# TPA3D: Triplane Attention for Fast Text-to-3D Generation

Bin-Shih Wu[1*], Hong-En Chen[2*], Sheng-Yu Huang[1], and Yu-Chiang Frank Wang[1,3]

[1] Graduate Institute of Communication Engineering, National Taiwan University
[2] National Taiwan University
[3] NVIDIA
{r12942090, b08901058, f08942095}@ntu.edu.tw
frankwang@nvidia.com

**Abstract.** Due to the lack of large-scale text-3D correspondence data, recent text-to-3D generation works mainly rely on utilizing 2D diffusion models for synthesizing 3D data. Since diffusion-based methods typically require significant optimization time for both training and inference, the use of GAN-based models would still be desirable for fast 3D generation. In this work, we propose Triplane Attention for text-guided 3D generation (TPA3D), an end-to-end trainable GAN-based deep learning model for fast text-to-3D generation. With only 3D shape data and their rendered 2D images observed during training, our TPA3D is designed to retrieve detailed visual descriptions for synthesizing the corresponding 3D mesh data. This is achieved by the proposed attention mechanisms on the extracted sentence and word-level text features. In our experiments, we show that TPA3D generates high-quality 3D textured shapes aligned with fine-grained descriptions, while impressive computation efficiency can be observed.

**Keywords:** 3D computer vision · text-to-3D generation

## 1 Introduction

3D object generation has become a thriving area of computer vision research in recent years, particularly with the increasing prevalence and requirement of AR/VR technologies, video game development, movie visual effects, and robotic simulations [16,23]. In the pursuit of automating the creation of 3D objects, numerous researchers strive to develop approaches for generating high-quality 3D assets. Early methods in 3D object generation [1,4,9,15,31,33,36,55] predominantly focus on learning representations that are both efficient and effective for generating 3D objects. However, the inherent unconditionality of these methods not only impedes the customization of generated shapes based on specific preferences or requirements but also limits the ease of subsequent manipulation of the resulting objects.

[*] These authors contributed equally to this work.

Motivated by the recent achievements in text-to-image generative models [20, 41, 42, 48], several studies [30, 51, 52] seek to emulate this success by conditioning on particular textual prompts. As a pioneer in the domain of text-guided 3D generation, Text2Shape [7] introduces the first large-scale dataset of natural language descriptions for 3D furniture objects and combines conditional WGAN [2] with 3D CNN to achieve supervised training for text-guided 3D object generation. This 3D dataset with human-annotated captions encourages various language-guided 3D generation methods [14, 28, 30, 34], especially with implicit 3D representations. These approaches directly learn the mapping between captions and corresponding 3D shapes, enabling the production of objects that align closely with specific details mentioned in the input text. While the inclusion of human-defined text supervision enhances the correlation between text prompt and generated shape, the scarcity of human-captioned 3D datasets for various object types (beyond furniture) restricts their applicability to specific object classes. Consequently, accurately aligning text descriptions with resulting 3D objects remains a challenging task.

To mitigate reliance on human-annotated datasets and achieve unsupervised text-to-3D generation, various methods [18, 29, 32, 37, 43, 44, 52] leverage pre-trained text-driven 2D image synthesis network [41, 42] or large vision and language models [11, 25, 26, 38] to address the inherent modality difference between text and vision element. For instance, some approaches [29, 32, 37] draw guidance from powerful 2D diffusion models [41, 42] by aligning the rendered RGB images from a random initialized NeRF [33] with text-guided 2D diffusion priors, ensuring the optimized NeRF corresponds accurately to the textual description. Despite enabling zero-shot generation, additional optimization processes of these methods significantly increase the inference time as noted in [29, 37], which limits real-time responsiveness to user inputs.

Conversely, visual-language-based (V-L-based) methods [18, 43, 44, 52] tackle this challenge by training a latent generator using either rendered image embeddings or pseudo text embedding encoded by CLIP [38]. Leveraging the aligned latent space of vision and text of CLIP, these methods can generate the correct latent for shape generation based on text prompt embeddings. While current V-L-based methods ease the necessity of paired 3D shapes with captions, they primarily generate shapes and textures at a general class and color level due to utilizing global (sentence) features of input texts as guidance for generating 3D objects. This implies the potential loss of detailed information in the text prompts, leading to similar shapes and textures generated from different fine-grained descriptions.

In this paper, we propose TriPlane Attention 3D Generator (TPA3D), a GAN-based text-guided 3D object generation network. Inspired by the unconditional GAN-based model of GET3D [15], our TPA3D only utilizes 3D objects and their rendered images for generating high-fidelity 3D textured mesh. With text features extracted from the pre-trained CLIP text encoder, our proposed TriPlane Attention (TPA) block performs sentence and word feature refinement for the geometry and texture triplanes, allowing fine-grained 3D textured mesh

to be produced via generator-based modules. We note that our TPA3D trains generators and discriminators in an unsupervised setting, performing instantly generation of high-fidelity 3D textured triplane corresponding to the detailed description without the human-annotated text-3D pairs for training supervision. This also makes GAN-based generation methods [18, 52] preferable over diffusion-based models [29, 37] which require significant optimization costs.

