SF3D: Stable Fast 3D Mesh Reconstruction with UV-unwrapping and Illumination Disentanglement

Mark Boss[1] Zixuan Huang[1,2†] Aaryaman Vasishta[1] Varun Jampani[1]

[1]Stability AI [2]UIUC

Figure 1. **SF3D Creates High-quality Object Meshes from Single Images** with Materials, Delighting and UV-unwrapped textured meshes in 0.5 s. Here, we show sample SF3D results from different input images. SF3D handles both realistic and non-realistic styles well.

Abstract

We present SF3D, a novel method for rapid and high-quality textured object mesh reconstruction from a single image in just 0.5 seconds. Unlike most existing approaches, SF3D is explicitly trained for mesh generation, incorporating a fast UV unwrapping technique that enables swift texture generation rather than relying on vertex colors. The method also learns to predict material parameters and normal maps to enhance the visual quality of the reconstructed 3D meshes. Furthermore, SF3D integrates a delighting step to effectively remove low-frequency illumination effects, ensuring that the reconstructed meshes can be easily used in novel illumination conditions. Experiments demonstrate the superior performance of SF3D over the existing techniques.

Project page with code and model: https://stable-fast-3d.github.io

1. Introduction

High-quality object meshes are essential for various use cases in movies, gaming, e-commerce, and AR/VR. In this work, we tackle the problem of generating high-quality 3D object mesh from a single image. This is a ill-posed and challenging problem as this requires reasoning about the object's 3D shape and texture from only a single 2D projection (image) of that object. Single-image object generation can simplify the tedious and manual object creation process.

The quality of object mesh generation from a single image has dramatically improved in the past couple of years

[†]Work done during internship at Stability AI.

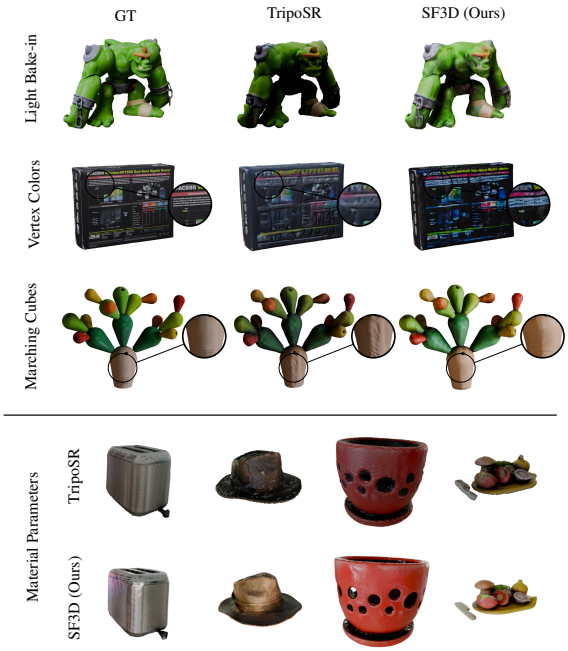

Figure 2. **SF3D Improvements on Different Prevalent Issues.** Here, we compare our results with TripoSR [54]. The top shows the effect of light bake-in when relighting the asset. SF3D produces a more plausible relighting. By not using vertex colors, our method is capable of encoding finer details while also having a lower polygon count. Our vertex displacement enables estimating smooth shapes, which do not introduce stair-stepping artifacts from marching cubes. Lastly, our material property prediction allows us to express a variety of different surface types.

with the advances in transformer models [18, 20, 54], large synthetic datasets [11] and 3D-aware image/video generative models [31, 51, 57, 67]. Especially, the transformer-based reconstruction models [18, 20, 54] demonstrate phenomenal generalization capabilities on real-world images despite being trained only on synthetic datasets while also generating 3D assets from a single image in under 1 s.

Despite this rapid progress, several issues remain in these feed-forward fast 3D reconstruction models [18, 20, 54]. These techniques often produce 3D assets that are not usable for downstream applications or require laborious manual post-processing. We identify several key issues in these techniques and propose a fast generation technique called 'Stable Fast 3D' (SF3D) that generates higher quality and more usable 3D assets from single images, while also retaining the fast generation speed within 0.5 seconds on a H100 GPU. Next, we briefly introduce these issues and how we tackle those in SF3D.

Light Bake-in. Having shadows or other illumination effects in a given input image is common. Most existing works bake these effects into textures, making the resulting 3D assets less usable. Having consistent lighting helps in easy integration into graphics pipelines. In SF3D, we propose decomposing the illumination and reflective properties by incorporating explicit illumination and a differentiable shading model using Spherical Gaussians (SG). Fig. 2 (top row) shows a sample result of SF3D, where the light bake-in is considerably reduced compared to the prior art.

Vertex Coloring. Another issue we found in most 3D generation models is that they produce meshes with a high vertex count, using vertex coloring to represent object texture. This makes the resulting 3D assets inefficient to use in applications such as games. A key issue is the additional processing time of UV unwrapping, which can take longer than the entire object generation. For example, xatlas [72] and geogram [27], can take up to 30 s or 10 s for a single asset, respectively. To tackle this, we propose a highly parallelizable fast box projection-based UV unwrapping technique to achieve a 0.5 s generation time. The effect of relying on vertex coloring *vs*. UV Unwrapping can be seen in Fig. 2 (middle row), where TripoSR captures fewer details than SF3D despite requiring 10× higher polygon count.

Marching Cubes Artifacts. The feed-forward networks often create volumetric representations such as Triplane-NeRFs [9] which are converted to meshes using the Marching Cubes (MC) [35] algorithm. MC can cause 'stair-stepping' artifacts, which can be somewhat reduced by increasing the volume resolution. However, this comes at the cost of large computational overhead. In contrast, SF3D uses more efficient architecture for higher resolution triplanes and also produces meshes using DMTet [46] with learned vertex displacements and normal maps, resulting in smoother mesh surfaces. Fig. 2 demonstrates the smoothness of the SF3D mesh compared to that of TripoSR.

Lack of Material Properties. The generations from previous feed-forward techniques often look dull when they are rendered using different illuminations. This is mainly due to the lack of explicit material properties in the output generations, which can influence the light reflection. To tackle this, we predict non-spatially varying material properties. This addition is apparent when rendering different generated objects in Fig. 2 (bottom row).

With these advances, SF3D can generate high-quality 3D meshes from a single image with several desirable properties for downstream applications on both shapes (low-polygon yet smooth) and textures (Illumination disentangled UV maps with material properties). The 3D assets are small in size (under 1 MB) and can be created in 0.5 s. For text-to-3D mesh generation, a fast Text-to-Image (T2I) model [45] can be combined with SF3D to produce meshes in about 1 s. Experimental results demonstrate higher quality results with SF3D compared to the existing works. In short, SF3D provides a comprehensive technique for fast, high-quality 3D object generation from single images, addressing both speed and usability in practical applications.

2. Related Work

3D Reconstruction using Image Generative Priors. Diffusion models [19, 50] have proven to be powerful generative models for various tasks [2, 3, 15, 43, 44, 56]. Several works such as Zero123 [31] and others [11, 25, 51, 80] leverage the object priors in these diffusion models by adapting the generative models for 3D generation. Score Distillation Sampling (SDS) [41] is often used to optimize 3D representation using the 2D diffusion models. However, it was found that relying solely on the image prior does not always produce consistent multi-view results. This issue is improved in follow-up works [32, 37, 47, 49] by simultaneously generating multiple views of an object. Another approach is to explicitly introduce 3D awareness [29, 30, 71] or use a multi-view diffusion process [2, 14, 26, 33, 34, 36, 49, 57, 62, 79] to generate 3D objects. While diffusion models can generate videos or multi-view images relatively quickly, they require a 3D reconstruction step to create 3D mesh from a single image. Even with fast techniques, generating an object can still take several minutes. Our work focuses on fast generation speeds in $0.5\,\mathrm{s}$ for image-to-3D.

