Direct3D-S2: Gigascale 3D Generation Made Easy with Spatial Sparse Attention

Shuang Wu[1,2*] **Youtian Lin**[1,2*] **Feihu Zhang**[2] **Yifei Zeng**[1,2] **Yikang Yang**[1]

Yajie Bao[2] **Jiachen Qian**[2] **Siyu Zhu**[3] **Xun Cao**[1] **Philip Torr**[4] **Yao Yao**[1†]

[1]Nanjing University [2]DreamTech [3]Fudan University [4]University of Oxford

{wushuang,linyoutian}@smail.nju.edu.cn {caoxun,yaoyao}@nju.edu.cn

Abstract

Generating high-resolution 3D shapes using volumetric representations such as Signed Distance Functions (SDFs) presents substantial computational and memory challenges. We introduce Direct3D-S2, a scalable 3D generation framework based on sparse volumes that achieves superior output quality with dramatically reduced training costs. Our key innovation is the **Spatial Sparse Attention (SSA)** mechanism, which greatly enhances the efficiency of Diffusion Transformer (DiT) computations on sparse volumetric data. SSA allows the model to effectively process large token sets within sparse volumes, significantly reducing computational overhead and achieving a 3.9× speedup in the forward pass and a 9.6× speedup in the backward pass. Our framework also includes a variational autoencoder (VAE) that maintains a consistent sparse volumetric format across input, latent, and output stages. Compared to previous methods with heterogeneous representations in 3D VAE, this unified design significantly improves training efficiency and stability. Our model is trained on public datasets, and experiments demonstrate that Direct3D-S2 not only surpasses state-of-the-art methods in generation quality and efficiency, but also enables **training at 1024^3 resolution using only 8 GPUs**, a task typically requiring at least 32 GPUs for volumetric representations at 256^3 resolution, thus making gigascale 3D generation both practical and accessible. Project page: https://www.neural4d.com/research-page/direct3d-s2.

1 Introduction

Generating high-quality 3D models directly from text or images offers significant creative potential, enabling rapid 3D content creation for virtual worlds, product prototyping, and various real-world applications. This capability has garnered increasing attention across domains such as gaming, virtual reality, robotics, and computer-aided design.

Recently, large-scale 3D generative models based on implicit latent representations have made notable progress. These methods leverage neural fields for shape representation, benefiting from compact latent codes and scalable generation capabilities. For instance, 3DShape2Vecset [46] pioneered diffusion-based shape synthesis by using a Variational Autoencoder (VAE) [13] to encode 3D shapes into a latent vecset, which can be decoded into neural SDFs or occupancy fields and rendered via Marching Cubes [23]. The latent vecset is then modeled with a diffusion process to generate

*Equal contribution.
†Corresponding author.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

Figure 1: Mesh generation results from our method on different input images. Using our high-resolution SS-VAE and SS-DiT, we are able to generate detailed and complex 3D shapes. The meshes show fine geometry and high visual quality, demonstrating the strength of our approach for high-resolution 3D generation.

diverse 3D shapes. CLAY [48] extended this pipeline with Diffusion Transformers (DiT) [29], while TripoSG [17] further improved fidelity through rectified flow transformers and hybrid supervision. However, implicit latent-based methods often rely on VAEs with asymmetric 3D representations, resulting in lower training efficiency that typically requires hundreds of GPUs.

Explicit latent methods have emerged as a compelling alternative to implicit ones, offering better interpretability, simpler training, and direct editing capabilities, while also adopting scalable architectures such as DiT [29]. For instance, Direct3D [38] proposes to use tri-plane latent representations to accelerate training and convergence. XCube [31] introduces hierarchical sparse voxel latent diffusion for 1024^3 sparse volume generation, but only restricted to millions of valid voxels, limiting the final output quality. Trellis [39] integrates sparse voxel representations of 256^3 resolution, with the rendering supervision for the VAE training. In general, due to high memory demands, existing explicit-latent methods are limited in output resolution. Scaling to 1024^3 with sufficient latent tokens and valid voxels remains challenging, as the quadratic cost of full attention in DiT renders high-resolution training computationally prohibitive.

To address the challenge of high-resolution 3D shape generation, we propose Direct3D-S2, a unified generative framework that utilizes sparse volumetric representations. At the core of our approach is a novel Spatial Sparse Attention (SSA) mechanism, which significantly improves the scalability of diffusion transformers in high-resolution 3D shape generation by selectively attending to spatially important tokens via learnable compression and selection modules. Specifically, we draw inspiration from the key principles of Native Sparse Attention (NSA) [45], which integrates compression, selection, and windowing to identify relevant tokens based on global-local interactions. While NSA is designed for structurally organized 1D sequences, it is not directly applicable to unstructured sparse 3D data. To adapt it, we redesign the block partitioning to preserve 3D spatial coherence and revise the core modules to accommodate the irregular nature of sparse volumetric tokens. This enables efficient processing of large token sets within sparse volumes. We implement a custom Triton [36] GPU kernel for SSA, achieving a **3.9$\times$ speedup in the forward pass** and a **9.6$\times$ speedup in the backward pass** compared to FlashAttention-2 at 1024^3 resolution.

Our framework also includes a VAE that maintains a consistent sparse volumetric format across input, latent, and output stages. This unified design eliminates the need for cross-modality translation, commonly seen in previous methods using mismatched representations such as point cloud input, 1D

vector latent, and dense volume output, thereby improving training efficiency, stability, and geometric fidelity. After the VAE training, the DiT with the proposed SSA will be trained on the converted latents, enabling scalable and efficient high-resolution 3D shape generation.

Extensive experiments demonstrate that our approach successfully achieves high-quality and efficient gigascale 3D generation, a milestone previously unattainable by explicit 3D latent diffusion methods. Compared to prior native 3D diffusion techniques, our model consistently generates highly detailed 3D shapes while significantly reducing computational costs. Notably, Direct3D-S2 requires only 8 GPUs to train on public datasets [8, 9, 20] at a resolution of 1024^3, in stark contrast to prior state-of-the-art methods, which typically require 32 or more GPUs even for training at 256^3 resolution.

2 Related work

2.1 Multi-view Generation and 3D Reconstruction

Large-scale 3D generation has been advanced by methods such as [15, 21, 22, 41], which employ multi-view diffusion models [37] trained on 2D image prior models like Stable Diffusion [32] to generate multi-view images of 3D shapes. These multi-view images are then used to reconstruct 3D shapes via generalized sparse-view reconstruction models. Follow-up works [19, 26, 35, 42, 47] further improve the quality and efficiency of reconstruction by incorporating different 3D representations. Despite these advances, these methods still face challenges in maintaining multi-view consistency and shape quality. The synthesized images may fail to faithfully represent the underlying 3D structure, which could result in artifacts and reconstruction errors. Another limitation is the reliance on rendering-based supervision, such as Neural Radiance Fields (NeRF)[27] or DMTet[33]. While this avoids the need for direct 3D supervision (e.g., meshes), it adds significant complexity and computational overhead to the training process. Rendering-based supervision can be slow and costly, especially when scaled to large datasets.

2.2 Large Scale 3D Latent Diffusion Model

Motivated by recent advances in Latent Diffusion Models (LDMs) [32] in 2D image generation, several methods have extended LDMs to 3D shape generation. These approaches broadly fall into two categories: vecset-based methods and voxel-based methods. Implicit vecset-based methods, such as 3DShape2Vecset [46], Michelangelo [50], CLAY [48], and CraftsMan3D [16], represent 3D shapes using latent vecset and reconstruct meshes through neural SDFs or occupancy fields. However, implicit methods are typically constrained by vecset size: larger vecset lead to more complex mappings to the 3D shape and require longer training times. In contrast, voxel-based methods, such as XCube [31] and Trellis [39], employs voxel grids as latent representations, providing more interpretability and easier training. Nevertheless, voxel-based methods face limitations in latent resolution due to cubic growth in GPU memory requirements and high computational costs associated with attention mechanisms. To address this issue, our work specifically targets reducing the computational overhead of attention mechanisms, thereby enabling the generation of high-resolution voxel-based latent representations that were previously infeasible.

