DG3D: Generating High Quality 3D Textured Shapes by Learning to Discriminate Multi-Modal Diffusion-Renderings

Qi Zuo Yafei Song Jianfang Li Lin Liu Liefeng Bo

Alibaba Group

{muyuan.zq,huaizhang.syf,wuhui.ljf,lorrain.ll,liefeng.bo}@alibaba-inc.com

Abstract

*Many virtual reality applications require massive 3D content, which impels the need for low-cost and efficient modeling tools in terms of quality and quantity. In this paper, we present a **D**iffusion-augmented **G**enerative model to generate high-fidelity **3D** textured meshes that can be directly used in modern graphics engines. Challenges in directly generating textured mesh arise from the instability and texture incompleteness of a hybrid framework which contains conversion between 2D features and 3D space. To alleviate these difficulties, DG3D incorporates a diffusion-based augmentation module into the min-max game between the 3D tetrahedral mesh generator and 2D renderings discriminators, which stabilizes network optimization and prevents mode collapse in vanilla GANs. We also suggest using multi-modal renderings in discrimination to further increase the aesthetics and completeness of generated textures. Extensive experiments on the public benchmark and real scans show that our proposed DG3D outperforms existing state-of-the-art methods by a large margin,* i.e.*, $5\% \sim 40\%$ in FID-3D score and $5\% \sim 10\%$ in geometry-related metrics. Code is available at https://github.com/seakforzq/DG3D.*

1. Introduction

The production of 3D content plays a crucial role in the film, gaming, simulation, and social industries. As the demand continues to grow, manually made 3D models could not keep up with the expansion of the industries. Researchers have made substantial progress in generating geometry for 3D models [9, 10, 22, 27, 35, 2, 28], but the corresponding textures received less attention. While some methods have considered texture generation [3, 25, 41, 36], most of them infer textures based on fixed geometric inputs, which means that they are not fully generative models and do not take into account the interdependent relationship between geometry and texture.

3D-aware image generation [4, 24, 34, 3], a related research field with textured mesh generation, aims to synthesize images from all viewpoints. These methods ensure consistency between different views by incorporating the priority constraint of volume rendering. They can produce high-resolution images with a super-resolution network. However, though the shapes can always be extracted from the density field by setting specific levels-set values, the quality of the extracted shapes is often much lower than the quality of the synthetic novel views. Since conversion from implicit representations to explicit meshes significantly degrades fidelity, methods that directly optimize explicit meshes are urgently needed.

Recently, differentiable mesh reconstruction [11, 33, 12] has shed light on the generation of textured mesh. DefTet [11] utilizes a neural network to deform an initial tetrahedral mesh's vertices and predict the occupancy for each tetrahedron. DMTet [33] instead reconstructs the shapes by predicting the SDF defined on a deformable tetrahedral grid. Then it converts the SDF to a surface mesh by a novel differential Marching Tetrahedral layer. GET3D [12] further incorporates DMTet [33] in StyleGAN2 [18] and uses a hybrid tri-plane representation for transformation from 2D feature space to 3D tetrahedral space. It extracts textured meshes by the differentiable Marching Tetrahedron layer and adopts DiffRast [20] to render the textured meshes as RGB images and silhouettes for discrimination. Though GET3D [12] achieves an impressive performance, a potential deficiency that causes instability in training with rendered images is the inadequate 3D supervision that discriminators can receive. As a result, we often observe mode collapse in such a paradigm that combines a 3D generator and multiple 2D discriminators [12]. Moreover, we find that direct discrimination on the entire image results in texture incompleteness, significantly degrading the quality of shapes.

Figure 1: We export the generated shapes in $.fbx$ format and render them with a transparent background. We add sunlight in the scene so the shadow effect can be observed. DG3D can generate a fine texture with arbitrary topology.

Luckily, the latest achievements in 2D image generation and texture generation have motivated us. To facilitate high-resolution image data generation, Wang *et al*. [38] propose a method that combines the diffusion model with GANs and successfully generates more realistic images with higher stability. Furthermore, for the quality of shapes, Texturify [36] suggests taking a union of normals, curvature, and patches as the input of discriminators.

Based on the above, we propose DG3D, a novel 3D GAN-based paradigm augmented by an adaptive diffusion-augmented module, to properly deal with mode collapse and enhance the robustness of both generation and discrimination. To further improve the texture completeness of the generated shapes, we set up different discriminators for multi-modal renderings. In the forward stage, DG3D first generates 2D features and aligns the features to the 3D tetrahedral space by a hybrid tri-plane representation. Then DG3D utilizes the differentiable Marching Tetrahedron layer [33] to extract the underlying geometry. Meanwhile, the corresponding texture is generated by another tri-plane and aligned to the geometry by 3D coordinates. The textured mesh is further rendered as RGB, alpha transparency, and patch images. The multi-modal renderings will be fed through the adaptive diffusion-augmented module and corresponding discriminators sequentially. DG3D is end-to-end trainable and outperforms the baseline [12] by a large margin. As depicted in Fig. 1, DG3D faithfully generates feasible shapes with high-fidelity textures. Note, despite that DG3D introduces extra parameters of discriminators and operations on renderings, it maintains the same inference speed as [12] as they share a common generator.