Feed-Forward 3D Reconstruction. Recently, LRM [20] and the follow-up works [18, 54] demonstrated that the fast and reliable 3D generation is possible using feed-forward networks trained on large synthetic datasets. These works use large transformer models to directly predict triplanes as the volume representations, which can be ray-marched using NeRF [38]. This allows these models to train on multi-view datasets, as only the rendering loss is required for training. A set of follow-up works address to remedy the reliance on multi-view datasets [23, 58, 65]. This can produce meshes from images in seconds compared to the generative-prior based approaches.

Several follow-up works emerged which use Gaussian Splatting [24] as the representation [52, 53, 70, 76, 82] or directly integrate a mesh prediction [60, 63, 66, 68, 73, 81]. Another group of methods integrates diffusion models with the feed-forward models by either directly generating the triplanes [59], using LRM as the denoiser [69], or using it as the conditioning [61]. As single-image reconstruction is a challenging task, several models leverage the extensive prior of multi-view diffusion models to generate multiple views of an object which the feed-forward model then uses to produce a 3D output [28, 53, 60, 63, 66, 68, 70, 73, 76, 81]. Generally, these models learn the scene's radiance, meaning lighting information is baked into the objects. LDM [66] tries to remedy this by learning an additional shading color to capture the shading information. However, training requires having access to the ground-truth albedo color as a supervision signal. Compared to LDM, our method also learns an illumination model, enabling training on regular multi-view datasets, and we predict more material properties. We also tackle fast UV unwrapping, so unlike previous works, we do not have to rely on vertex colors while keeping the 3D generation time short.

Material Decomposition. Predicting only the radiance of an object has the downside that relighting does not produce convincing results. Current works in single scene optimization are often based on NeRF [38] or Gaussian Splatting [24] and decompose multiple input images (> 50) into light and materials [4–6, 13, 17, 39, 75]. These methods often predict materials properties of a Physically Based Rendering (PBR) [7] shading model. A few recent works tackle joint 3D shape and material generation, such as UniDream [33] or Fantasia3D [10] by optimizing the material properties using a SDS loss. However, this optimization takes several hours to converge. Another set of works aims to texture existing objects [55, 78] using diffusion models. Our work generates a textured object with homogeneous material properties from a single image under natural illumination at fast generation speeds.

3. Method

We propose SF3D, which converts a single object image into a textured and UV-unwrapped 3D model with de-lit albedo and material properties. As explained in the introduction section, with SF3D, we aim to fix the issues shown in Fig. 2 and introduce additional quality enhancements.

Preliminaries. Our method is based on TripoSR [20, 54], which trains a large transformer-based network that outputs a Triplane [9] based 3D representation from a single image. TripoSR is trained with multi-view image datasets without explicit 3D supervision. In TripoSR, the image is encoded using DINO [8] and passed through a Transformer network to generate a 3D triplane at a resolution of 64×64. The triplane features are then decoded into RGB colors and rendered using standard NeRF [38] rendering into multiple views for training. TripoSR only learns the view-independent colors and cannot model reflective objects. Several issues of the TripoSR (and other similar networks) are explained in the introduction (Fig. 2).

SF3D Overview. With SF3D, we propose several improvements to TripoSR [54] to improve the output quality in different aspects. As illustrated in Fig. 3, SF3D has 5 main components: 1. An enhanced transformer network that predicts higher resolution triplanes, which helps in reducing aliasing artifacts (top left in the figure); 2. A material estimation network (bottom left) predicts material properties, which helps handle an object's reflective properties. 3. Illumination prediction (bottom right) to tackle illumination disentangling, which helps in outputting homogeneous objects without shadows; 4. Mesh extraction and refinement with the prediction of vertex offsets and surface normals (top right), which helps in smoother output shapes

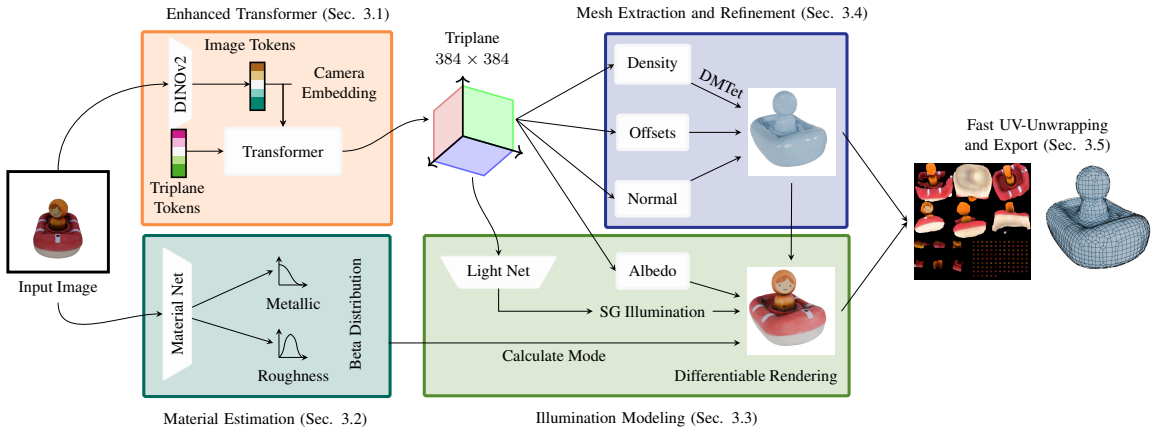

Figure 3. **SF3D Overview.** SF3D improves on TripoSR by addressing the issues in Fig. 2 with 5 novel modules: 1. An enhanced transfomer for higher resolution triplanes (top left); 2. Material estimation with Material Net (bottom left); 3. Explicit illumination estimation using Light Net (bottom right); 4. Smooth mesh extraction with the estimation of vertex offsets and normals (top right); and finally 5. an export pipeline with fast UV-unwrapping (right).

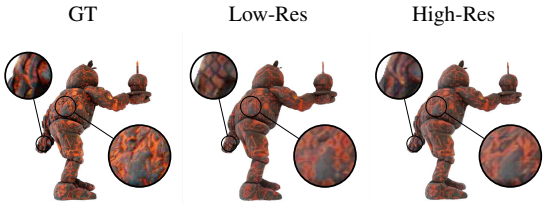

Figure 4. **Triplane Resolution Aliasing.** We found that low-resolution triplanes struggle with high-frequency and high-contrast textures and produce grid-like aliasing artifacts. Our method increases the triplane size from 64^2 to 384^2 allowing our method to reproduce these textures with fewer artifacts.

with fewer mesh extraction artifacts; and 5. A fast UV-unwrapping and export module (right) that helps produce low-poly meshes and high-resolution textures. Next, we explain each of these modules in detail.