2.3 Efficient Large Tokens Generation

Generating large tokens efficiently is a challenging problem, especially for high-resolution data. Native Sparse Attention (NSA) [45] addresses this by introducing adaptive token compression that reduce the number of tokens involved in attention computation, while maintaining performance comparable to full attention. NSA has been successfully applied to large language models [45, 30] and video generation [34], showing significant reductions in attention cost. In this paper, we extend token compression to 3D data and propose a new Spatial Sparse Attention (SSA) mechanism. SSA adapts the core ideas of NSA but modifies the block partitioning strategy to respect 3D spatial coherence. We also redesign the compression, selection, and window modules to better fit the properties of sparse 3D token sets. Another line of work, such as linear attention [12], reduces attention complexity by approximating attention weights with linear functions. Variants of this technique have been applied in image [40, 52] and video generation [25] to improve efficiency. However, the absence of non-linear similarity can lead to a significant decline in the performance of the model.

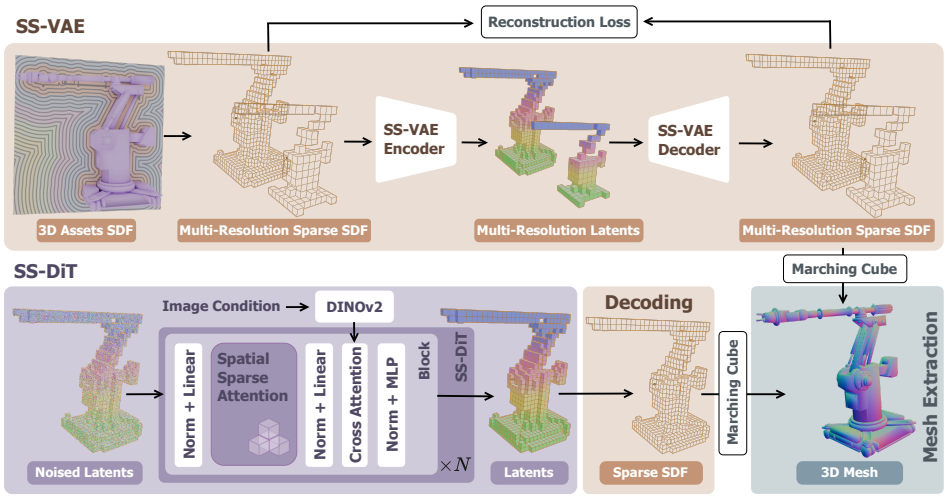

Figure 2: The framework of our Direct3D-S2. We propose a fully end-to-end sparse SDF VAE (SS-VAE), which employs a symmetric encoder-decoder network to efficiently encode high-resolution sparse SDF volumes into sparse latent representations $\mathbf{z}$. Then we train an image-conditioned diffusion transformer (SS-DiT) based on $\mathbf{z}$, and design a novel Spatial Sparse Attention (SSA) mechanism that significantly improves the training and inference efficiency of the DiT.

3 Sparse SDF VAE

While variational autoencoders (VAEs) have become the cornerstone of 2D image generation by compressing pixel representations into compact latent spaces for efficient diffusion training, their extension to 3D geometry faces fundamental challenges. Unlike images with standardized pixel grids, 3D representations lack a unified structure, while meshes, point clouds, and implicit fields each require specialized processing. This fragmentation forces existing 3D VAEs into asymmetric architectures with compromised efficiency. For instance, prominent approaches [6, 48, 49] based on vecset [46] encode the input point cloud into a vector set latent space before decoding it into SDF field, while Trellis [39] and XCube [31] rely on differentiable rendering or post-hoc neural kernel surface reconstruction [11] to bridge their latent spaces to usable meshes. These hybrid pipelines introduce computational bottlenecks and geometric approximations that limit their scalability to high-resolution 3D generation. In this paper, we propose a fully end-to-end sparse SDF VAE that employs a symmetric encoding-decoding network to encode high-resolution sparse SDF volumes into a sparse latent representation, improving training efficiency while maintaining geometric precision.

Given a mesh represented as a signed distance field (SDF) volume V with resolution R^3 (e.g., 1024^3), the SS-VAE first encodes it into a latent representation $\mathbf{z} = E(V)$, then reconstructs the SDF through the decoder $\tilde{V} = D(\mathbf{z})$. Direct processing of dense R^3 SDF volumes proves computationally prohibitive. To address this, we strategically focus on valid sparse voxels where absolute SDF values fall below threshold τ:

$$V = \{(\mathbf{x}_i, s(\mathbf{x}_i)) \big| |s(\mathbf{x}_i)| < \tau\}_{i=1}^{|V|}, \tag{1}$$

where $s(\mathbf{x}_i)$ denotes the SDF value at position $\mathbf{x}_i$.

3.1 Symmetric Network Architecture

Our fully end-to-end SDF VAE framework adopts a symmetric encoder-decoder network architecture, as illustrated in the upper half of Fig 2. Specifically, the encoder employs a hybrid framework combining sparse 3D convolution networks and transformer networks. We first extract local geometric features through a series of residual sparse 3D CNN blocks interleaved with 3D mean pooling operations, progressively downsampling the spatial resolution. We then process the sparse voxels as variable-length tokens and utilize transformer layers to capture global contextual information between the valid voxels. Inspired by Trellis [39], the feature of each valid voxel is augmented with positional encoding based on its 3D coordinates before being fed into multi-head self-attention layers. This

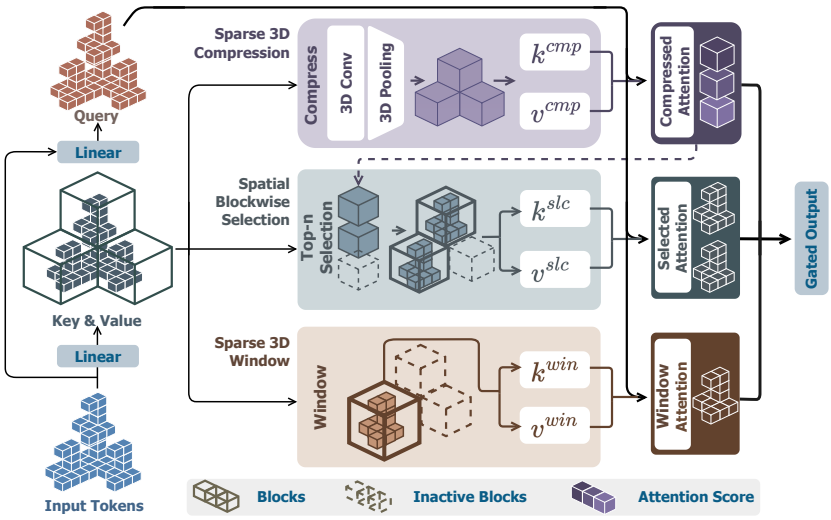

Figure 3: The framework of our Spatial Sparse Attention (SSA). We partition the input tokens into blocks based on their 3D coordinates, and then construct key-value pairs through three distinct modules. For each query token, we utilize sparse 3D compression module to capture global information, while the spatial blockwise selection module selects important blocks based on compression attention scores to extract fine-grained features, and the sparse 3D window module injects local features. Ultimately, we aggregate the final output of SSA from the three modules using predicted gate scores.

hybrid design outputs sparse latent representations at reduced resolution $(\frac{R}{f})^3$, where f denotes the downsampling factor. The decoder of our SS-VAE adopts a symmetric structure with respect to the encoder, leveraging transformer layers and sparse 3D CNN blocks to progressively upsample the latent representation and reconstruct the SDF volume $\tilde{V}$.

3.2 Training Losses

The decoded sparse voxels $\tilde{V}$ contain both the input voxels $\tilde{V}_{\text{in}}$ and additional valid voxels $\tilde{V}_{\text{extra}}$ We enforce supervision on the SDF values across all these spatial positions. To enhance geometric fidelity, we impose additional supervision on the active voxels situated near the sharp edges of the mesh, specifically in regions exhibiting high-curvature variations on the mesh surface. Moreover, the term of KL-divergence regularization is imposed on the latent representation $\mathbf{z}$ to constrain excessive variations in the latent space. The overall training objective of our SS-VAE is formulated as:

$$\mathcal{L}_c = \frac{1}{|\tilde{V}_c|} \sum_{(\mathbf{x},\, \tilde{s}(\mathbf{x})) \in \tilde{V}_c} \left\| s(\mathbf{x}) - \tilde{s}(\mathbf{x}) \right\|_2^2, \quad c \in \{\text{in}, \text{ext}, \text{sharp}\}, \tag{2}$$

$$\mathcal{L}_{\text{total}} = \sum_c \lambda_c \, \mathcal{L}_c + \lambda_{\text{KL}} \, \mathcal{L}_{\text{KL}}, \tag{3}$$

where λ_{in}, λ_{ext}, λ_{sharp} and λ_{KL} denote the weight of each term.

