper, we retrain their models for Coin3D and LAS-Diffusion, bacause the original training data of Coin3D and LAS-Diffusion do not fully cover the categories used in our experiments. Therefore, in Fig. 12, we present the results generated using the official pretrained models of Coin3D and LAS-Diffusion. As shown in Fig. 12, for some categories(e.g.bottle for Coin3D, mug for LAS-Diffusion), the results generated using the official pretrained models are very poor and far inferior to those presented in Fig. 6 in the main paper.

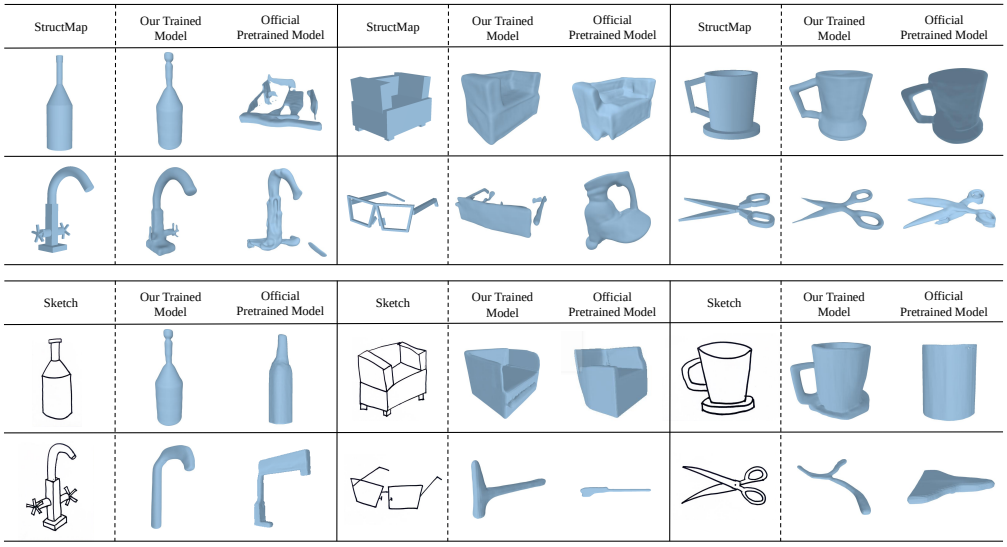

Figure 12: For Coin3D and LAS-Diffusion, we present a comparison between the results generated using our trained models and those produced by the official pretrained models. **[Top]** The Coin3D results. **[Bottom]** The LAS-Diffusion results.

D.4 Generation Diversity Analysis

To evaluate generation diversity under fixed structural conditions, we generate five outputs for each of selected StructMaps using five independent inference seeds and a single trained checkpoint. We measure diversity using pairwise Chamfer Distance (CD) and feature distance between outputs generated from the same StructMap. We additionally evaluate CD, IoU, and F-score against the corresponding reference object to examine whether variation across outputs is accompanied by changes in structural fidelity. As shown in Table 7, the nonzero pairwise CD and feature distance confirm stochastic variation across seeds. Meanwhile, CD, IoU, and F-score remain close to the main results (0.0330/0.5883/0.9409), with small deviations. StructBridge therefore produces variation under a fixed condition without materially weakening structural fidelity.

Table 7: Generation diversity and reference fidelity under fixed structural conditions using five inference seeds. Pairwise CD and feature distance measure differences between generated outputs.

Method	Pairwise CD	Feature distance	Reference CD ↓	IoU ↑	F-score ↑
Result	0.0051 ± 0.0007	0.038 ± 0.008	0.034 ± 0.004	0.585 ± 0.016	0.936 ± 0.009

D.5 Comparison with a Retrieval Baseline

To examine whether the performance of StructBridge can be explained by directly retrieving similar training objects, we construct a nearest-training-object baseline. For each test StructMap, we compute the symmetric point-cloud Chamfer Distance between the query and each training object, and return the object with the smallest distance. As shown in Table 8, StructBridge outperforms direct retrieval across all reported metrics. These results suggest that its performance cannot be explained by retrieving the training object closest to the input StructMap.

Table 8: Comparison with nearest-training-object retrieval under the reference-based evaluation protocol of Exp. 2.

Method	MMD ↓	COV ↑	1-NNA	FID ↓	KID ↓	CD ↓	IoU ↑	F-score ↑
Nearest training object	0.0281	0.3067	0.8265	35.29	0.8328	0.1624	0.2803	0.5597
StructBridge	**0.0251**	**0.5323**	**0.5675**	**21.68**	**0.3312**	**0.0330**	**0.5883**	**0.9409**

D.6 Generalization to Unseen Categories

To evaluate category-level generalization, we test StructBridge on four categories absent from its diffusion-bridge training data: lighter, pliers, safe, and washingmachine. We use the trained model without additional fine-tuning and evaluate generation quality and structural fidelity using the same metrics as in Exp. 2. As shown in Table 9, generation quality degrades on unseen categories, while the structural-fidelity metrics suggest that StructBridge retains a degree of alignment with the prescribed configuration. This is expected because realistic geometry and details rely on category-specific priors absent from training, whereas StructMap still explicitly specifies the core structure. Thus, even when realism generalizes imperfectly, StructBridge remains reliably grounded in the prescribed structural configuration.