3.1. Enhanced Transformer

First, as illustrated in Fig. 3, we transitioned from DINO [8] used in TripoSR to the improved DINOv2 [40] network to obtain image tokens for the transformer. We observed that low-resolution (64×64) triplanes used in TripoSR and other networks [18, 20] introduce noticeable artifacts, especially in scenarios with high-frequency and high-contrast texture patterns. Fig. 4 (Middle) illustrates these aliasing artifacts. The triplane resolution is directly correlated with the presence of these artifacts, which we identify as an aliasing issue that can be mitigated by increasing the resolution. The increased capacity also improves the geometry.

Naively increasing the triplane resolution quadratically increases the transformer complexity. We take inspiration from the recent PointInfinity [22] work and propose an enhanced Transformer network that outputs higher-resolution triplanes. PointInfinity proposes an architecture where the complexity remains linear concerning the input size by avoiding the self-attention on the higher resolution triplane tokens. With this addition, we produce 96×96 resolution triplanes with 1024 channels. We further increased the triplane resolution by shuffling the output features across dimensions, resulting in 40-channel features at 384×384 resolution. Further details about the architecture are available in the supplements. Fig. 4 (Right) shows fewer aliasing artifacts with our higher resolution triplanes.

3.2. Material Estimation

To enhance the output mesh appearance for the reflective objects, SF3D also outputs the material properties of metallic and roughness parameters. Ideally, one would like to estimate the spatially varying material properties at 3D output locations, but this is an inherently challenging and ill-posed learning problem that requires many high-quality 3D data with spatially varying materials. To overcome these challenges, we propose simplifying the material estimation problem by estimating a single metallic and roughness value for the entire object. Although this non-spatially varying material mainly applies to homogeneous objects, we find that it significantly improves the visual quality of our mesh predictions. Specifically, as illustrated in Fig. 3, we propose 'Material Net' that predicts the metallic and roughness values from the input image.

For pre-training the Material Net, we selected a subset of 3D objects with PBR material properties from the synthetic training dataset and rendered them under different illuminations and viewpoints. We observe that directly regressing the material values often leads to training collapse, where the network always predicts a roughness value of 0.5 and

a metallic value of 0. As a remedy, we propose a probabilistic prediction approach, where we predict the parameters of a Beta distribution and minimize the log-likelihood during training. This stabilizes the training by allowing for uncertainty in this ambiguous material estimation task and prevents the collapse observed with direct regression. During inference and training of SF3D, we do not sample the distribution but calculate the mode of the distribution.

We implement the Material Net by first passing the image through the frozen CLIP image encoder [42] to extract semantically meaningful latents and pass them through 2 separate MLPs with 3 hidden layers and 512 width to output the parameters for the distributions.

3.3. Illumination Modeling

We propose explicitly estimating the illumination in the input image to account for varying shadings (e.g., shadows). Otherwise, the 3D outputs would have baked-in illumination effects into their RGB colors, as illustrated in Fig. 2. To this end, we propose a *Light Net* (Fig. 3 bottom right) that predicts the spherical Gaussian (SG) illumination map from the estimated triplanes. The rationale here is that the triplanes encode the global structure and appearance of the input object and should account for the 3D spatial relationships and changes in illumination over the object surface. We use the 96×96 resolution triplanes from the transformer and pass them through 2 CNN layers, followed by a max pool and final MLP with three hidden layers and a feature dimension 512 for all layers. Light Net outputs the grayscale amplitude values for 24 SGs with a Softplus activation to ensure positive values. The axis and sharpness values for these SGs remain fixed and are set up to cover the entire sphere. These amplitude values allow us to implement a deferred physically based rendering approach similar to that used in NeRD [4].

Our method also incorporates a lighting demodulation loss $\mathcal{L}_{\text{Demod}}$ during the training phase, inspired by the works of Hasselgren *et al*. [17] and Voleti *et al*. [57]. This loss function ensures that the lighting on an object with an entirely white albedo closely matches the luminance of the input image. The demodulation loss enforces consistency between the learned illumination and the lighting conditions observed in the training data. This can be seen as a bias to resolve the ambiguity between appearance and shading [1].

3.4. Mesh Extraction and Refinement

Fig. 3 (top right) illustrates this module. We convert the estimated triplanes into a mesh using a differentiable Marching Tetrahedron (DMTet) [46] technique. As explained in the introduction (Fig. 2), Marching Cubes (MC) usually results in several staircase artifacts on the resulting meshes. We propose two new MLP heads to refine the meshes as a remedy. One predicts vertex offsets $\boldsymbol{v}_o \in \mathbb{R}^3$, and another one predicts the world space vertex normals $\hat{\boldsymbol{n}} \in \mathbb{R}^3$. Inspired by MeshLRM [60], we also implemented small split decoder MLPs for these two networks, which proved beneficial for performance and efficiency. We found that the vertex offsets can reduce artifacts from the tetrahedral grids, and the world space normals can add details to the flat mesh triangles. Given that the normal predictions are initially unreliable, we stabilize the training by using spherical linear interpolation (slerp) between the geometry normals $\boldsymbol{n} \in \mathbb{R}^3$ and our predictions. This slerp is used during the initial 5K training steps.

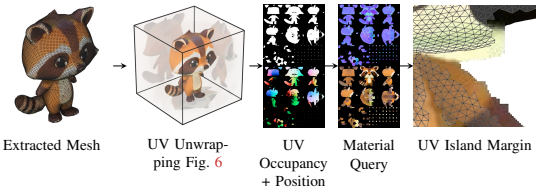

Figure 5. **Export Pipeline.** Our export process starts with the mesh and is followed by UV unwrapping, occupancy and world position baking, material querying, and UV island margins.

To regularize the mesh estimation, we use several training losses: a normal consistency loss $\mathcal{L}_{\text{Nrm consistency}}$, a Laplacian smoothness loss $\mathcal{L}_{\text{Laplacian}}$ as implemented by threestudio [16], and a vertex offset regularization $\mathcal{L}_{\text{Offset}} = \boldsymbol{v}_o^2$. For supervising the normal prediction, we use a geometry normal replication loss $\mathcal{L}_{\text{Nrm repl}} = 1 - \boldsymbol{n} \cdot \hat{\boldsymbol{n}}$, where $\cdot$ is the dot product and a normal smoothness loss to ensure the smoothness of normal predictions in 3D. This loss is achieved by adding a small offset $\boldsymbol{\epsilon} \in \mathbb{R}^3$ around a query location $\boldsymbol{x} \in \mathbb{R}^3$. The loss is then defined as $\mathcal{L}_{\text{Nrm smooth}} = (\hat{\boldsymbol{n}}(\boldsymbol{x}) - \hat{\boldsymbol{n}}(\boldsymbol{x} + \boldsymbol{\epsilon}))^2$.

3.5. Fast UV-Unwrapping and Export

The final stage of SF3D is an export pipeline that outputs the final 3D mesh along with the corresponding UV atlas. Our export pipeline follows multiple stages to ensure efficient and effective handling of 3D models. An overview of these stages is provided in Fig. 5, where we first do fast UV-unwrapping. We then bake the world positions and occupancy to the UV atlas, which we use for querying the albedo and normal. This results in the final textured 3D mesh. The entire export process only takes 150 ms.

UV unwrapping is traditionally a computationally intensive process. Existing methods require several seconds for UV unwrapping, which is impractical when we are aiming for sub-second generation speeds. To address this inefficiency, we propose a Cube projection-based unwrapping method. The key advantage of this approach is it is parallelizable: each face of the mesh can independently decide which cube face to project onto, based on its surface normal.