3.3 Multi-resolution Training

To enhance training efficiency and enable our SS-VAE to encode meshes across varying resolutions, we utilize the multi-resolution training paradigm. Specifically, during each training iteration, we randomly sample a target resolution from the candidate set $\{256^3, 384^3, 512^3, 1024^3\}$, then trilinearly interpolate the input SDF volume to the selected resolution before feeding it into the SS-VAE.

4 Spatial Sparse Attention and DiT

Through our SS-VAE framework, 3D shapes can be encoded into latent representations $\mathbf{z}$. We adopt a multi-stage generation pipeline, where the sparse structure of $\mathbf{z}$ is generated first, followed by the

generation of the sparse latent representations $\mathbf{z}$. For the first stage, we employ the same Structure Flow Model as Trellis [39]. In the stage of generating the sparse latent representations $\mathbf{z}$, to ensure efficient generation of high-resolution meshes, we propose spatial sparse attention that substantially accelerates both training and inference processes. Furthermore, we introduce a sparse conditioning mechanism to extract the foreground region of the input images, thereby reducing the number of conditioning tokens. The architecture of the DiT is illustrated in the lower half of Fig 2.

4.1 Spatial Sparse Attention

Given input tokens $\mathbf{q}$, $\mathbf{k}$, $\mathbf{v} \in \mathbb{R}^{N \times d}$, where N denotes the token length, and d represents the head dimension, the standard full attention is formulated as:

$$\mathbf{o}_t = \text{Attn}(\mathbf{q}_t, \mathbf{k}, \mathbf{v}) = \sum_{i=1}^{N} \frac{\mathbf{p}_{t,i}\mathbf{v}_i}{\sum_{j=1}^{N} \mathbf{p}_{t,j}}, \quad \mathbf{p}_{t,j} = e^{\frac{\mathbf{q}_t^T \mathbf{k}_j}{\sqrt{d}}}, \quad t \in [0, N). \tag{4}$$

As the resolution of SS-VAE escalates, the length of input tokens grows substantially, reaching over 100k at a resolution of 1024^3, leading to prohibitively low computational efficiency in attention operations. Inspired by NSA (Native Sparse Attention) [45], we proposes Spatial Sparse Attention mechanism, which partitions key and value tokens into spatially coherent blocks based on their geometric relationships and performs blockwise token selection to achieve significant acceleration.

A naive implementation involves treating latent tokens $\mathbf{z}$ as a 1D sequence and partitioning it into fixed-length blocks based on token indices, analogous to NSA. However, this approach suffers from two critical limitations: On the one hand, tokens within the same block may not be spatially adjacent in 3D space, despite sharing contiguous indices. On the other hand, due to the sparse voxel structure, blocks with identical indices across different samples occupy divergent spatial regions. These issues collectively lead to unstable training convergence. To resolve these challenges, we propose partitioning blocks based on 3D coordinates. As illustrated in Fig 3, we divide the 3D space into subgrids of size m^3, where active tokens from sparse voxels residing in the same subgrid are grouped into one block. Our Spatial Sparse Attention comprises three core modules: sparse 3D compression, spatial blockwise selection, and sparse 3D window. The attention computation proceeds as follows:

$$\mathbf{o}_t = \omega_t^{\text{cmp}} \text{Attn}(\mathbf{q}_t, \mathbf{k}_t^{\text{cmp}}, \mathbf{v}_t^{\text{cmp}}) + \omega_t^{\text{slc}} \text{Attn}(\mathbf{q}_t, \mathbf{k}_t^{\text{slc}}, \mathbf{v}_t^{\text{slc}}) + \omega_t^{\text{win}} \text{Attn}(\mathbf{q}_t, \mathbf{k}_t^{\text{win}}, \mathbf{v}_t^{\text{win}}), \tag{5}$$

where $\mathbf{k}_t$ and $\mathbf{v}_t$ represent the selected key and value tokens in each module for query $\mathbf{q}_t$, respectively. ω_t is the gating score for each module, obtained by applying a linear layer followed by a sigmoid activation to the input features.

Sparse 3D Compression. After partitioning input tokens into spatially coherent blocks based on their 3D coordinates, we leverage a compression module to extract block-level representations of the input tokens. Specifically, we first incorporate intra-block positional encoding for each token within a block of size m_{cmp}^3, then employ sparse 3D convolution followed by sparse 3D mean pooling to compress the entire block:

$$\mathbf{k}_t^{\text{cmp}} = \delta(\mathbf{k}_t + \text{PE}(\mathbf{k}_t)), \tag{6}$$

where $\mathbf{k}_t^{\text{cmp}}$ denotes the block-level key token, $\text{PE}(\cdot)$ is absolute position encoding, and $\delta(\cdot)$ represents operations of sparse 3D convolution and sparse 3D mean pooling. The sparse 3D compression module effectively captures block-level global information while significantly reducing the token count, thereby enhancing computational efficiency.

Spatial Blockwise Selection. The block-level representations only contain coarse-grained information, necessitating the retention of token-level features to enhance the fine details in the generated 3D shapes. However, the excessive number of input tokens leads to computationally inefficient operations if all tokens are utilized. By leveraging the sparse 3D compression module, we compute the attention scores $\mathbf{s}_{\text{cmp}}$ between the query $\mathbf{q}$ and each compression block, subsequently selecting all tokens within the top-k blocks exhibiting the highest scores. The resolution m_{slc} of the selection blocks must be both greater than and divisible by the resolution m_{cmp} of the compression blocks. The relevance score $\mathbf{s}_t^{\text{slc}}$ for a selection block is aggregated from its constituent compression blocks. GQA (Grouped-Query Attention) [4] is employed to further improve computational efficiency, the attention

scores of the shared query heads within each group are accumulated as follows:

$$\mathbf{s}_t^{\text{slc}} = \sum_{i \in \mathcal{B}_{\text{cmp}}} \sum_{h=1}^{h_s} s_{t,h}^{\text{cmp},i}, \tag{7}$$

where $\mathcal{B}_{\text{cmp}}$ denotes the set of compression blocks within the selection block, and h_s represents the number of shared heads within a group. The top-k selection blocks with the highest $\mathbf{s}_t^{\text{slc}}$ scores are selected, and all tokens contained within them are concatenated to form $\mathbf{k}_t^{\text{slc}}$ and $\mathbf{v}_t^{\text{slc}}$, which are used to compute the spatial blockwise selection attention. We implement the attention kernel using Triton [36], with a key challenge arising from the sparse 3D voxel structure where the number of tokens per block varies. To address this, we pre-compute the index $\mathcal{C}$ of the first token in each block as an input to the kernel and sort the input tokens based on their block indices, ensuring that all tokens belonging to the same block are contiguous in memory. We describe the spatial blockwise selection attention forward pass in Algorithm.1 of the appendix.

Sparse 3D Window. In addition to sparse 3D compression and spatial blockwise selection modules, we further employ an auxiliary sparse 3D window module to explicitly incorporate localized feature interactions. Drawing inspiration from Trellis [39], we partition the input token-containing voxels into non-overlapping windows of size m_{win}^3. For each token, we formulate its contextual computation by dynamically aggregating active tokens within the corresponding window to form $\mathbf{k}_t^{\text{win}}$ and $\mathbf{v}_t^{\text{win}}$, followed by localized self-attention calculation exclusively over this constructed token subset.

Through the modules of sparse 3D compression, spatial blockwise selection, and sparse 3D window, corresponding key-value pairs are constructed. Subsequently, attention calculations are performed for each module, and the results are aggregated and weighted according to gate scores to produce the final output of the spatial sparse attention mechanism.

4.2 Sparse Conditioning Mechanism

Existing image-to-3D models [16, 38, 48] typically employ DINO-v2 [28] to extract pixel-level features from conditional images, followed by cross-attention operation with noisy tokens to achieve conditional generation. However, for a majority of input images, more than half of the regions consist of background, which not only introduces additional computational overhead but may also adversely affect the alignment between the generated meshes and the conditional images. To mitigate this issue, we propose a sparse conditioning mechanism that selectively extracts and processes sparse foreground tokens from input images for cross-attention computation. Formally, given an input image $\mathcal{I}$, the sparse conditioning tokens $\mathbf{c}$ are computed as follows:

$$\mathbf{c} = \text{Linear}(f(E_{\text{DINO}}(\mathcal{I}))) + \text{PE}(f(E_{\text{DINO}}(\mathcal{I}))), \tag{8}$$

where E_{DINO} is the DINO-v2 encoder, $f(\cdot)$ denotes the operation of extracting the foreground tokens based on the mask, $\text{PE}(\cdot)$ is the absolute position encoding, and $\text{Linear}(\cdot)$ represents a linear layer. Then we perform cross attention using the finalized sparse conditioning tokens $\mathbf{c}$ and the noisy tokens.