Our main contributions include: 1) We propose a hybrid generative model which combines a generative adversarial network with a novel adaptive diffusion-augmented module, achieving better performance in 3D textured mesh generation. 2) We design multiple 2D discriminators for multi-modal diffusion-renderings from the textured mesh, which provide more supervision information and significantly boost the generated quality. 3) We conduct extensive experiments and ablations to demonstrate the superiority of

DG3D, analyze the limitations, and discuss applications and feasible improvements in future work.

2. Related Work

3D-Aware Image Generation. The success of neural volume rendering [23] have motivated recent work [4, 24, 34, 3] in 3D-aware image generation. A base consumption of these methods is that RGB and the alpha transparency of volume depend on the angle of view, which makes it 3D-aware and also a non-trivial endeavor to extract explicit surfaces. Despite falling short in producing feasible explicit surfaces, the paradigm combining a generative model and an efficient 3D representation has provided afflatus to recent advanced 3D generative models.

3D Generative Models. Recent 3D approaches have primarily focused on geometry generation using various representations, such as mesh [11, 33], point cloud [39], and occupancy [22, 28]. PointFlow [39] leverages normalizing flow for sampling in the latent space of PointNet [30], leading to superior point cloud generation performance. ConvONet [28] provides detailed experimental results on implicit occupancy prediction with different encoded dimensions. It is further developed into the tri-plane geometry hybrid representation, which is first proposed by EG3D [3] for 3D-aware image generation.

The first method which is capable of generating explicit free-topology textured meshes is GET3D [12]. It leverages a GAN-based, differentiable paradigm and a tri-plane hybrid representation [3] to bridge the gap between 3D generation and 2D discrimination. GET3D generates feature maps like vanilla GANs and reformulates feature maps as a tri-plane, which is further sampled and mapped as signed distance function values and offsets of a predefined standard tetrahedron template [33]. Then a differentiable marching tetrahedron layer is adopted for extracting raw meshes, and texture will be aligned to meshes by interpolation in 3D space. Finally, DiffRast [20] is adopted for rendering RGB and mask images from the generated shapes for discrimination. Even though GET3D adopts an imaginative design and an efficient hybrid representation, it still needs improvements in certain aspects. As highlighted by DecorGAN [8], traditional GAN with CNN architectures tends to perform inadequately on binary images due to the discriminator's inability to distinguish between authentic and inauthentic pixels. Furthermore, we experimentally find that global-style discrimination often results in texture incompleteness. In contrast, DG3D is optimized stably and produces complete texture most time.

3D Generation By Lifting 2D. Due to the lack of 3D assets, researchers [29, 21, 7] also seek to utilize 2D image generation models and differentiable rendering to optimize a 3D representation according to certain text prompts. Despite failure under some prompts and long optimization time(about one hour), they can generate high-fidelity textured shapes across classes. We acknowledge that these methods are nonnegligible and take equal importance with 3D Generative Models in the development of 3D Generation.

Hybrid Generative Models. The design of a single generative model often has certain limitations. VAEs, for instance, struggle to balance the latent priority and reconstruction accuracy. Meanwhile, GANs are difficult to optimize and prone to mode collapse. Normalizing flows and diffusion models are much slower during the inference stage, despite the exploration of faster sampling methods. Consequently, hybrid generative models have been increasingly used in real-world applications. For example, Stable Diffusion [31] relieves the limitations of VAE architectures by incorporating a diffusion model to broaden the acceptable distribution field of the latent priority, which enables VAEs to achieve both high reconstruction precision and a Gaussian latent space assumption. DS-Fusion [37] combines pre-trained stylized diffusion models with discriminators on real letters and shows visually favorable results of stylized font generation. LION [42] incorporates a point diffusion module and a latent diffusion module to augment a point cloud VAE, achieving state-of-the-art performance in point cloud generation and voxel-based completion. Whether in 2D or 3D research domains, hybrid generative models have shown more excellent capabilities in diverse tasks, which also motivates our design.

3. Method

The framework is illustrated in Fig. 2. It comprises three parts: Textured Mesh Generator, Diffusion-Augmented Module, and Multi-Modal Discriminators. The textured mesh generator will be first discussed in Sec. 3.1. Further, we will introduce the specific design of our diffusion-augmented module in Sec. 3.2 and the multi-modal discriminators in Sec. 3.3.

3.1. Textured Mesh Generator

Following the practice in Gao *et al*. [12], we adopt an improved generator as the baseline. We will briefly describe the improved generator and refer the reader to the original paper for further details.

Network Architecture. The network mainly preserves the design of the generator in StyleGAN2 [18], mapping the geometry latent code $z_G \in \mathbb{R}^{512}$, the texture latent code $z_T \in \mathbb{R}^{512}$ to the feature maps which is then reformulated as the tri-plane [3, 28] of both texture and geometry. The geometry part of the tri-plane is converted to the signed distance function value s_i and deformation d_i at each tetrahedron vertex v_i by grid sample. A differentiable Marching Tetrahedron layer [33] is used to extract shapes from the SDF values and offsets of vertices. The texture part of the