Our UV unwrapping process is illustrated in Fig. 6. We initially align the output mesh based on the most dominant axes with the cube projection coordinate system. After each

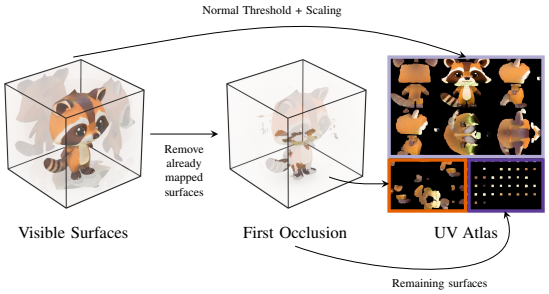

Figure 6. **UV Unwrapping.** Our UV unwrapping technique uses projection mapping, allowing each face to independently select a projection, enabling easy parallelization. A naive approach could lead to the same UV coordinates being assigned to different vertices due to occlusions. We identify potential overlaps from occlusions on the 2D mapped surfaces and relocate them to different areas within the UV atlas. Any remaining areas are placed in the bottom right of the UV atlas. This method minimizes distortion and ensures most surfaces are preserved in a connected area.

mesh face selects the appropriate cube direction, we address potential occlusions. Without managing occlusions, different faces could share the same UV coordinates, leading to artifacts in the texture. We detect occlusions in the UV atlas by performing 2D triangle-triangle intersection tests. We filter triangles by their proximity to the triangle centers to make the process efficient. If an intersection is detected, we sort the intersecting triangles based on their depth in the plane, keeping the first intersection and marking the others for reassignment to different UV atlas areas. The first intersection is placed in the top third of the UV atlas and the second intersection is placed in the bottom left area. The remaining triangles are organized into a grid in the bottom right section of the atlas. We also rotate each island to minimize the shading seams by following radial z tangent orientation. We then assign each face to a position in the UV atlas, as illustrated in Fig. 6.

Next, we bake the world positions and the occupancy data with UV-unwrapping into the final UV atlas. This allows us to query the world positions within the chart from our triplanes and decode the albedo and surface normals into additional textures. We transform the world-space normal map into a tangent-space normal map using the tangent and bitangent vectors. We add margins to the UV atlases to prevent visible seams at UV island borders. This is achieved through an iterative process: in each iteration, we perform a 3×3 partial convolution based on the occupied areas, using the valid regions of the kernel. We then use a 3×3 max pooling operation to expand the occupied regions of the UV atlas, placing the mean values in the newly expanded areas while preserving the original regions. This iterative extension ensures that the textures smoothly blend outwards.

We incorporate our image estimator's metallic and roughness values and pack everything into a GLB file, ready for efficient rendering and use in various applications.

3.6. Overall Training and Loss Functions

Directly training our method with mesh rendering yielded unsatisfactory results. Hence, we pre-trained it on the NeRF task. Following this pre-training, we transitioned to mesh training, replacing the NeRF rendering with differentiable mesh rendering and SG-based shading. Given the introduction of light estimation, we found that using larger batch sizes aids convergence. We initiate training with a batch size of 192 and a rendering resolution of 128×128, training for 10K steps. In the subsequent stage, we reduce the batch size to 128 and increase the resolution to 256×256, continuing for 20K steps. The final stage involves 80K steps at a 512×512 resolution with a batch size of 96.

The loss functions remain consistent across all mesh training stages. We primarily use image-based metrics to compare our rendered and shaded reconstructions $\hat{\boldsymbol{I}}$ with the GT image $\boldsymbol{I}$. These include MSE $\mathcal{L}_{\text{MSE}}$ and LPIPS [77] $\mathcal{L}_{\text{LPIPS}}$ losses. We also incorporate a mask loss $\mathcal{L}_{\text{Mask}}$ between the GT mask M and the predicted opacity $\hat{M}$, defined as an MSE loss. We then define three loss formulations for the rendering, mesh regularization, and shading:

$$\mathcal{L}_{\text{render}} = \underbrace{\lambda_{\text{MSE}}}_{10}\mathcal{L}_{\text{MSE}} + \underbrace{\lambda_{\text{LPIPS}}}_{2}\mathcal{L}_{\text{LPIPS}} + \underbrace{\lambda_{\text{Mask}}}_{10}\mathcal{L}_{\text{Mask}} \tag{1}$$

$$\mathcal{L}_{\text{mesh}} = \underbrace{\lambda_{\text{Laplacian}}}_{0.01}\mathcal{L}_{\text{Laplacian}} + \underbrace{\lambda_{\text{Nrm Consistency}}}_{0.001}\mathcal{L}_{\text{Nrm consistency}} + \underbrace{\lambda_{\text{Offset}}}_{0.1}\mathcal{L}_{\text{Offset}} \tag{2}$$

$$\mathcal{L}_{\text{shading}} = \underbrace{\lambda_{\text{Nrm repl}}}_{0.2}\mathcal{L}_{\text{Nrm repl}} \underbrace{\lambda_{\text{Nrm smooth}}}_{0.02}\mathcal{L}_{\text{Nrm smooth}} + \underbrace{\lambda_{\text{Demod}}}_{0.01}\mathcal{L}_{\text{Demod}} \tag{3}$$

The total loss is defined as:

$$\mathcal{L} = \mathcal{L}_{\text{render}} + \mathcal{L}_{\text{mesh}} + \mathcal{L}_{\text{shading}} \tag{4}$$

4. Results

Datasets. For comparisons, we select GSO [12] and OmniObject3D [64] as our primary datasets for evaluation. We select 278 random scenes from GSO and 308 scenes from OmniObject3D for comparison. We render 16 views around the object and select a frontal as the conditioning view.

Baselines. We compare with several recent methods for fast 3D object reconstruction from a single image. We mainly focus on fast reconstruction models to maintain a consistent evaluation protocol across different techniques. We also mainly consider meshes as outputs and perform the evaluations on the mesh. We use the official implementation for all the baslines, and we evaluate all the methods under the same protocol. We selected several recent and concurrent works with source code releases for comparisons. Specifically, we compare against OpenLRM [18], TripoSR [54], LGM [53], CRM [59], InstantMesh [68], and