4.3 Rectified Flow

We employ rectified flow objective [10, 18] to train our generative model. Rectified flow defines forward process as linear trajectory between data distribution and standard normal distribution:

$$\mathbf{x}(t) = (1-t)\mathbf{x}_0 + t\epsilon, \tag{9}$$

where ϵ is the noise, and t denotes the timestep. Our generative model is trained to predict the velocity field from noisy samples to the data distribution. The training loss is formulated using conditional flow matching, formulated as follows:

$$\mathcal{L}_{\text{CFM}} = \mathbb{E}_{t,\mathbf{x}_0,\epsilon} \|\mathbf{v}_\theta(\mathbf{x}_t, \mathbf{c}, t) - (\epsilon - \mathbf{x}_0)\|_2^2, \tag{10}$$

where $\mathbf{v}_\theta$ is the neural networks.

5 Experiments

5.1 Datasets

Our Direct3D-S2 is trained on publicly available 3D datasets including Objaverse [9], Objaverse-XL [8], and ShapeNet [5]. Due to the prevalence of low-quality meshes in these collections, we

Table 1: Quantitative comparisons of meshes generated by different methods in the image-to-3D task.

Methods	ULIP-2 ↑	Uni3D ↑	OpenShape ↑
Trellis [39]	0.2825	0.3755	0.1732
Hunyuan3D 2.0 [49]	0.2535	0.3738	0.1699
TripoSG [17]	0.2626	0.3870	0.1728
Hi3DGen [44]	0.2725	0.3723	0.1689
Ours	**0.3111**	**0.3931**	**0.1752**

Figure 4: Qualitative comparisons between other image-to-3D methods and our approach.

curated approximately 452k 3D assets through rigorous filtering for training. Following prior approach [48] in geometry processing, we first convert the original non-watertight meshes into watertight ones, then compute ground-truth SDF volumes that serve as both input to and supervision for our SS-VAE. For training our image-conditioned DiT, we render 45 RGB images per mesh at 1024×1024 resolution with random camera parameters. The camera configuration space is defined as follows: elevation angles ranging from 10° to 40°, azimuth angles spanning $[0^\circ, 180^\circ]$, and focal

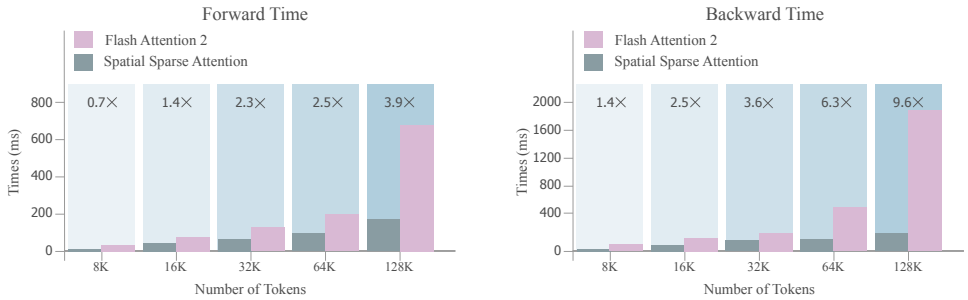

Figure 5: Comparison of the forward and backward time of our SSA and FlashAttention-2.

lengths varying between 30mm and 100mm. To compare the geometric fidelity of meshes generated by Direct3D-S2 and other methods, we established a challenging benchmark comprising highly detailed images sourced from professional communities including Neural4D [3], Meshy [2], and CivitAI [1]. The quantitative assessment employs ULIP-2 [43], Uni3D [51] and OpenShape [20] metrics to measure shape-image alignment between generated meshes and conditional input images, enabling systematic comparison with state-of-the-art 3D generation methods. For the ablation studies, we conduct qualitative comparisons on this benchmark, and perform quantitative evaluations on a subset from the Objaverse dataset that does not overlap with the training set. We employ the FID metric to more rigorously evaluate the geometry quality of generated meshes, which requires GT meshes. We render paired normal maps for both the generated meshes and GT meshes from the same viewpoints, then calculate the Normal-FID between them to assess model performance. The implementation details of our Direct3D-S2 are provided in the appendix.

5.2 Quantitative and Qualitative Comparisons

To empirically validate the effectiveness of our Direct3D-S2 framework, we conducted comprehensive experiments against state-of-the-art image-to-3D approaches. The quantitative results are reported in Tab 1, where it is evident that our Direct3D-S2 outperforms the other approaches across three metrics, indicating that the meshes produced by our Direct3D-S2 achieve better alignment with the input images. Moreover, we present qualitative comparisons in Fig 4. Although the other methods generate overall satisfactory results, they struggle to capture finer structures due to resolution limitations, as illustrated by the railings of the house and surrounding branches of trees in the first row. In contrast, thanks to our proposed SSA mechanism, our Direct3D-S2 is capable of generating high-resolution meshes, achieving superior quality even for these intricate details. We provide additional qualitative comparisons in the appendix.

5.3 Ablation Studies

Runtime of Different Attention Mechanisms. We implemented a custom Triton [36] GPU kernel for SSA. And we compare the forward and backward execution times of our SSA with those of FlashAttention-2 [7] across various token counts, using the implementation from Xformers [14] for FlashAttention-2. The comparison results are shown in Fig 5, which indicate that our SSA achieves comparable speeds to FlashAttention-2 when the token count is low; however, as the token count increases, the speed advantage of our SSA becomes more pronounced. Specifically, when the token count reaches 128k, the forward and backward speeds of our SSA are $3.9\times$ and $9.6\times$ faster than those of Flash Attention 2, respectively, demonstrating the efficiency of our SSA.

Table 2: Quantitative comparisons of meshes generated using different attention mechanisms. For *Full Attention*, we apply a latent packing strategy to downsample the tokens.

Methods	**Normal-FID** ↑
Full Attention	76.58
NSA	71.93
SSA (Ours)	**69.31**

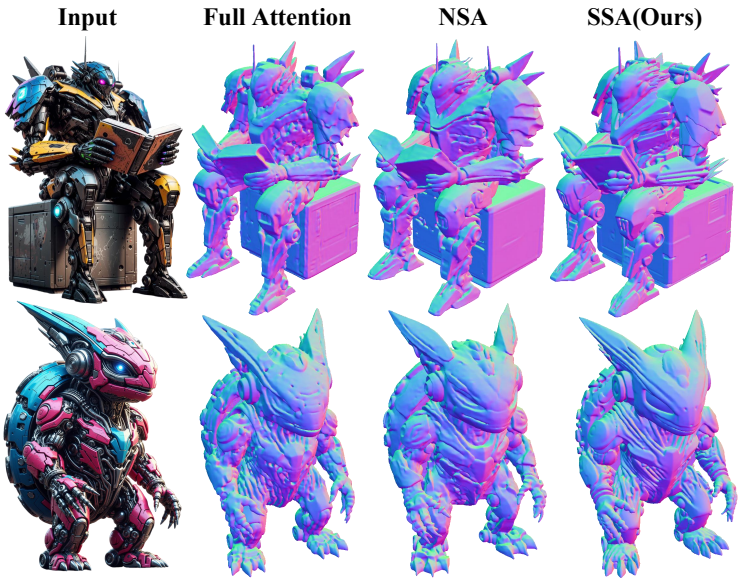

Figure 6: Ablation studies of our proposed SSA mechanism.

Effectiveness of SSA. We conduct ablation studies to validate the robustness of SSA. Given the insufficient geometric fidelity at 256^3/ 384^3 resolutions, which do not adequately reflect the model's precision, and prohibitive computational costs at 1024^3 resolution, we perform experiments at 512^3 resolution. We establish three comparative configurations: 1) Full attention: directly training the DiT with full attention proves to be inefficient. Therefore, following Trellis' latent packing strategy [39], we group latent tokens within 2^3 local regions to reduce the token count before feeding them into the DiT blocks. 2) NSA: process latent tokens as 1D sequences with fixed-length block partitioning, disregarding spatial coherence. 3) Our proposed SSA. We report the quantitative results in Tab 2, where SSA outperforms the other attention mechanisms. The qualitative results are illustrated in Fig 6. It is evident that the *Full Attention* variant produces meshes with high-frequency surface artifacts, attributed to its forced packing operation that disrupts local geometric continuity. The NSA implementation exhibits training instability due to positional ambiguity in block partitioning, resulting in less smooth meshes. In contrast, our SSA not only preserves the details of the meshes, but also yields a smoother and more organized surface, thereby demonstrating the effectiveness of our proposed SSA mechanism.