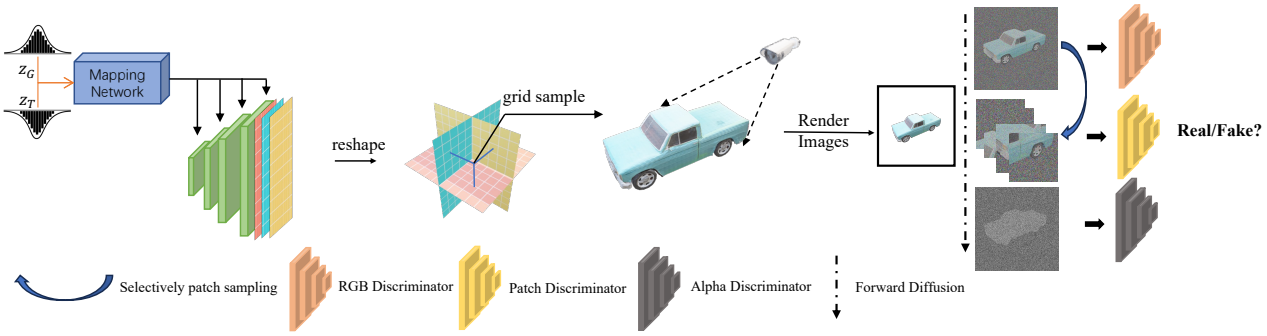

Figure 2: The framework comprises three parts: Textured Mesh Generator, Diffusion-Augmented Module, and Multi-modal Discriminators. The textured mesh generator follows the basic design of the improved G in [12] and it can produce textured meshes given specific geometry and texture latent codes. Then the Diffusion-Augmented Module takes the rendered images from textured meshes as input and randomly samples the images by t forward diffusion steps. Multi-Modal Discriminators take the diffused RGB, alpha transparency, and patch images as input and take the time t, the angle of rendering as conditions, and finally output the probability that a sample is treated as real. The whole framework can be optimized end-to-end with a vanilla objective proposed in 2D GANs. We refer the reader to Appendix for further structural details of the textured mesh generator and the multi-modal discriminators.

tri-plane is further converted to the color $c_i \in \mathbb{R}^3$ of each vertex v_i to align with the extracted shapes by interpolation.

The improved generator can produce textured mesh with arbitrary topology. However, in a min-max game, it is harmful if one side is much stronger than the other. A typical phenomenon we observe is that the discriminators often collapse and regard all samples as real. The reason behind it, as we find, is the unequal ability acquired respectively by the generator and the discriminator. We visualize typical mode collapse cases in Fig. 3. To this end, we design a diffusion-augmented module and use multi-modal renderings from the textured mesh for discrimination.

3.2. Diffusion-Augmented Module

We propose to use diffusion sampling to inject gaussian noise into the renderings from the textured mesh and adaptively adjust the magnitude of deviation based on the discriminator's output. The output rendering image $x_g \in \mathbb{R}^{H\times W\times 4}$ will be sampled by $t \in [0, T]$ steps using a forward diffusion process. T is adaptively adjusted as:

$$T = T + \text{sign}(r_d - d_{target}) * C, \tag{1}$$

where r_d is the mean output probability of the discriminators for real samples, d_{target} is the probability we expect the discriminators to output for real samples, C is a constant that controls the speed of adjusting. The equation encourages larger T when the discriminators could output probability higher than the target for real samples and vice versa. The motivation behind this is to gradually increase the difficulty for discrimination since larger T represents possible larger noise injection. Here we follow the same symbol as Karras *et al*. [16] since we modify r_d from their version for

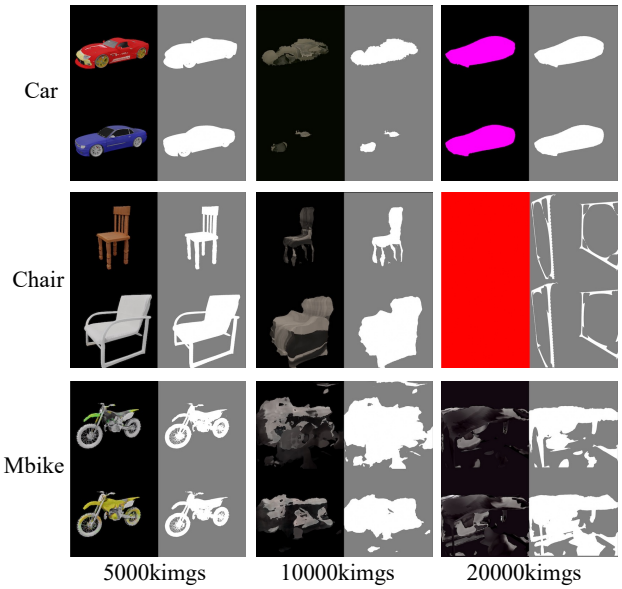

Figure 3: We visualize the generated RGB and mask images of Car, $Chair$, and $Motorbike$ of ShapeNet [5] after training for 5000, 10000, and 20000 thousand images respectively. As depicted, the mode collapse phenomenon often occurs in the training process of [12](also in NeRF-based GAN framework [3] as we observed), which we attribute to the unequal ability acquired by the generator and the discriminator.