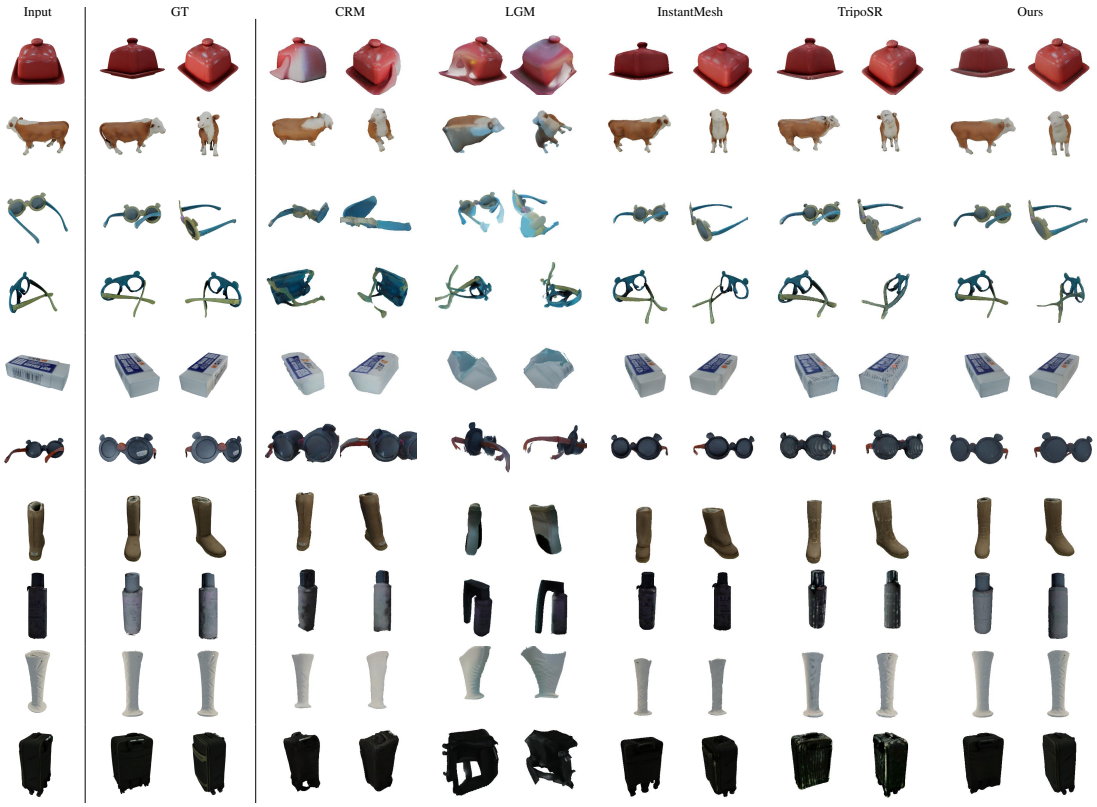

Figure 7. **Comparison on GSO and OmniObject3D.** Notice how our reconstructions produce consistent results with detailed textures and smooth shading.

		GSO				OmniObject			
Method	Time [s]↓	CD↓	FS@0.1↑	FS@0.2↑	FS@0.5↑	CD↓	FS@0.1↑	FS@0.2↑	FS@0.5↑
ZeroShape [21]	0.9	0.160	0.489	0.759	0.952	0.144	0.507	0.786	0.969
OpenLRM [18]	2.0	0.160	0.472	0.751	0.954	0.139	0.521	0.798	0.971
TripoSR [54]	**0.3**	0.111	0.645	0.869	0.980	0.103	0.672	0.889	0.986
LGM [53]	64.6	0.195	0.376	0.654	0.928	0.205	0.344	0.631	0.921
CRM [59]	10.2	0.179	0.411	0.699	0.945	0.158	0.469	0.752	0.960
InstantMesh [68]	32.4	0.138	0.549	0.801	0.967	0.138	0.560	0.811	0.964
SF3D (Ours)	0.5	**0.098**	**0.701**	**0.894**	**0.988**	**0.090**	**0.726**	**0.920**	**0.990**

Table 1. **Comparison on 3D Metrics Demonstrating State-of-the-art Performance of SF3D.** It is worth pointing out that all other methods produce meshes with drastically higher polygon counts. This allows them to follow the surfaces more closely than our low polygon meshes. Our meshes can outperform the others by using our learned vertex offsets. Our method is also one of fastest to generate a mesh from an image due to our efficient extraction pipeline.

	GSO			
Method	CD↓	FS@0.1↑	FS@0.2↑	FS@0.5↑
TripoSR [54]	0.111	0.645	0.869	0.980
SF3D w/o Enh. Transformer	0.108	0.660	0.872	0.982
SF3D (Ours)	**0.098**	**0.701**	**0.894**	**0.988**

Table 2. **Ablation.** As our model builds upon TripoSR, we use it as our baseline. Without our enhanced transformer, our mesh training has already improved upon it. With our architectural improvements, we outperform the baselines significantly.

ZeroShape [21]. For OpenLRM, we selected the large Objaverse 1.1 model. We use H100 GPU for comparisons.

Evaluations. For runtime comparisons, we consider the meshes as the final output and calculate the entire run time required to go from the input image to the final mesh. We then perform separate evaluations for the shape quality. Several models cannot be conditioned on intrinsic and extrinsic camera parameters, so we propose performing an alignment step. We normalize the mesh, perform brute-force rotation alignment tests, and select the rotation with the lowest Chamfer Distance (CD). We then run another

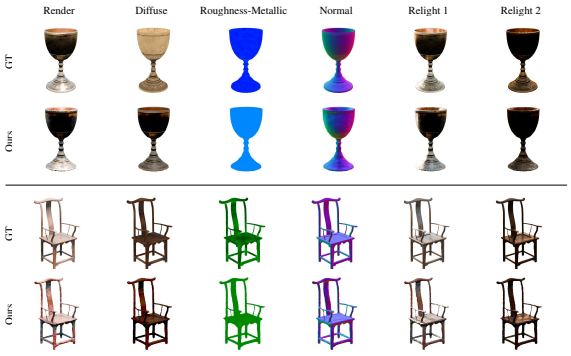

Figure 8. **Decomposition Results.** Here, we use high-quality objects from Polyhaven [74] and render them under natural illumination. These illuminations are highly challenging for material estimation. Still our model estimates sensible material properties, which allow for a convincing relighting.

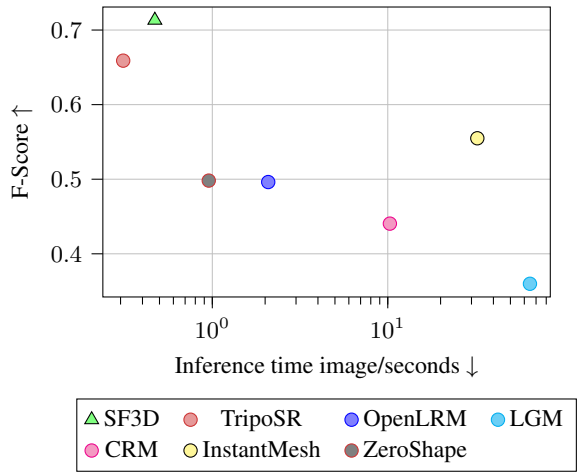

Figure 9. **Image-To-Mesh Time *vs*. Reconstruction Quality.** Our method is not only one of the fastest reconstruction methods, it is also capable of producing highly accurate geometry.

alignment step using Iterative Closest Point (ICP), where we further optimize the rotation and translation. We then calculate the standard shape metrics of CD and F-score (FS) after this alignment. This alignment might still misalign symmetric objects for re-rendering. Hence, we only report indicative rendering metrics in the supplements.

Triangle Counts. We run each method under the default configuration. As the triangle count varies drastically based on the 3D mesh, we report the triangle count for a single mesh here: InstantMesh 57.3K, CRM 24.1K, LGM 42.1K, TripoSR 32.1K, OpenLRM 662K, Ours 27.4K.

Results. In Table 1, we compare our method with all the baselines. Here, our method outperforms all current and concurrent baselines on both CD and F-scores. This indicates that our method can reconstruct accurate shapes, even if our models have fewer polygons than the other baselines. From the visual comparisons in Fig. 7, it can be seen that the accurate shape reconstruction of SF3D also translates well to the visual quality of the 3D assets. Here, it is worth noting that SF3D also handles fine geometry like the glasses well and produces consistent shapes with more detailed textures than the SOTA methods. Moreover, the results also show sensible material properties and albedo as seen in Fig. 8. This is highly challenging as the objects are only rendered under natural illumination. Estimating material properties without any knowledge about the illumination is a highly ambiguous problem.