Additional ablation studies and the evaluation of our SS-VAE are provided in the appendix.

6 Conclusion

In this work, we presented a novel framework for high-resolution 3D shape generation, dubbed Direct3D-S2. The key contribution of our approach is the design of Spatial Sparse Attention (SSA) mechanism, which significantly accelerates the training and inference speed of DiT. The integration of fully end-to-end symmetric sparse SDF VAE further enhances training stability and efficiency. Extensive experiments demonstrate that our Direct3D-S2 outperforms existing state-of-the-art image-to-3D methods in generation quality, while requiring only 8 GPUs for training.

Acknowledgment. This work is supported by National Natural Science Foundation of China (62472213), Gusu Innovation & Entrepreneurship Leading Talents Program (ZXL2024361), and computing resources from DreamTech.

References

[1] Civitai. https://civitai.com/.

[2] Meshy. https://www.meshy.ai/.

[3] Neural4d. https://www.neural4d.com/.

[4] Joshua Ainslie, James Lee-Thorp, Michiel De Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. Gqa: Training generalized multi-query transformer models from multi-head checkpoints. *arXiv preprint arXiv:2305.13245*, 2023.

[5] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[6] 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. *arXiv preprint arXiv:2412.17808*, 2024.

[7] Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*, 2023.

[8] Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, et al. Objaverse-xl: A universe of 10m+ 3d objects. In *NeurIPS*, 2023.

[9] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *CVPR*, 2023.

[10] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *ICML*, 2024.

[11] Jiahui Huang, Zan Gojcic, Matan Atzmon, Or Litany, Sanja Fidler, and Francis Williams. Neural kernel surface reconstruction. In *CVPR*, 2023.

[12] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *ICML*, 2020.

[13] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014.

[14] Benjamin Lefaudeux, Francisco Massa, Diana Liskovich, Wenhan Xiong, Vittorio Caggiano, Sean Naren, Min Xu, Jieru Hu, Marta Tintore, Susan Zhang, Patrick Labatut, Daniel Haziza, Luca Wehrstedt, Jeremy Reizenstein, and Grigory Sizov. xformers: A modular and hackable transformer modelling library. https://github.com/facebookresearch/xformers, 2022.

[15] Jiahao Li, Hao Tan, Kai Zhang, Zexiang Xu, Fujun Luan, Yinghao Xu, Yicong Hong, Kalyan Sunkavalli, Greg Shakhnarovich, and Sai Bi. Instant3d: Fast text-to-3d with sparse-view generation and large reconstruction model. In *ICLR*, 2024.

[16] Weiyu Li, Jiarui Liu, Rui Chen, Yixun Liang, Xuelin Chen, Ping Tan, and Xiaoxiao Long. Craftsman: High-fidelity mesh generation with 3d native generation and interactive geometry refiner. *arXiv preprint arXiv:2405.14979*, 2024.

[17] Yangguang Li, Zi-Xin Zou, Zexiang Liu, Dehu Wang, Yuan Liang, Zhipeng Yu, Xingchao Liu, Yuan-Chen Guo, Ding Liang, Wanli Ouyang, et al. Triposg: High-fidelity 3d shape synthesis using large-scale rectified flow models. *arXiv preprint arXiv:2502.06608*, 2025.

[18] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[19] Minghua Liu, Ruoxi Shi, Linghao Chen, Zhuoyang Zhang, Chao Xu, Xinyue Wei, Hansheng Chen, Chong Zeng, Jiayuan Gu, and Hao Su. One-2-3-45++: Fast single image to 3d objects with consistent multi-view generation and 3d diffusion. In *CVPR*, 2024.

[20] Minghua Liu, Ruoxi Shi, Kaiming Kuang, Yinhao Zhu, Xuanlin Li, Shizhong Han, Hong Cai, Fatih Porikli, and Hao Su. Openshape: Scaling up 3d shape representation towards open-world understanding. In *NeurIPS*, 2023.

[21] Minghua Liu, Chao Xu, Haian Jin, Linghao Chen, Mukund Varma T, Zexiang Xu, and Hao Su. One-2-3-45: Any single image to 3d mesh in 45 seconds without per-shape optimization. In *NeurIPS*, 2024.

[22] 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. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9970–9980, 2024.

[23] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Seminal graphics: pioneering efforts that shaped the field*, pages 347–353. 1998.

[24] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[25] Kaiyue Lu, Zexiang Liu, Jianyuan Wang, Weixuan Sun, Zhen Qin, Dong Li, Xuyang Shen, Hui Deng, Xiaodong Han, Yuchao Dai, et al. Linear video transformer with feature fixation. *arXiv preprint arXiv:2210.08164*, 2022.

[26] Yuanxun Lu, Jingyang Zhang, Shiwei Li, Tian Fang, David McKinnon, Yanghai Tsin, Long Quan, Xun Cao, and Yao Yao. Direct2. 5: Diverse text-to-3d generation via multi-view 2.5 d diffusion. In *CVPR*, 2024.

[27] B Mildenhall, PP Srinivasan, M Tancik, JT Barron, R Ramamoorthi, and R Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

[28] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

[29] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *ICCV*, pages 4195–4205, 2023.

[30] Piotr Piękos, Róbert Csordás, and Jürgen Schmidhuber. Mixture of sparse attention: Content-based learnable sparse attention via expert-choice routing. *arXiv preprint arXiv:2505.00315*, 2025.

[31] Xuanchi Ren, Jiahui Huang, Xiaohui Zeng, Ken Museth, Sanja Fidler, and Francis Williams. Xcube: Large-scale 3d generative modeling using sparse voxel hierarchies. In *CVPR*, 2024.

[32] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022.

[33] Tianchang Shen, Jun Gao, Kangxue Yin, Ming-Yu Liu, and Sanja Fidler. Deep marching tetrahedra: a hybrid representation for high-resolution 3d shape synthesis. In *NeurIPS*, 2021.

[34] Xin Tan, Yuetao Chen, Yimin Jiang, Xing Chen, Kun Yan, Nan Duan, Yibo Zhu, Daxin Jiang, and Hong Xu. Dsv: Exploiting dynamic sparsity to accelerate large-scale video dit training. *arXiv preprint arXiv:2502.07590*, 2025.

[35] Jiaxiang Tang, Zhaoxi Chen, Xiaokang Chen, Tengfei Wang, Gang Zeng, and Ziwei Liu. Lgm: Large multi-view gaussian model for high-resolution 3d content creation. In *ECCV*, 2024.

[36] Philippe Tillet, Hsiang-Tsung Kung, and David Cox. Triton: an intermediate language and compiler for tiled neural network computations. In *Proceedings of the 3rd ACM SIGPLAN International Workshop on Machine Learning and Programming Languages*, 2019.

[37] Peng Wang and Yichun Shi. Imagedream: Image-prompt multi-view diffusion for 3d generation. *arXiv preprint arXiv:2312.02201*, 2023.

[38] 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. In *NeurIPS*, 2024.

[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] Enze Xie, Junsong Chen, Junyu Chen, Han Cai, Haotian Tang, Yujun Lin, Zhekai Zhang, Muyang Li, Ligeng Zhu, Yao Lu, et al. Sana: Efficient high-resolution image synthesis with linear diffusion transformers. *arXiv preprint arXiv:2410.10629*, 2024.

[41] Jiale Xu, Weihao Cheng, Yiming Gao, Xintao Wang, Shenghua Gao, and Ying Shan. Instantmesh: Efficient 3d mesh generation from a single image with sparse-view large reconstruction models. *arXiv preprint arXiv:2404.07191*, 2024.

[42] Yinghao Xu, Zifan Shi, Wang Yifan, Hansheng Chen, Ceyuan Yang, Sida Peng, Yujun Shen, and Gordon Wetzstein. Grm: Large gaussian reconstruction model for efficient 3d reconstruction and generation. In *ECCV*, 2024.