multi-modal discrimination:

$$r_d = \frac{1}{3}\sum_{\phi\in\phi_c,\phi_\alpha,\phi_p} \mathbb{E}_{y,t\sim p(y,t)}[\text{sign}(D_\phi(y,\theta,t) - 0.5)], \tag{2}$$

where $D_{\phi_c}, D_{\phi_\alpha}, D_{\phi_p}$ denote RGB, alpha transparency, patches images discriminator respectively, y denotes real samples. θ denotes the angle of rendering, and t denotes the steps of sampling. We adopt a priority distribution p_π

from Wang *et al*. [38] to sample t from $[0, T]$:

$$t \sim p_\pi = \text{Discrete}(\frac{1}{\sum_{t=1}^{T} t}, \frac{2}{\sum_{t=1}^{T} t}, \cdots, \frac{T}{\sum_{t=1}^{T} t}), \quad (3)$$

where $\frac{n}{\sum_{t=1}^{T} t}$ denotes the probability of $t = n$ sampling steps. The priority favors larger t to encourage the discriminators to see more new samples from the new steps when T increases, which makes the discriminators focus on harder samples that it has never met. For each forward iteration, rendering images $x_g \sim p_g(x)$ and the corresponding real images $x \sim p(x)$ will be sampled by t steps [15] as:

$$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1 - \sqrt{\bar{\alpha}_t})\sigma^2\boldsymbol{I}), \quad (4)$$

where $\alpha_t := 1 - \beta_t$ with β_t the deviation of noise of t sampling steps, $\bar{\alpha}_t := \prod_{s=1}^{t} \alpha_s$, q denotes the forward sampling process, and $\mathcal{N}$ denotes the standard Gaussian distribution. We use a linear scheduler $\beta_t = \beta_0 + \frac{t}{T}(\beta_T - \beta_0)$, where $\beta_0 = 1e-4$ and $\beta_T = 1e-3$ for all experiments. Since we apply the same sampling to the generated image x_g and real image x in one loop, we unify the symbol in Eq. (4). In each iteration, the deviation is dynamically changed with T according to Eq. (1), so we modify all discriminators to be time-dependent to accept t as a condition, and following Gao *et al*. [12], we also add the rendering camera pose θ as another condition.

Trained with binary images, the discriminator will treat pixels with any value other than 0 or 1 as fake, thus preventing continuous optimization. A simple approach is to apply a gaussian filter with a fixed deviation to the binary image [8]. However, in our experiments, it also causes mode collapse to fix the deviation since the discriminator can not handle complex samples at the beginning of the training stage. Based on this, we design a novel diffusion-augmented module that can adjust the deviation by assigning different maximum timesteps T, thus gradually strengthening the ability of discriminators. Compared with the baseline [12], we have not observed mode collapse anymore, and the model can stably converge to a better minimum. For a theoretical proof of the equivalence between optimization on the original generated image and the diffusion-augmented images, we refer the reader to the supplementary material of [38].

3.3. Multi-Modal Discriminators

To further improve the texture generation performance, we augment the supervision information by feeding multi-modal renderings to the discriminators. We maintain the branch of RGB images, make the masks continuous as alpha transparency images by diffusion sampling, and further, we clip patches from the RGB images with a criterion to reject patches with sparse texture pixels. In detail, for a generated image x_g with corresponding alpha transparency image x_α, we reject a patch x_p of it if:

$$\sum_{i=0}^{H_p}\sum_{j=0}^{W_p} x_\alpha(i + r_p, j + c_p) < \gamma H_p W_p, \quad (5)$$

where r_p and c_p represent the coordinate of the left-top corner of x_p in x_g, $x_\alpha(i + r_p, j + c_p) \in [0, 1]$ represents the transparency of the pixel with coordinate (i, j) in x_p, H_p and W_p is the height and the width of x_p, and γ is a weight factor which decides the minimum summation of transparency of x_p. We set $\gamma = 0.15, H_p = 256, W_p = 256$ for all experiments. Next, we design a patch discriminator which accepts patches from a single generated image as input. The Zoom-In rendering method works as a general technique to augment the texture quality as also demonstrated in DreamHuman [19].

We adopt the same non-saturating GAN objective with R1 regularization from StyleGAN [17]. For an RGB image or an alpha transparency image, the optimization objective is defined as follows:

$$\begin{aligned} &L(D_x, G, t) = \\ &\mathbb{E}_{z_G, z_T \in \mathcal{N}, \theta \in [0,2\pi]}[f(D_x(d[R(G(z_G, z_T), \theta), t], \theta, t))] \\ &+\mathbb{E}_{I_x \in p_x, \beta \in [0,2\pi]}[f(-D_x(d[I_x, t], \beta, t)))] \\ &+\xi||\nabla D_x(d[I_x, t], \beta, t)||_2^2], \end{aligned} \quad (6)$$

where $d[]$ is the diffusion function defined in Equ. Eq. (4), $f(u)$ is defined as $f(u) = -\log(1 + \exp(-u))$, p_x is the distribution of real images, R denotes rendering process, ξ is a hyperparameter, and β, θ are camera angles for rendering. As for patches of RGB images, the objective is defined as:

$$\begin{aligned} &L(D_p, G, t) = \\ &\frac{1}{N}\sum_{i=1}^{N}(\mathbb{E}_{z_G, z_T \in \mathcal{N}, \theta \in [0,2\pi]}[f(D_p(p^i_{fake}, \theta, t))] \\ &+\mathbb{E}_{I_x \in p_x, \beta \in [0,2\pi]}[f(-D_p(P(d[I_x, t], i), \beta, t)))] \\ &+\xi||\nabla D_p(P(d[I_x, t], i), \beta, t)||_2^2]), \end{aligned} \quad (7)$$

where $p^i_{fake} = P(d[R(G(z_G, z_T), \theta), t], i)$. P is the process that generates patches according to Eq. (5) and N is set as 4 in all experiments. The overall objective is defined as:

$$L = L(D_c, G, t) + L(D_\alpha, G, t) + \lambda L(D_p, G, t) + \mu L_{reg}, \quad (8)$$

where L_{reg} is a cross-entropy loss defined between the signed distance function values of neighboring vertices, which is used to remove invisible internal faces [33], μ is the weighted factor of regularization, and λ is the weighted factor of adversarial objective defined on patches.