Inference Speed *vs*. Reconstruction Quality. In Fig. 9, we plot the inference speed *vs*. reconstruction quality for different techniques. The best-performing methods should be located in the top left corner. While our method is slightly slower than TripoSR, the reconstruction accuracy is considerably better for SF3D. It is also worth noting that our reconstruction has smoother shading with less pronounced marching cube artifacts, as seen in Fig. 7. The final apparent asset quality is thus even higher for our method.

Ablations. We evaluate our additions in Table 2 against baseline models. Our method is built upon TripoSR, so we use it as our initial baseline. If we add mesh training and relighting as a fine-tuning step, we can see that 'SF3D w/o Enhanced Transformer' already improves upon TripoSR, demonstrating the use of our mesh-based training. This improvement is mainly due to the efficient rendering, enabling higher resolution supervision during training and smoother meshes from vertex offsets. If we add our high-resolution triplanes using our enhanced transformer, SF3D further outperforms the baseline considerably.

Limitations and Outlook. As seen in Fig. 7 (top row), the cup's albedo is not perfectly matched. This is related to the LDR input, where the dark spots do not contain any useful information. Furthermore, our roughness and metallic properties are homogeneous, which limits their usefulness for objects containing multiple drastically different materials that are spatially-varying. In addition, our method also introduces material prediction and delighting without explicit supervision. As we do not require explicit supervision of these parameters, our method could be extended to train on real-world datasets, which we leave for future work. Similarly, the UV unwrapping could leverage existing datasets for further improvements.

5. Conclusion

We present SF3D, a fast single view to uv-unwrapped textured object reconstruction method. In addition to our fast extraction pipeline, we introduce several architectural improvements to feed-forward-based 3D reconstruction methods, which help our model produce highly detailed geometry and texture. In our extensive evaluation, we show that our method outperforms existing and concurrent baselines in both speed and quality.

References

[1] E.H. Adelson and A.P. Pentland. *The perception of shading and reflectance*, page 409–424. Cambridge University Press, 1996. 5

[2] Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable Video Diffusion: Scaling latent video diffusion models to large datasets. *arXiv*, 2023. 3

[3] Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. Align your Latents: High-Resolution Video Synthesis with Latent Diffusion Models. *arXiv*, 2023. 3

[4] Mark Boss, Raphael Braun, Varun Jampani, Jonathan T. Barron, Ce Liu, and Hendrik P.A. Lensch. NeRD: Neural reflectance decomposition from image collections. *ICCV*, 2021. 3, 5

[5] Mark Boss, Varun Jampani, Raphael Braun, Ce Liu, Jonathan T. Barron, and Hendrik P.A. Lensch. Neural-pil: Neural pre-integrated lighting for reflectance decomposition. *NeurIPS*, 2021.

[6] Mark Boss, Andreas Engelhardt, Abhishek Kar, Yuanzhen Li, Deqing Sun, Jonathan T. Barron, Hendrik P.A. Lensch, and Varun Jampani. SAMURAI: Shape And Material from Unconstrained Real-world Arbitrary Image collections. *NeurIPS*, 2022. 3

[7] Brent Burley. Physically based shading at disney. *ACM Transactions on Graphics (SIGGRAPH)*, 2012. 3

[8] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. *ICCV*, 2021. 3, 4

[9] Eric R. Chan, Connor Z. Lin, Matthew A. Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas Guibas, Jonathan Tremblay, Sameh Khamis, Tero Karras, and Gordon Wetzstein. Efficient geometry-aware 3D generative adversarial networks. In *arXiv*, 2021. 2, 3

[10] Rui Chen, Yongwei Chen, Ningxin Jiao, and Kui Jia. Fantasia3D: Disentangling geometry and appearance for high-quality text-to-3D content creation. In *ICCV*, 2023. 3

[11] 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. *arXiv*, 2023. 2, 3

[12] Laura Downs, Anthony Francis, Nate Koenig, Brandon Kinman, Ryan Hickman, Krista Reymann, Thomas B McHugh, and Vincent Vanhoucke. Google Scanned Objects: A high-quality dataset of 3D scanned household items. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2553–2560. IEEE, 2022. 6

[13] Andreas Engelhardt, Amit Raj, Mark Boss, Yunzhi Zhang, Abhishek Kar, Yuanzhen Li, Deqing Sun, Jonathan T. Barron, Hendrik P.A. Lensch, and Varun Jampani. SHINOBI: Shape and Illumination using Neural Object decomposition via Brdf optimization In-the-wild. In *CVPR*, 2024. 3

[14] Ruiqi Gao, Aleksander Holynski, Philipp Henzler, Arthur Brussee, Ricardo Martin-Brualla, Pratul P. Srinivasan, Jonathan T. Barron, and Ben Poole. CAT3D: Create anything in 3D with multi-view diffusion models. *arXiv*, 2024. 3

[15] Rohit Girdhar, Mannat Singh, Andrew Brown, Quentin Duval, Samaneh Azadi, Sai Saketh Rambhatla, Akbar Shah, Xi Yin, Devi Parikh, and Ishan Misra. EMU VIDEO: Factorizing Text-to-Video Generation by Explicit Image Conditioning, 2023. 3

[16] Yuan-Chen Guo, Ying-Tian Liu, Ruizhi Shao, Christian Laforte, Vikram Voleti, Guan Luo, Chia-Hao Chen, Zi-Xin Zou, Chen Wang, Yan-Pei Cao, and Song-Hai Zhang. threestudio: A unified framework for 3d content generation. https://github.com/threestudio-project/threestudio, 2023. 5

[17] Jon Hasselgren, Nikolai Hofmann, and Jacob Munkberg. Shape, Light & Material Decomposition from Images using Monte Carlo Rendering and Denoising. *NeurIPS*, 20222. 3, 5

[18] Zexin He and Tengfei Wang. OpenLRM: Open-source large reconstruction models. https://github.com/3DTopia/OpenLRM, 2023. 2, 3, 4, 6, 7, 12

[19] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020. 3

[20] Yicong Hong, Kai Zhang, Jiuxiang Gu, Sai Bi, Yang Zhou, Difan Liu, Feng Liu, Kalyan Sunkavalli, Trung Bui, and Hao Tan. LRM: Large reconstruction model for single image to 3D. *ICLR*, 2024. 2, 3, 4

[21] Zixuan Huang, Stefan Stojanov, Anh Thai, Varun Jampani, and James M. Rehg. ZeroShape: Regression-based zero-shot shape reconstruction. *arXiv*, 2023. 7

[22] Zixuan Huang, Justin Johnson, Shoubhik Debnath, James M Rehg, and Chao-Yuan Wu. Pointinfinity: Resolution-invariant point diffusion models. In *CVPR*, 2024. 4, 12

[23] Hanwen Jiang, Qixing Huang, and Georgios Pavlakos. Real3D: Scaling up large reconstruction models with real-world images. *arXiv*, 2024. 3

[24] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM TOG*, 42(4), 2023. 3

[25] Xin Kong, Shikun Liu, Xiaoyang Lyu, Marwan Taher, Xiaojuan Qi, and Andrew J Davison. EscherNet: A generative model for scalable view synthesis. *arXiv*, 2024. 3

[26] Jeong-gi Kwak, Erqun Dong, Yuhe Jin, Hanseok Ko, Shweta Mahajan, and Kwang Moo Yi. ViVid-1-to-3: Novel view synthesis with video diffusion models. *CVPR*, 2024. 3