[43] Le Xue, Ning Yu, Shu Zhang, Artemis Panagopoulou, Junnan Li, Roberto Martín-Martín, Jiajun Wu, Caiming Xiong, Ran Xu, Juan Carlos Niebles, et al. Ulip-2: Towards scalable multimodal pre-training for 3d understanding. In *CVPR*, 2024.

[44] Chongjie Ye, Yushuang Wu, Ziteng Lu, Jiahao Chang, Xiaoyang Guo, Jiaqing Zhou, Hao Zhao, and Xiaoguang Han. Hi3dgen: High-fidelity 3d geometry generation from images via normal bridging. *arXiv preprint arXiv:2503.22236*, 2025.

[45] Jingyang Yuan, Huazuo Gao, Damai Dai, Junyu Luo, Liang Zhao, Zhengyan Zhang, Zhenda Xie, YX Wei, Lean Wang, Zhiping Xiao, et al. Native sparse attention: Hardware-aligned and natively trainable sparse attention. *arXiv preprint arXiv:2502.11089*, 2025.

[46] Biao Zhang, Jiapeng Tang, Matthias Niessner, and Peter Wonka. 3dshape2vecset: A 3d shape representation for neural fields and generative diffusion models. *ACM Transactions On Graphics (TOG)*, 42(4):1–16, 2023.

[47] Kai Zhang, Sai Bi, Hao Tan, Yuanbo Xiangli, Nanxuan Zhao, Kalyan Sunkavalli, and Zexiang Xu. Gs-lrm: Large reconstruction model for 3d gaussian splatting. In *ECCV*, 2024.

[48] 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. *ACM Transactions on Graphics (TOG)*, 43(4):1–20, 2024.

[49] Zibo Zhao, Zeqiang Lai, Qingxiang Lin, Yunfei Zhao, Haolin Liu, Shuhui Yang, Yifei Feng, Mingxin Yang, Sheng Zhang, Xianghui Yang, et al. Hunyuan3d 2.0: Scaling diffusion models for high resolution textured 3d assets generation. *arXiv preprint arXiv:2501.12202*, 2025.

[50] 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 *NeurIPS*, 2023.

[51] Junsheng Zhou, Jinsheng Wang, Baorui Ma, Yu-Shen Liu, Tiejun Huang, and Xinlong Wang. Uni3d: Exploring unified 3d representation at scale. *arXiv preprint arXiv:2310.06773*, 2023.

[52] Lianghui Zhu, Zilong Huang, Bencheng Liao, Jun Hao Liew, Hanshu Yan, Jiashi Feng, and Xinggang Wang. Dig: Scalable and efficient diffusion models with gated linear attention. *arXiv preprint arXiv:2405.18428*, 2024.

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: We explicitly delineate the contributions and scope of our approach in both the abstract and the introduction in Section. 1 of our paper.

   Guidelines:

   - The answer NA 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 NA 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: We discusses the limitations of the work in the appendix.

   Guidelines:

   - The answer NA 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: [NA]

Justification: Our paper does not contain theoretical results.

Guidelines:

- The answer NA 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 paper provides detailed descriptions of the experimental setup, including the algorithms used, parameter settings, datasets, and evaluation metrics. Additionally, it outlines the procedures followed to obtain the results, ensuring that other researchers can replicate the experiments and verify the findings. We mention this in Section. 5 and the appendix.

Guidelines:

- The answer NA 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: [No]

   Justification: We will release the code after the completion of the review process.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://nips.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://nips.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, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We specify all the training and testing details in Section. 5 and the appendix.

   Guidelines:

   - The answer NA 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: [No]

   Justification: Repeated training runs of large transformer models strain available GPU resources.

   Guidelines:

   - The answer NA 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: We provide comprehensive information on the type of computing workers, memory, and time of execution required for each experiment. We mention this in Section. 5

   Guidelines:

   - The answer NA 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: We have carefully reviewed the NeurIPS Code of Ethics, and we confirm that our research adheres to all outlined ethical guidelines, including considerations for fairness, transparency, and respect for all participants involved.

   Guidelines:

   - The answer NA 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: [NA]

    Justification: There is no societal impact of the work performed.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA 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., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper does not involve the release of data or models that pose a high risk for misuse.

    Guidelines:

    - The answer NA 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: The creators or original owners of all assets used in the paper are properly credited. The licenses and terms of use for these assets are explicitly mentioned and have been respected in accordance with their respective requirements.

    Guidelines:

    - The answer NA 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: [Yes]

    Justification: The paper includes comprehensive documentation for all new assets introduced. This documentation is provided alongside the assets, ensuring that users have access to detailed descriptions, usage guidelines, and any necessary supporting information.

    Guidelines:

    - The answer NA 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: The paper does not include the full text of instructions given to participants, screenshots, or details about compensation.

    Guidelines:

    - The answer NA 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: [No]

    Justification: We request several volunteers to participate in a simple anonymous questionnaire for user study, with no potential risks involved for the volunteers.

    Guidelines:

    - The answer NA 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 rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer NA 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 (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.

Appendix

A Algorithm of Spatial Sparse Attention

In our Triton-based implementation of the spatial blockwise selection attention kernel, two key challenges arise within sparse 3D voxel structures: 1) the number of tokens varies across different blocks, and 2) tokens within the same block may not be contiguous in HBM. To address these, we first sort the input tokens based on their block indices, then compute the starting index $\mathcal{C}$ of each block as kernel input. In the inner loop, $\mathcal{C}$ dynamically governs the loading of corresponding block tokens. The complete procedure of forward pass is formalized in Algorithm 1.

Algorithm 1 Spatial Blockwise Selection Attention Forward Pass

Require: $\mathbf{q} \in \mathbb{R}^{N\times(h_{kv}\times h_s)\times d}$, $\mathbf{k} \in \mathbb{R}^{N\times h_{kv}\times d}$, $\mathbf{v} \in \mathbb{R}^{N\times h_{kv}\times d}$, number of key/value heads h_{kv}, number of the shared heads h_s, number of the selected blocks T, indices of the selected blocks $\mathbf{I} \in \mathbb{R}^{N\times h_{kv}\times T}$, the number of divided key/value blocks N_b, index of the first token in each block $\mathcal{C} \in \mathbf{R}^{N_b+1}$, block size B_k.

Divide the output $\mathbf{o} \in \mathbb{R}^{N\times(h_{kv}\times h_s)\times d}$ into (N, h_{kv}) blocks, each of size $h_s \times d$. Divide the logsumexp $l \in \mathbb{R}^{N\times(h_{kv}\times h_s)}$ into (N, h_{kv}) blocks, each of size h_s.
Sort all tokens within $\mathbf{q}$, $\mathbf{k}$ and $\mathbf{v}$ according to their respective block indices.
for $t = 1$ **to** N **do**
  for $h = 1$ **to** h_{kv} **do**
    Initialize $\mathbf{o}_{t,h} = (0)_{h_s\times d} \in \mathbb{R}^{h_s\times d}$, logsumexp $l_{t,h} = (0)_{h_s} \in \mathbb{R}^{h_s}$, and $\mathbf{m}_{t,h} = (-\text{inf})_{h_s} \in \mathbb{R}^{h_s}$.
    Load $\mathbf{q}_{t,h} \in \mathbb{R}^{h_s\times d}$, $\mathbf{I}_{t,h} \in \mathbb{R}^{T}$ from HBM to on-chip SRAM.
    for $j = 1$ **to** T **do**
      Load the starting token index $b_s = \mathcal{C}^{(\mathbf{I}^{(j)}_{t,h})}$ and ending token index $b_e = \mathcal{C}^{(\mathbf{I}^{(j)}_{t,h})+1} - 1$ of the $\mathbf{I}^{(j)}_{t,h}th$ block from HBM to on-chip SRAM.
      for $i = b_s$ **to** b_e **by** B_k **do**
        Load $\mathbf{k}_i$, $\mathbf{v}_i \in \mathbb{R}^{B_k\times d}$ from HBM to on-chip SRAM.
        Compute $\mathbf{s}^{(i)}_{t,h} = \mathbf{q}_{t,h}\mathbf{k}^T_i \in \mathbb{R}^{h_s\times B_k}$.
        Compute $\mathbf{m}^{(i)}_{t,h} = \max(\mathbf{m}_{t,h}, \text{rowmax}(\mathbf{s}^{(i)}_{t,h})) \in \mathbb{R}^{h_s}$.
        Compute $\mathbf{p}^{(i)}_{t,h} = e^{\mathbf{s}^{(i)}_{t,h} - \mathbf{m}^{(i)}_{t,h}} \in \mathbb{R}^{h_s\times B_k}$.
        Compute $\mathbf{o}_{t,h} = e^{\mathbf{m}_{t,h} - \mathbf{m}^{(i)}_{t,h}}\mathbf{o}_{t,h} + \mathbf{p}^{(i)}_{t,h}\mathbf{v}_i$.
        Compute $l_{t,h} = \mathbf{m}^{(i)}_{t,h} + \log(e^{l_{t,h} - \mathbf{m}^{(i)}_{t,h}} + \text{rowsum}(\mathbf{p}^{(i)}_{t,h}))$, $\mathbf{m}_{t,h} = \mathbf{m}^{(i)}_{t,h}$.
      end for
    end for
    Compute $\mathbf{o}_{t,h} = e^{\mathbf{m}_{t,h} - l_{t,h}}\mathbf{o}_{t,h}$.
    Write $\mathbf{o}_{t,h}$ and $l_{t,h}$ to HBM as the (t, h)-th block of $\mathbf{o}$ and l, respectively.
  end for
end for
Return the output $\mathbf{o}$ and the logsumexp l.