Table 9: Comparison between in-distribution and out-of-distribution evaluation on four unseen categories.

Setting	MMD ↓	COV ↑	1-NNA	FID ↓	KID ↓	CD ↓	IoU ↑	F-score ↑
In-distribution	0.0251	0.5323	0.5675	21.68	0.3312	0.0330	0.5883	0.9409
Unseen categories	0.0329	0.4518	0.6790	41.18	0.9865	0.0747	0.4903	0.8084

Table 10: Error bar of generation based on freely designed structure(Exp 1).

Method	Condition	MMD ↓	COV ↑	1-NNA	FID ↓	KID ↓	MOS-Q ↑	MOS-S ↑
Direct3D-s2	Fine-grained Sketch	±0.00045	±0.0062	±0.0060	±0.75	±0.0108	±0.08	±0.07
Hunyuan3D-2.1	Fine-grained Sketch	±0.00050	±0.0060	±0.0062	±0.80	±0.0130	±0.09	±0.10
LAS-Diffusion	Sketch	±0.00055	±0.0055	±0.0070	±1.50	±0.045	±0.10	±0.11
Coin3d	StructMap	±0.0015	±0.0045	±0.0080	±1.60	±0.050	±0.09	±0.10
StructBridge	StructMap	±0.0004	±0.0065	±0.0055	±0.70	±0.0095	±0.07	±0.06

Table 11: Error bar of generation based on reference structure (Exp 2).

Method	Condition	MMD ↓	COV ↑	1-NNA	FID ↓	KID ↓	CD ↓	IoU ↑	F-score ↑	MOS-Q ↑	MOS-S ↑
Direct3D-s2	F.g. Sketch	±0.0006	±0.0068	±0.0060	±0.70	±0.011	±0.004	±0.007	±0.006	±0.08	±0.07
Hunyuan3D-2.1	F.g. Sketch	±0.0007	±0.0065	±0.0060	±1.0	±0.025	±0.0045	±0.0065	±0.007	±0.09	±0.10
LAS-Diffusion	Sketch	±0.0008	±0.0058	±0.0070	±1.4	±0.048	±0.005	±0.006	±0.008	±0.11	±0.12
Coin3d	StructMap	±0.0012	±0.0050	±0.0075	±1.3	±0.045	±0.0045	±0.006	±0.007	±0.09	±0.10
StructBridge	StructMap	±0.0006	±0.0068	±0.0055	±0.60	±0.009	±0.001	±0.006	±0.006	±0.06	±0.05

D.7 Error Bars

We provide error bars for our experiments in Table. 10 and 11. The error bars are calculated using the standard deviation (SD) of the results. Specifically, for each experimental result, we compute the standard deviation across 5 independent trials, and the error bars represent this standard deviation.

E Limitation

While our method achieves effective structure-preserving and appearance-guided geometric detail synthesis, it still has several limitations. First, although the normal refinement module can introduce rich fine-grained details, it is still limited in modeling extremely complex or highly non-local geometric patterns that require global reasoning. Second, our method is primarily designed for object-level generation and may not directly generalize to large-scale scenes without additional adaptation.

F Societal Impact

Our work has both positive and potential negative societal impacts. On the positive side, it provides a controllable pipeline for generating high-quality 3D assets, which can benefit applications in digital content creation, gaming, virtual reality, and industrial design by significantly reducing manual modeling effort. It may also lower the barrier for non-expert users to create complex 3D content. On the other hand, like many generative modeling techniques, our approach could be misused to generate large amounts of synthetic 3D content, potentially leading to issues such as content overproduction or misuse in misleading digital representations. Additionally, improvements in automated 3D generation may impact certain manual modeling workflows. We believe these risks can be mitigated through appropriate usage policies and responsible deployment.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly present the paper's key contributions and scope.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitations are discussed in the Appendix.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification: The paper does not include theoretical results.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The implementation details are provided in Appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The code is publicly available at `https://github.com/wngnbr/structbridge`.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Implementation and experimental details are provided in Appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Error bars are reported in Appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Details on computational resources are provided in Appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research fully complies with the NeurIPS Code of Ethics, with no foreseeable ethical concerns.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Potential positive and negative societal impacts are discussed in Appendix.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All external assets used in the paper are properly credited, and their licenses and terms of use are respected.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not release new assets

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [Yes]

    Justification: We provide the full text of participant instructions and compensation details in Appendix.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [Yes]

    Justification: Appendix C.5 describes the potential risks to participants and how these risks were disclosed, and explains why ethics approval was not required under the applicable institutional guidelines.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.

- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: Our method pipeline involves the use of an MLLM, and we provide a detailed description of its usage in the paper.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.