Table 1: Quantitative metrics: IG denotes the improved G setting of GET3D [12]. Our method outperforms related works by a large margin on Car, $Chair$, and $Motorbike$ classes of ShapeNet [5] in geometry diversity (COV), geometry similarity (MMD), and texture quality (FID-3D).

Category	Method	COV↑(%)		MMD↓($\times 10^{-3}$)		FID-3D↓
		LFD	CD	LFD	CD	
Car	PointFlow [39]	51.91	57.16	1971	0.82	-
	OccNet [22]	27.29	42.63	1717	**0.61**	-
	Pi-GAN [4]	0.82	0.55	6626	25.54	104.29
	GRAF [32]	1.57	1.57	6012	10.63	52.85
	EG3D [3]	60.16	49.52	1527	0.72	21.89
	GET3D [12]	66.78	58.39	1491	0.71	10.25
	GET3D [12]+subdiv	62.48	55.93	1553	0.72	12.14
	GET3D [12](IG)	59.00	47.95	1473	0.81	10.60
	Ours(DG3D)	**74.63**	**59.30**	**1363**	0.67	**6.09**
Chair	PointFlow [39]	49.58	71.87	3755	**3.03**	-
	OccNet [22]	61.10	67.13	3494	3.98	-
	Pi-GAN [4]	53.76	39.65	4092	6.65	120.53
	GRAF [32]	50.23	39.28	4055	6.80	61.63
	EG3D [3]	58.31	50.14	3444	4.72	46.06
	GET3D [12]	69.08	69.91	3167	3.72	23.28
	GET3D [12]+subdiv	71.59	70.84	3163	3.95	23.17
	GET3D [12](IG)	71.96	71.96	3125	3.96	22.41
	Ours(DG3D)	**74.80**	**73.65**	**3024**	3.58	**18.84**
Mbike	PointFlow [39]	50.68	63.01	4023	**1.38**	-
	OccNet [22]	30.14	47.95	4551	2.04	-
	Pi-GAN [4]	2.74	6.85	8864	21.08	131.38
	GRAF [32]	43.84	50.68	4528	2.40	113.39
	EG3D [3]	38.36	34.25	4199	2.21	89.97
	GET3D [12]	67.12	67.12	3631	1.72	65.60
	GET3D [12]+subdiv	63.01	61.64	3440	1.79	54.12
	GET3D [12](IG)	69.86	65.75	3393	1.79	48.90
	Ours(DG3D)	**75.34**	**71.23**	**3032**	1.68	**46.84**

4. Experiments

We conduct experiments on synthetic datasets and real-world scans in Sec. 4.1 to evaluate our method. We then ablate the design of DG3D in Sec. 4.2. More experimental results are provided in Appendix.

4.1. Results on Synthetic and Real Datasets

Datasets. We use ShapeNet [5], Twindom[1] and ScanHead[2] for evaluation. We use three classes, Car, $Chair$, and $Motorbike$ from ShapeNet [5], and follow the same splits and rendering views for each category as [12]. We collect 991 real human scans from Twindom, which have various poses and detailed dressings. We use 700 scans for training, 100 for validation, and 191 for testing. We also collect 265 scans from a head scanner which provides a coarse reconstruction for restoration. We use 24 random views for Twindom and ScanHead datasets.

Baselines. We compare our model to three groups of methods: **1)** 3D generative models that only generate geometry without texture: PointFlow [39] and OccNet [22]. **2)** 3D-aware image generation methods: GRAF [32], PiGAN [4], and EG3D [3]. **3)** Methods that simultaneously output geometry with texture: GET3D [12]. All methods will be included for ShapeNet [5] classes, and only GET3D [12] will be included as a comparison for Twindom and ScanHead datasets.

Metrics. We adopt the FID-3D [12] metric, which is more aware of explicit texture quality. Specifically, we render the generated textured meshes into 2D images and compute the FID [14] metric between 50k images. As for geometry, we adopt the Chamfer Distance [1] and Light Field Distance [6] to compute the Coverage score and Minimum Matching Distance. For methods [32, 4, 3] that do not directly output meshes, we use Marching Cubes to extract the underlying geometry of the radiance field.

Experimental results. Qualitative results on ShapeNet [5] are provided in Tab. 1. We achieve better performance on both geometry and texture when compared to methods of 3D-aware image synthesis and simultaneously textured mesh generation and achieve comparable performance in generating geometry when compared with methods using direct 3D supervision. Compared with methods(Pi-GAN [4], GRAF [32], and EG3D [3]) that focus on the rendering quality and GET3D series [12] which focus on the quality of the underlying explicit mesh, DG3D achieve significantly better performance in geometry diversity (COV), geometry quality (MMD) and rendering quality (FID-3D) on all datasets. Compared with PointFlow [39], which samples 2048 points from a mesh and directly optimizes the positions of the sampled points, we achieve better performance on Car dataset and achieve a suboptimal performance on the $Chair$ and $Motorbike$ datasets for the reason that DG3D uses random views for training which makes it hard to capture the information of hidden internal structures in a mesh. For the same reason, DG3D achieves a suboptimal performance compared with OccNet [22] on the Car dataset while achieving better performance in the $Chair$ and $Motorbike$ datasets.