[27] Bruno Levy. geogram. https://github.com/BrunoLevy/geogram, 2024. 2

[28] 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. *arXiv*, 2023. 3

[29] 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. *arXiv*, 2023. 3

[30] 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. *NeurIPS*, 2023. 3

[31] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3D object. *ICCV*, 2023. 2, 3

[32] 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*, 2023. 3

[33] Zexiang Liu, Yangguang Li, Youtian Lin, Xin Yu, Sida Peng, Yan-Pei Cao, Xiaojuan Qi, Xiaoshui Huang, Ding Liang, and Wanli Ouyang. Unidream: Unifying diffusion priors for relightable text-to-3D generation. *arXiv*, 2023. 3

[34] 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*, 2023. 3

[35] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM Transactions on Graphics (SIGGRAPH)*, 1987. 2

[36] Luke Melas-Kyriazi, Iro Laina, Christian Rupprecht, Natalia Neverova, Andrea Vedaldi, Oran Gafni, and Filippos Kokkinos. IM-3D: Iterative multiview diffusion and reconstruction for high-quality 3D generation. *arXiv*, 2024. 3

[37] Antoine Mercier, Ramin Nakhli, Mahesh Reddy, Rajeev Yasarla, Hong Cai, Fatih Porikli, and Guillaume Berger. HexaGen3D: Stablediffusion is just one step away from fast and diverse Text-to-3D generation. *arXiv*, 2024. 3

[38] Ben Mildenhall, Pratul Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. *ECCV*, 2020. 3

[39] Jacob Munkberg, Jon Hasselgren, Tianchang Shen, Jun Gao, Wenzheng Chen, Alex Evans, Thomas Mueller, and Sanja Fidler. Extracting Triangular 3D Models, Materials, and Lighting From Images. *CVPR*, 2022. 3

[40] Maxime Oquab, Timothée Darcet, Theo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Russell Howes, Po-Yao Huang, Hu Xu, Vasu Sharma, Shang-Wen Li, Wojciech Galuba, Mike Rabbat, Mido Assran, Nicolas Ballas, Gabriel Synnaeve, Ishan Misra, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2023. 4

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

[42] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 5

[43] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, pages 10684–10695, 2022. 3

[44] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. DreamBooth: Fine tuning text-to-image dissusion models for subject-driven generation. *arXiv*, 2022. 3

[45] Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial diffusion distillation. *arXiv*, 2023. 2

[46] 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 *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 2, 5

[47] Ruoxi Shi, Hansheng Chen, Zhuoyang Zhang, Minghua Liu, Chao Xu, Xinyue Wei, Linghao Chen, Chong Zeng, and Hao Su. Zero123++: a single image to consistent multi-view diffusion base model. *arXiv*, 2023. 3

[48] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P. Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. *CVPR*, 2016. 12

[49] Yichun Shi, Peng Wang, Jianglong Ye, Mai Long, Kejie Li, and Xiao Yang. MVDream: Multi-view diffusion for 3d generation. *arXiv*, 2023. 3

[50] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv*, 2020. 3

[51] StabilityAI. Stable Zero123, 2023. 2, 3

[52] Stanislaw Szymanowicz, C. Rupprecht, and Andrea Vedaldi. Splatter image: Ultra-fast single-view 3D reconstruction. *CVPR*, 2024. 3

[53] 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. *arXiv*, 2024. 3, 6, 7, 12

[54] Dmitry Tochilkin, David Pankratz, Zexiang Liu, Zixuan Huang, Adam Letts, Yangguang Li, Ding Liang, Christian Laforte, Varun Jampani, and Yan-Pei Cao. TripoSR: Fast 3D object reconstruction from a single image. *arXiv*, 2024. 2, 3, 6, 7, 12

[55] Shimon Vainer, Mark Boss, Mathias Parger, Konstantin Kutsy, Dante De Nigris, Ciara Rowles, Nicolas Perony, and Simon Donné. Collaborative control for geometry-conditioned PBR image generation. *arXiv*, 2024. 3

[56] Vikram Voleti, Alexia Jolicoeur-Martineau, and Christopher Pal. MCVD: Masked conditional video diffusion for prediction, generation, and interpolation. In *NeurIPS*, 2022. 3

[57] Vikram Voleti, Chun-Han Yao, Mark Boss, Adam Letts, David Pankratz, Dmitrii Tochilkin, Christian Laforte, Robin Rombach, and Varun Jampani. SV3D: Novel multi-view synthesis and 3D generation from a single image using latent video diffusion. *arXiv*, 2024. 2, 3, 5

[58] Peng Wang, Hao Tan, Sai Bi, Yinghao Xu, Fujun Luan, Kalyan Sunkavalli, Wenping Wang, Zexiang Xu, and Kai Zhang. PF-LRM: Pose-free large reconstruction model for joint pose and shape prediction. *arXiv*, 2023. 3

[59] Zhengyi Wang, Yikai Wang, Yifei Chen, Chendong Xiang, Shuo Chen, Dajiang Yu, Chongxuan Li, Hang Su, and Jun Zhu. CRM: Single image to 3D textured mesh with convolutional reconstruction model. *arXiv*, 2024. 3, 6, 7, 12

[60] Xinyue Wei, Kai Zhang, Sai Bi, Hao Tan, Fujun Luan, Valentin Deschaintre, Kalyan Sunkavalli, Hao Su, and Zexiang Xu. MeshLRM: Large reconstruction model for high-quality mesh. *arXiv*, 2024. 3, 5

[61] Hao Wen, Zehuan Huang, Yaohui Wang, Xinyuan Chen, Yu Qiao, and Lu Sheng. Ouroboros3D: Image-to-3D generation via 3D-aware recursive diffusion. *arXiv*, 2024. 3

[62] Haohan Weng, Tianyu Yang, Jianan Wang, Yu Li, Tong Zhang, C. L. Philip Chen, and Lei Zhang. Consistent123: Improve consistency for one image to 3D object synthesis. *arXiv*, 2023. 3

[63] Kailu Wu, Fangfu Liu, Zhihan Cai, Runjie Yan, Hanyang Wang, Yating Hu, Yueqi Duan, and Kaisheng Ma. Unique3D: High-quality and efficient 3D mesh generation from a single image. *arXiv*, 2024. 3

[64] Tong Wu, Jiarui Zhang, Xiao Fu, Yuxin Wang, Liang Pan Jiawei Ren, Wayne Wu, Lei Yang, Jiaqi Wang, Chen Qian, Dahua Lin, and Ziwei Liu. Omniobject3d: Large-vocabulary 3d object dataset for realistic perception, reconstruction and generation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 6

[65] Desai Xie, Sai Bi, Zhixin Shu, Kai Zhang, Zexiang Xu, Yi Zhou, Soren Pirk, Arie E. Kaufman, Xin Sun, and Hao Tan. LRM-Zero: Training large reconstruction models with synthesized data. *arXiv*, 2024. 3

[66] Rengan Xie, Wenting Zheng, Kai Huang, Yizheng Chen, Qi Wang, Qi Ye, Wei Chen, and Yuchi Huo. LDM: Large tensorial SDF model for textured mesh generation. *arXiv*, 2024. 3

[67] Yiming Xie, Chun-Han Yao, Vikram Voleti, Huaizu Jiang, and Varun Jampani. Sv4d: Dynamic 3d content generation with multi-frame and multi-view consistency. *arXiv preprint arXiv:2407.17470*, 2024. 2