B Implementation Details

Table 3: The training configuration of our SS-DiT.

Resolution	Token Count	Learning Rate	Batch Size	Training Time
256^3	≈2058	1e-4	8×8	2 days
384^3	≈5510	1e-4	8×8	2 days
512^3	≈10655	5e-5	8×8	2 days
1024^3	≈45904	2e-5	2×8	1 day

VAE. We utilize active voxels from volumes with SDF values less than $\tau = \frac{1}{128}$ as inputs to the SS-VAE. The downsampling factor f for the encoder is set to 8, and the channel dimension of the latent representation $\mathbf{z}$ is configured to 16. The weights for the various losses are set as: $\lambda_{\text{in}} = 1.0$, $\lambda_{\text{ext}} = 1e-1$, $\lambda_{\text{sharp}} = 1.0$, and $\lambda_{\text{KL}} = 1e-3$. We employ the AdamW [24] optimizer with an initial learning rate of $1e-4$. To enhance training efficiency, we first conduct multi-resolution training using SDF volumes at three resolutions of $\{256^3, 384^3, 512^3\}$ over a period of one day on 8 A100 GPUs, with a batch size of 4 per GPU. Subsequently, we fine-tune the SS-VAE for one additional day at 1024^3 resolution with a learning rate of $1e-5$ with a batch size of 1 per GPU.

DiT. Our SS-DiT comprises 24 layers of DiT blocks with a hidden dimension of 1024. We employ Grouped-Query Attention (GQA) [4] with a group number set to 2, where each group contains 16 attention heads. The hidden dimension of each head is configured as 32. For the spatial sparse attention (SSA) mechanism, we configure the resolution of the compression blocks to $m_{\text{cmp}} = 4$, the resolution of the selection blocks to $m_{\text{slc}} = 8$, and the size of the sparse 3D windows $m_{\text{win}} = 8$. We utilize DINO-v2 Large [28] to extract features from conditional images, with input images having a resolution of 518×518. For the DiT, we implement a progressive training strategy that gradually increases the resolution from 256^3 to 1024^3 to accelerate convergence. Tab 3 presents the average latent token count, learning rate, batch size, and training duration settings at different resolutions. We employ the AdamW optimizer and trained the model for a total of 7 days on 8 A100 GPUs. For the 1024^3 resolution, we further filtered 68k high-fidelity 3D assets for training. Additionally, similar to Trellis [39], we trained an extra DiT to predict the indices of the sparse latent tokens $\mathbf{z}$, which was trained for 7 days on 8 A100 GPUs.

C User Study for Image-to-3D Generation

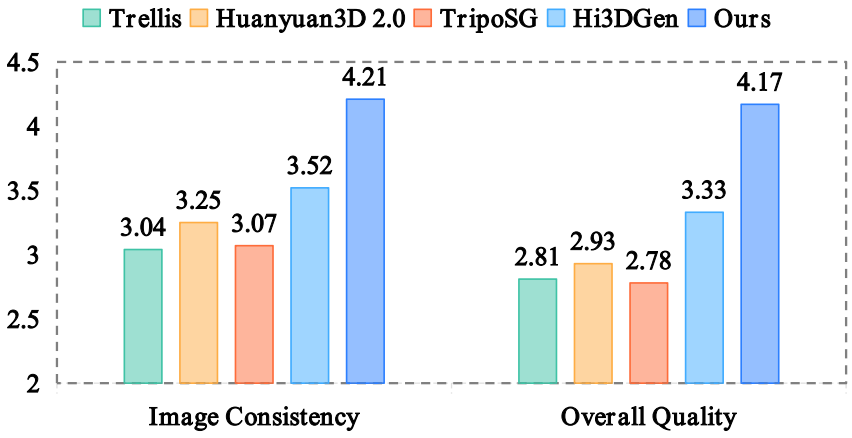

Figure 7: User study for image-to-3D generation.

We conducted a user study with 40 participants evaluating 75 unfiltered meshes generated by our Direct3D-S2 and other image-to-3D methods. Participants scored each output using two criteria: image consistency and overall geometric quality, with scores ranging from 1 (poorest) to 5 (excellent). As shown in Fig 7, Our Direct3D-S2 demonstrates statistically superiority over other approaches across both evaluation metrics.

D Comparison of VAE

To validate the reconstruction quality of our SS-VAE, we curated a challenging validation set from the Objaverse [9] dataset, comprising meshes with complex geometric structures. Qualitative and quantitative comparisons with competing methods are shown in Fig 8 and Tab 4. It can be observed our SS-VAE achieves superior reconstruction accuracy at 512^3 resolution, and demonstrates markedly improved performance on complex geometries at 1024^3 resolution. Notably, thanks to our fully end-to-end SDF reconstruction framework, SS-VAE requires only 2 days of training on 8 A100 GPUs, significantly fewer than competing methods that typically demand at least 32 GPUs for equivalent training durations.

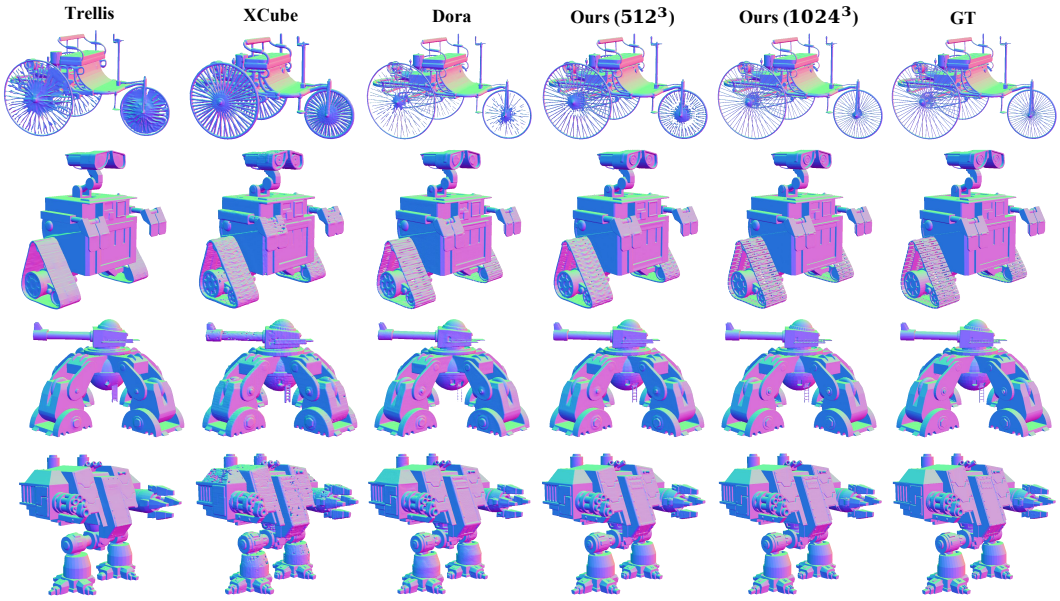

Figure 8: Qualitative comparisons of VAE reconstruction results. Note that we used a latent token length of 4096 during the inference of Dora [6].

Table 4: Quantitative comparisons of meshes reconstructed by different VAE.