For qualitative comparison, we visualize the output of all methods in Fig. 4. For PointFlow [39], we use the official mitsuba renderer[3] to visualize the point cloud. For OccNet [22], GRAF [32], PiGAN [4], and EG3D [3], we use Marching Cubes and adjust the level-set values to get a proper extraction. For GET3D [12], we directly visualize the textured mesh generated by pre-trained weights. PointFlow [39] can not produce reasonable geometries from sparse point clouds, and OccNet [22] is comparable with DG3D in geometry but can not produce any texture. 3D-aware image generation methods fall short in explicit topology quality, although they can generate high-quality synthetic views. GET3D [12] generates geometry and texture simultaneously but suffers from texture incompleteness and training instability. DG3D generates textures nearly free of impurities and achieves higher visual quality, which sets it

[1] https://web.twindom.com/

[2] We collect the scans from a scanner of 28 views and manually fix the topology holes.

[3] https://github.com/zekunhao1995/PointFlowRenderer

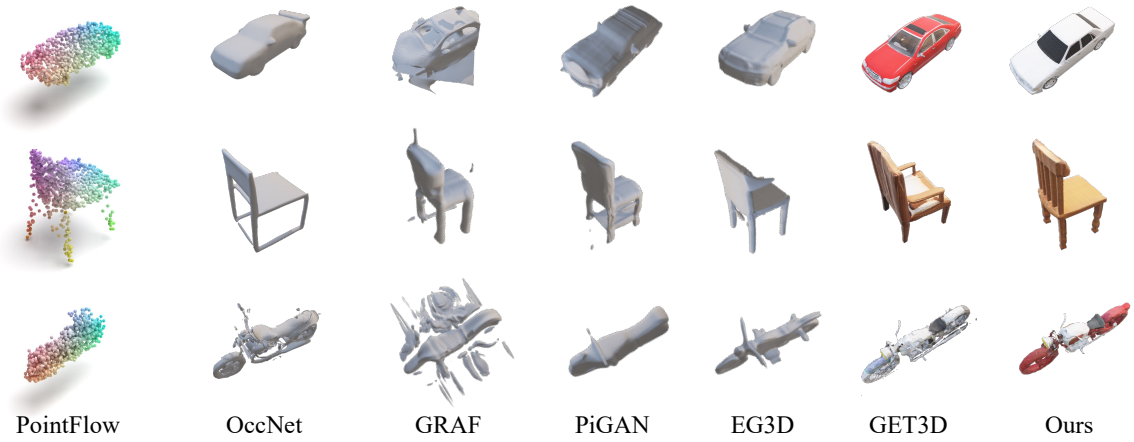

Figure 4: Qualitative comparison on ShapeNet [5]. DG3D can generate shapes with higher quality in both texture and geometry among all categories.

Figure 5: Qualitative comparison on Twindom and ScanHead. DG3D also performs better on real-world scans in geometry(more smooth) and texture(no black artifacts).

Table 2: Quantitative Metrics on Twindom and ScanHead. On real scans, DG3D also faithfully outperforms GET3D [12].

Category	Method	COV↑(%)		MMD↓($\times 10^{-3}$)		FID-3D↓
		LFD	CD	LFD	CD	
Twindom	GET3D [12](IG)	70.81	67.19	3272	2.34	23.11
	Ours(DG3D)	**73.39**	**69.75**	**3134**	**2.07**	**21.37**
ScanHead	GET3D [12](IG)	69.20	64.31	2286	1.59	43.29
	Ours(DG3D)	**75.11**	**72.48**	**2037**	**1.32**	**35.18**

apart from related methods.

For real datasets (Twindom and ScanHead), we show results of qualitative and quantitative comparisons with GET3D [12]. As seen from Fig. 5, DG3D generates more visually favorable textures with fewer artifacts. Quantitative results shown in Tab. 2 again demonstrate the superiority of DG3D on real scans. Note that the poses and appearance of body scans in Twindom are more diverse than that of Renderpeople [12], so both GET3D and DG3D do not generate clear facial texture. We demonstrate the ability to generate detailed facial textures by training DG3D on ScanHead. We zoom in on the textured mesh generated by GET3D and DG3D for a better view in Fig. 6. Compared with GET3D, DG3D consistently produces 3D shapes with higher quality.

To demonstrate the editing ability explicitly, we interpolate between random latent codes and visualize them in Fig. 7. DG3D can generate meaningful shapes faithfully in the interpolated space.

4.2. Ablation

We conduct ablation experiments in two ways: **1)** w/ or w/o the diffusion-augmented module, **2)** w/ or w/o multi-modal discriminators. As shown in Tab. 3, MD(Multi-Modal Discriminators) significantly improves the FID-3D metric, which represents the quality of texture, and slightly improves COV and MMD metrics represent the quality of geometry. DAM(Diffusion-Augmented Module) can improve the geometry and texture quality to the same extent. Both MD and DAM conform to the intention of our design, *i.e.*, MD for better texture quality and DAM for better comprehensive quality. The combination of MD and DAM further shows performance improvement in both geometry-related and texture-related metrics.

5. Applications

5.1. Multi-View Images Inversion

The inversion of StyleGAN [17] has been demonstrated as a reliable method for real-world image manipulation. In 3D vision, we consider multi-view images inversion as an application of our model. For multi-view in-the-wild images, we first estimate the coarse rotation and elevation angles by a neural network(ResNet18 [13]) trained on our rendering images and corresponding gound-truth poses. Then we jointly optimize the pose and the generated mesh [40] until convergence. Fig. 8 shows the multi-view optimization result of Car class. Through inversion, DG3D can be extended to be an image-conditional or multi-view image-conditional framework.