[68] 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*, 2024. 3, 6, 7, 12

[69] Yinghao Xu, Hao Tan, Fujun Luan, Sai Bi, Peng Wang, Jiahao Li, Zifan Shi, Kalyan Sunkavalli, Gordon Wetzstein, Zexiang Xu, and Kai Zhang. DMV3D: Denoising multi-view diffusion using 3D large reconstruction model. *arXiv*, 2023. 3

[70] 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. *arXiv*, 2024. 3

[71] Jianglong Ye, Peng Wang, Kejie Li, Yichun Shi, and Heng Wang. Consistent-1-to-3: Consistent image to 3D view synthesis via geometry-aware diffusion models. In *3DV*, 2024. 3

[72] Jonathan Young. xatlas. https://github.com/jpcy/xatlas, 2024. 2

[73] Meng yue Li, Xiaoxiao Long, Yixun Liang, Weiyu Li, Yuan Liu, Peng Li, Xiaowei Chi, Xingqun Qi, Wei Xue, Wenhan Luo, Qi fei Liu, and Yike Guo. M-LRM: Multi-view large reconstruction model. *arXiv*, 2024. 3

[74] Greg Zaal. Poly haven, 2024. https://polyhaven.com/. 8

[75] Kai Zhang, Fujun Luan, Qianqian Wang, Kavita Bala, and Noah Snavely. PhySG: Inverse rendering with spherical Gaussians for physics-based material editing and relighting. *CVPR*, 2021. 3

[76] Kai Zhang, Sai Bi, Hao Tan, Yuanbo Xiangli, Nanxuan Zhao, Kalyan Sunkavalli, and Zexiang Xu. GS-LRM: Large reconstruction model for 3D gaussian splatting. *arXiv*, 2024. 3

[77] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. *CVPR*, 2018. 6

[78] Yuqing Zhang, Yuan Liu, Zhiyu Xie, Lei Yang, Zhongyuan Liu, Mengzhou Yang, Runze Zhang, Qilong Kou, Cheng Lin, Wenping Wang, and Xiaogang Jin. DreamMat: High-quality PBR material generation with geometry- and light-aware diffusion models. *arXiv*, 2024. 3

[79] Ruowen Zhao, Zhengyi Wang, Yikai Wang, Zihan Zhou, and Jun Zhu. FlexiDreamer: Single image-to-3D generation with flexicubes. *arXiv*, 2024. 3

[80] Chuanxia Zheng and Andrea Vedaldi. Free3D: Consistent novel view synthesis without 3D representation. *arXiv*, 2023. 3

[81] Peiye Zhuang, Songfang Han, Chaoyang Wang, Aliaksandr Siarohin, Jiaxu Zou, Michael Vasilkovsky, Vladislav Shakhrai, Sergey Korolev, S. Tulyakov, and Hsin-Ying Lee. GTR: Improving large 3D reconstruction models through geometry and texture refinement. *arXiv*, 2024. 3

[82] Zi-Xin Zou, Zhipeng Yu, Yuanchen Guo, Yangguang Li, Ding Liang, Yan-Pei Cao, and Song-Hai Zhang. Triplane meets gaussian splatting: Fast and generalizable single-view 3D reconstruction with transformers. *arXiv*, 2023. 3

Supplementary Material

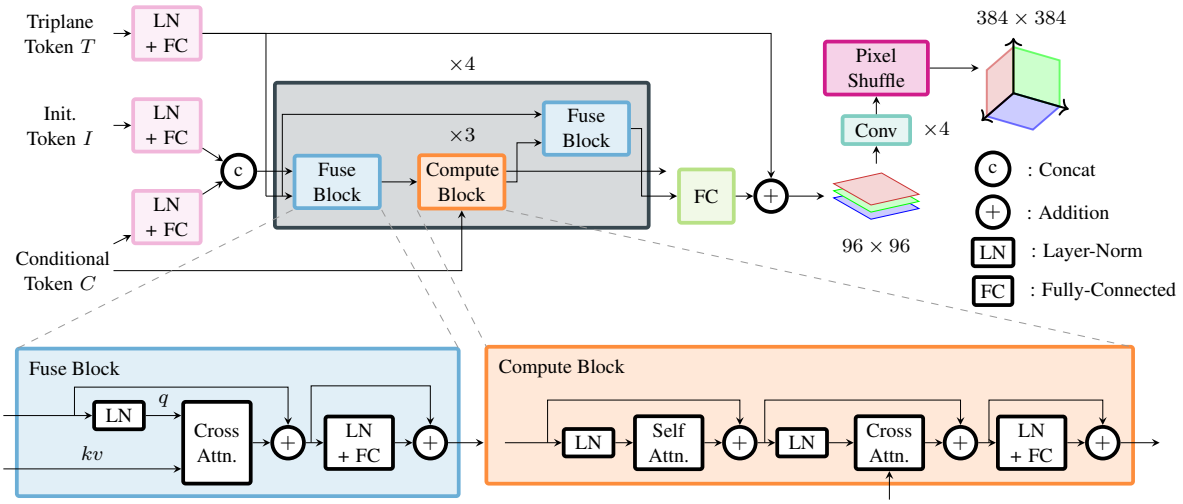

Figure A1. **Enhanced Transformer Architecture**Our new backbone produces higher resolution output triplanes. We further upscale them using a pixel shuffling [48]. This helps capture high-frequency, high-contrast textures with reduced aliasing as in Fig. 4.

A1. Enhanced Transformer

To reduce the aliasing artifact, we upgrade the transformer backbone to produce triplanes at a resolution of 384 × 384. However, naively increasing the triplane tokens in TripoSR [54] is computationally prohibitive due to the quadratic complexity of self-attention. Inspired by PointInfinity [22], we leverage a two-stream transformer, which has linear complexity w.r.t. the number of tokens. As illustrated in Fig. A1, our architecture consists of two processing streams, the triplane stream and the latent stream. The triplane stream consists of the raw triplane tokens to be processed. In each two-stream unit (gray box in Fig. A1), the latent stream fetches information from the triplane stream using cross attention, and performs the main computation upon a set of constant-sized latent tokens. The latent stream then updates the triplane stream with the processed latent tokens. Our full architecture consists of four such two-stream units. With this computationally detached design, our transformer is able to produce triplanes at a resolution of 96 × 96 with 1024 channels. To further increase the resolution and reduce aliasing, we integrated a pixel shuffling operation [48], enhancing the triplane resolution to 384 × 384 with a feature dimension of 40.

A2. Image metrics

For the image metrics, we follow the pipeline of the shape metric calculation. We further scale the normalized objects back to the actual GT scale and run another finer ICP optimization to adjust the fine scale. This transform is then used to render meshes. This can still result in texture misalignment for highly symmetrical objects, so we treat the image metrics as an auxiliary metric to evaluate the final quality of the asset reconstructions. Therefore, we only report this metric in the supplements. Table A1 also then supports the improved visual quality during rendering seen in the main paper.

	GSO			OmniObject		
Method	PSNR↑	SSIM ↑	LPIPS↓	PSNR↑	SSIM ↑	LPIPS↓
OpenLRM [18]	15.689	0.787	0.206	13.975	0.760	0.229
TripoSR [54]	16.445	0.789	0.194	14.331	0.755	0.224
LGM [53]	14.377	0.762	0.248	12.662	0.732	0.273
CRM [59]	15.054	0.778	0.228	13.462	0.755	0.245
InstantMesh [68]	15.434	0.785	0.203	13.531	0.757	0.235
Ours	**21.247**	**0.865**	**0.124**	**20.134**	**0.851**	**0.132**

Table A1. **Comparison on Image Metrics.**