Methods	**Chamfer Distance** ↓	**F1 (0.005)** ↑	**F1 (0.001)** ↑
Trellis [39]	28.60	88.77	11.57
Xcube [31]	15.42	99.79	31.85
Dora [6]	15.12	99.98	31.86
Ours (512)	13.59	99.98	41.11
Ours (1024)	**7.91**	**100.0**	**70.19**

E Additional Ablation Studies

Table 5: Quantitative comparisons of our method at various resolution.

Resolution	**Normal-FID** ↑
256	132.97
384	87.10
512	69.31
1024	**46.44**

Results of Image-to-3D Generation in Different Resolution. We present the generation results of our Direct3D-S2 across four resolutions $\{256^3, 384^3, 512^3, 1024^3\}$ in Fig 9 and in Tab 5. The results demonstrate that increasing resolution progressively improves mesh quality. At lower resolutions 256^3 and 384^3, the generated meshes exhibit limited geometric details and misalignment with input images. At 512^3 resolution, the meshes display significantly enhanced high-frequency geometric details. Further increasing the resolution to 1024^3 yields meshes with sharper edges and improved alignment with input image details.

Effect of Each Module in SSA. We validated the effect of the three modules in SSA at resolution 512^3, with the results presented in Fig 10 and Tab 6. When using only the sparse 3D window module (*win*), the generated meshes exhibited detailed structures but suffered from surface irregularities due to the lack of global context modeling. Introducing the sparse 3D compression module (*win+cmp*) showed minimal performance changes, which is reasonable as this module primarily serves to obtain

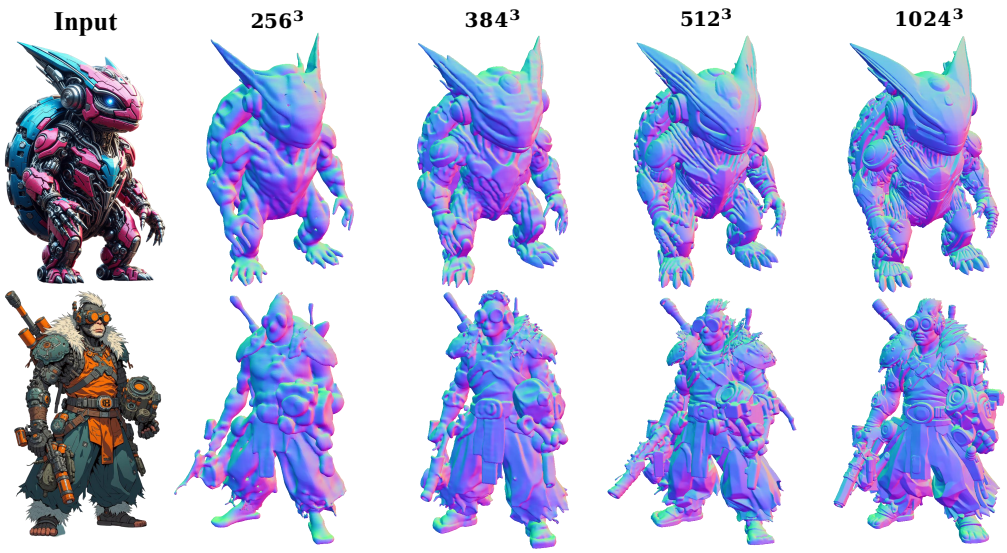

Figure 9: The visualization results of our Direct3D-S2 for image-to-3D generation across four resolutions: $\{256^3, 384^3, 512^3, 1024^3\}$.

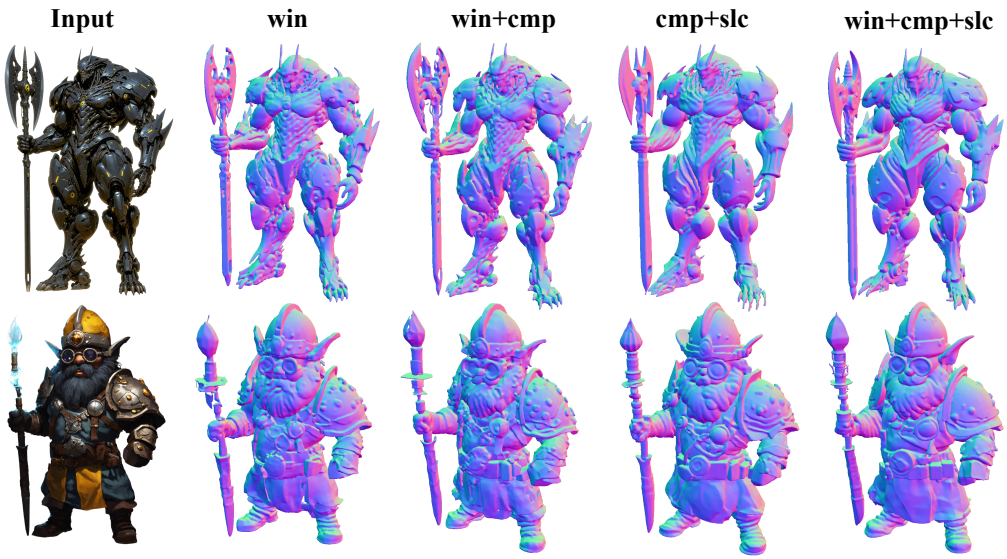

Figure 10: Ablation studies for the three modules of SSA, where *win*, *cmp*, *slc* denote the sparse 3D window, sparse 3D compression, and spatial blockwise selection modules, respectively.

Table 6: Ablation results for different modules of SSA at 512^3 resolution.

Modules	**Normal-FID** ↑
win	74.10
win+cmp	77.68
cmp+slc	70.05
win+cmp+slc	**69.31**

the attention scores for the blocks. After incorporating the spatial blockwise selection module (*win+cmp+slc*), the model can focus on the most important regions globally, resulting in a notable improvement in mesh quality. We also observed that not utilizing the window (*cmp+slc*) did not

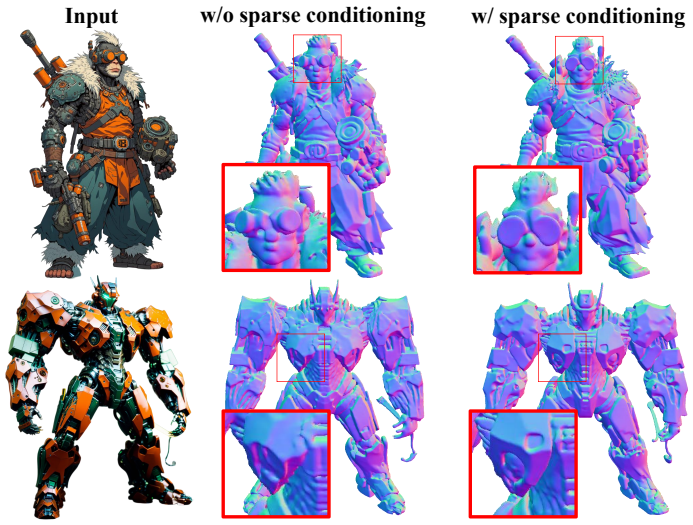

Figure 11: Ablation studies for sparse conditioning mechanism.

result in a significant drop in model performance, but slowed convergence, demonstrating that local feature interaction contributes to more stable training and enhances convergence speed.

Table 7: Ablation results for sparse conditioning at 512^3 resolution.

Methods	**Normal-FID** ↑
Ours w/o sparse conditioning	71.49
Ours w/ sparse conditioning	**69.31**

Effect of Sparse Conditioning Mechanism. We perform ablation experiments to validate the effect of the sparse conditioning mechanism at 512^3 resolution. As demonstrated in Fig 11 ann Tab 7, the exclusion of non-foreground conditioning tokens through sparse conditioning enables the generated meshes to achieve significantly better alignment with the input images.

F Additional Qualitative Comparisons for Image-to-3D Generation

We provide more qualitative comparisons with both open-source and closed-source models in Fig 12 and Fig 13.

G Limitations

Our proposed spatial sparse attention mechanism achieves significant speed improvements over FlashAttention-2. However, the forward pass exhibits a notably smaller acceleration ratio compared to the backward pass. This discrepancy primarily stems from the computational overhead introduced by top-k sorting operations during the forward pass. We acknowledge this limitation and will prioritize optimizing these operations in future work.

Figure 12: Qualitative comparisons between other open-source image-to-3D methods and our approach.

Figure 13: Qualitative comparisons between other open-source, closed-source image-to-3D models and our approach. Note that all the closed-source model we use the default setting of their web app. ***Best viewed with zoom-in.***