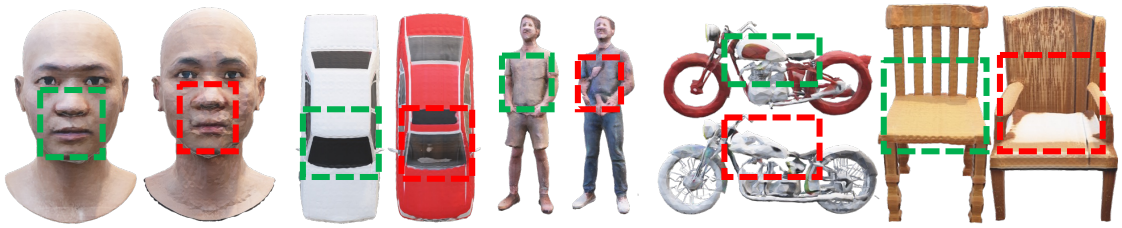

Figure 6: For the best view of details, we zoom in on the generated textured shapes of DG3D(Green) and GET3D [12](Red). DG3D generates shapes with smoother surfaces and more realistic and complete textures.

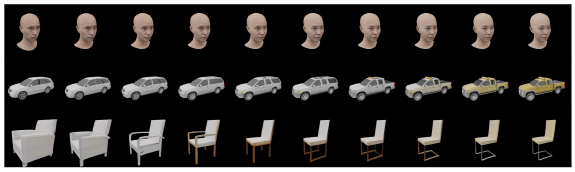

Figure 7: The geometry and texture latent codes are interpolated from left to right. We refer the reader to Appendix for more interpolation results.

Figure 8: The first five columns show the multi-view images and the final column shows the inversion result of DG3D.

5.2. Drag Edition

The recent progress in the interactive edition of images(*e.g.* DragGAN [26]) motivates us to design 3D manipulation tools. We incorporate DG3D with a drag-style edition tool[4] and enable 3D manipulation of the generated textured shapes. Fig. 9 shows manipulation results of $Twindom$ and Car class. The generated textured shapes can be seamlessly adjusted according to 2D control points on the projection space, which provides convenience for 3D creative design.

6. Conclusion

We propose DG3D, a novel framework that utilizes a Diffusion-Augmented Module for stabilizing the min-max game between a 3D generator and multiple 2D discriminators, and the Multi-Modal Discriminators, which offer many-sided supervision to improve the texture generation performance further. Experiments on synthetic datasets and real scans demonstrate the superiority of DG3D over current state-of-the-art methods. While we take a step closer to textured mesh generation of free topology, the resolution of generated meshes is still constrained by memory. The textured meshes generated by DG3D may lack details in tiny regions and tend to be watertight. In the future, we will try to adopt frameworks like the MinkowskiEngine[5] to design a 3D sparse tensor generator which will be memory efficient since only occupied space will consume the memory. Through this, we hope to produce shapes that still perform well in detailed tiny regions and serve more challenging applications.

Figure 9: Drag-style manipulation on generated textured shapes.

Table 3: Ablation of MD and DAM. MD denotes the Multi-Modal Discriminators, and DAM denotes the Diffusion-Augmented Module.

Category	Method	COV↑(%)		MMD↓($\times 10^{-3}$)		FID-3D↓
		LFD	CD	LFD	CD	
Car	DG3D(w/o MD)	71.96	59.21	1378	0.71	9.23
	DG3D(w/o DAM)	72.68	59.13	1389	0.68	8.12
	DG3D	**74.63**	**59.30**	**1363**	**0.67**	**6.09**
Chair	DG3D(w/o MD)	72.03	71.83	3099	3.74	21.07
	DG3D(w/o DAM)	72.54	73.21	3078	3.68	20.38
	DG3D	**74.80**	**73.65**	**3024**	**3.58**	**18.84**
Mbike	DG3D(w/o MD)	70.07	69.03	3086	1.73	48.72
	DG3D(w/o DAM)	71.42	67.33	3117	1.70	47.60
	DG3D	**75.34**	**71.23**	**3032**	**1.68**	**46.84**

[4]https://github.com/ashawkey/Drag3D

[5]https://github.com/NVIDIA/MinkowskiEngine

References

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3d point clouds. In *International conference on machine learning*, pages 40–49. PMLR, 2018. 6

[2] Dejan Azinović, Ricardo Martin-Brualla, Dan B Goldman, Matthias Nießner, and Justus Thies. Neural rgb-d surface reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6290–6301, 2022. 1

[3] Eric R Chan, Connor Z Lin, Matthew A Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas J Guibas, Jonathan Tremblay, Sameh Khamis, et al. Efficient geometry-aware 3d generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16123–16133, 2022. 1, 3, 4, 6

[4] Eric R Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5799–5809, 2021. 1, 3, 6

[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. 4, 6, 7

[6] Ding-Yun Chen, Xiao-Pei Tian, Yu-Te Shen, and Ming Ouhyoung. On visual similarity based 3d model retrieval. In *Computer graphics forum*, volume 22, pages 223–232. Wiley Online Library, 2003. 6

[7] Rui Chen, Yongwei Chen, Ningxin Jiao, and Kui Jia. Fantasia3d: Disentangling geometry and appearance for high-quality text-to-3d content creation. *arXiv preprint arXiv:2303.13873*, 2023. 3

[8] Zhiqin Chen, Vladimir G Kim, Matthew Fisher, Noam Aigerman, Hao Zhang, and Siddhartha Chaudhuri. Decor-gan: 3d shape detailization by conditional refinement. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15740–15749, 2021. 3, 5

[9] Julian Chibane, Thiemo Alldieck, and Gerard Pons-Moll. Implicit functions in feature space for 3d shape reconstruction and completion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6970–6981, 2020. 1

[10] Angela Dai, Christian Diller, and Matthias Nießner. Sg-nn: Sparse generative neural networks for self-supervised scene completion of rgb-d scans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 849–858, 2020. 1

[11] Jun Gao, Wenzheng Chen, Tommy Xiang, Alec Jacobson, Morgan McGuire, and Sanja Fidler. Learning deformable tetrahedral meshes for 3d reconstruction. *Advances In Neural Information Processing Systems*, 33:9936–9947, 2020. 1, 3

[12] Jun Gao, Tianchang Shen, Zian Wang, Wenzheng Chen, Kangxue Yin, Daiqing Li, Or Litany, Zan Gojcic, and Sanja Fidler. Get3d: A generative model of high quality 3d textured shapes learned from images. In *Advances In Neural Information Processing Systems*, 2022. 1, 2, 3, 4, 5, 6, 7, 8

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 7

[14] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017. 6

[15] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020. 5

[16] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. *Advances in neural information processing systems*, 33:12104–12114, 2020. 4

[17] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019. 5, 7

[18] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8110–8119, 2020. 1, 3

[19] Nikos Kolotouros, Thiemo Alldieck, Andrei Zanfir, Eduard Gabriel Bazavan, Mihai Fieraru, and Cristian Sminchisescu. Dreamhuman: Animatable 3d avatars from text. *arXiv preprint arXiv:2306.09329*, 2023. 5

[20] Samuli Laine, Janne Hellsten, Tero Karras, Yeongho Seol, Jaakko Lehtinen, and Timo Aila. Modular primitives for high-performance differentiable rendering. *ACM Transactions on Graphics (TOG)*, 39(6):1–14, 2020. 1, 3

[21] Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. Magic3d: High-resolution text-to-3d content creation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 300–309, 2023. 3

[22] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4460–4470, 2019. 1, 3, 6

[23] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 3

[24] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *Proceedings of the IEEE/CVF Conference on Computer*

Vision and Pattern Recognition, pages 11453–11464, 2021. 1, 3

[25] Michael Oechsle, Lars Mescheder, Michael Niemeyer, Thilo Strauss, and Andreas Geiger. Texture fields: Learning texture representations in function space. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4531–4540, 2019. 1

[26] Xingang Pan, Ayush Tewari, Thomas Leimkühler, Lingjie Liu, Abhimitra Meka, and Christian Theobalt. Drag your gan: Interactive point-based manipulation on the generative image manifold. *arXiv preprint arXiv:2305.10973*, 2023. 8

[27] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 165–174, 2019. 1

[28] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pages 523–540. Springer, 2020. 1, 3

[29] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022. 3

[30] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 3

[31] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022. 3

[32] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. *Advances in Neural Information Processing Systems*, 33:20154–20166, 2020. 6

[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. *Advances in Neural Information Processing Systems*, 34:6087–6101, 2021. 1, 2, 3, 5

[34] Yichun Shi, Divyansh Aggarwal, and Anil K Jain. Lifting 2d stylegan for 3d-aware face generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6258–6266, 2021. 1, 3

[35] Yawar Siddiqui, Justus Thies, Fangchang Ma, Qi Shan, Matthias Nießner, and Angela Dai. Retrievalfuse: Neural 3d scene reconstruction with a database. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12568–12577, 2021. 1

[36] Yawar Siddiqui, Justus Thies, Fangchang Ma, Qi Shan, Matthias Nießner, and Angela Dai. Texturify: Generating textures on 3d shape surfaces. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part III*, pages 72–88. Springer, 2022. 1, 2

[37] Maham Tanveer, Yizhi Wang, Ali Mahdavi-Amiri, and Hao Zhang. Ds-fusion: Artistic typography via discriminated and stylized diffusion. *arXiv preprint arXiv:2303.09604*, 2023. 3

[38] Zhendong Wang, Huangjie Zheng, Pengcheng He, Weizhu Chen, and Mingyuan Zhou. Diffusion-gan: Training gans with diffusion. *arXiv preprint arXiv:2206.02262*, 2022. 2, 5

[39] Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge Belongie, and Bharath Hariharan. Pointflow: 3d point cloud generation with continuous normalizing flows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4541–4550, 2019. 3, 6

[40] Zhenpei Yang, Zhile Ren, Miguel Angel Bautista, Zaiwei Zhang, Qi Shan, and Qixing Huang. Fvor: Robust joint shape and pose optimization for few-view object reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2497–2507, 2022. 7

[41] Rui Yu, Yue Dong, Pieter Peers, and Xin Tong. Learning texture generators for 3d shape collections from internet photo sets. In *British Machine Vision Conference*, 2021. 1

[42] Xiaohui Zeng, Arash Vahdat, Francis Williams, Zan Gojcic, Or Litany, Sanja Fidler, and Karsten Kreis. Lion: Latent point diffusion models for 3d shape generation. *arXiv preprint arXiv:2210.06978*, 2022. 3