We now highlight the contributions of this work below:

- We propose a GAN-based network, named TriPlane Attention 3D Generator (TPA3D), performing sentence and word-level refinements of triplane features for fast text-guided 3D textured mesh generation.
- Our TriPlane Attention (TPA) performs plane-wise self-attention, cross-plane attention, and cross-word attention, allowing us to preserve intra-plane consistency, enhance 3D spatial connectivity, and integrate fine-grained information from the input text prompt for producing triplane features.
- We demonstrate our method outperforms the state-of-the-art GAN-based text-to-3D method in various evaluation metrics, and has better textual alignment than SDS-based methods.

## 2 Related Works

### 2.1 Text-Guided 2D Image Synthesis

With the advent of large-scale datasets [19, 45] containing text-image pairs, 2D text-guided generative models are developed, which leverage direct supervision through RGB images and their corresponding captions. To retrieve desirable information from the text prompt, recent methodologies [12,40–42] adopt attention mechanisms [50] as a key strategy to integrate fine-grained text features into their designs. Specifically, auto-regressive-based models [12,40] utilize transformers [8] to establish connections between text tokens and image patch tokens. Likewise, Diffusion Models such as Imagen [42] incorporate multiple cross-attention layers within the encoder and decoder, facilitating the learning of the denoising process. To enhance the resolution of the generated images, the Latent Diffusion Model (LDM) [41] suggests shifting the denoising process to the latent space rather than the pixel level.

However, the substantial computation cost incurred during inference still poses the interactivity concern [20]. GigaGAN [20] thus delves into the prospect of upscaling conditional StyleGAN [22] to accommodate large-scale datasets [24]. It further integrates cross-attention layers within the generator, focusing on both textual and visual features. While enabling the extraction of local details from comprehensive captions, the extension of these 2D methodologies to a 3D context remains an intricate challenge.

### 2.2 Text-Guided 3D Object Generation

Based on the success of text-guided image synthesis, numerous works [7, 28–30, 34, 37, 43, 44, 52] thrive in developing approaches for text-guided 3D object gen-

eration. Notably, Text2Shape [7] pioneers the text-to-3D domain by introducing the first large-scale 3D dataset with human-annotated captions for 3D furniture objects in ShapeNet [5] and optimizing a conditional WGAN [2] through supervised training. Subsequent studies [28,30,34] adopt similar supervisory strategies to design text-to-3D networks. For instance, TITG3SG [30] uses Implicit Maximum Likelihood Estimation (IMLE) [27] as the latent generator, which minimizes the similarity between the ground truth latent vector and the most similar generated latent vector from a set of generated results. AutoSDF [34] utilizes a VQ-VAE [49] to encode 3D objects and updates an additional auto-regressive Transformer [50] with text-3D pairs in the latent space during training to achieve text-guided latent vector generation. While these methods successfully achieve text-driven 3D object generation, the scarcity of human-annotated 3D datasets confines the applicability of these methods to specific classes. Consequently, recent endeavors share a common objective of reducing dependence on 3D datasets with human-define captions.

Recent advancements in cross-modality models [11, 25, 26, 38] have been observed. Pre-trained on large-scale image and text-paired data, these models have become prominent tools for bridging natural language with visual elements. Consequently, several studies [29, 37, 43, 44, 52] are exploring the potential of leveraging knowledge from these large language and vision models for text-guided 3D generation. Among these approaches, Magic3D [29] and DreamFusion [37] utilize diffusion models [41] to align rendered RGB images from a random initialized NeRF with text-guided 2D diffusion priors. By capturing fine-grained information from word-level features, these methods can ensure the generated shape corresponds to the specific requirements in the text prompt. Despite successfully achieving zero-shot text-guided 3D generation, additional optimization is typically required as reported in [29,37]. Such extra computation efforts would limit their practicality as 3D generation tools.

To address the above issue, alternative approaches [43, 44, 51, 52] aim to capitalize on the aligned vision and language latent space of CLIP [38] to generate text-conditioned latent for 3D objects generation. For instance, CLIP-Forge [43] introduces a flow-based model to learn the mapping between CLIP image embedding of rendered RGB image and the 3D shape latent. Leveraging the aligned text-image latent space, the latent generator predicts the suitable shape latent based on the text features of user input. Conversely, TAPS3D [52] introduces a novel captioning module to identify the best pseudo caption for the rendered images of 3D objects in the training set by maximizing the CLIP score between each image and the formulated text. By obtaining suitable pseudo captions for the 3D objects, TAPS3D directly fine-tunes a pre-trained conditional 3D generator [15] with the supervision of generated pseudo captions. Although the above technique does not require human-defined captions, their reliance on CLIP global text embedding as the primary guidance constrains their ability to precisely generate 3D objects matching detailed text inputs.

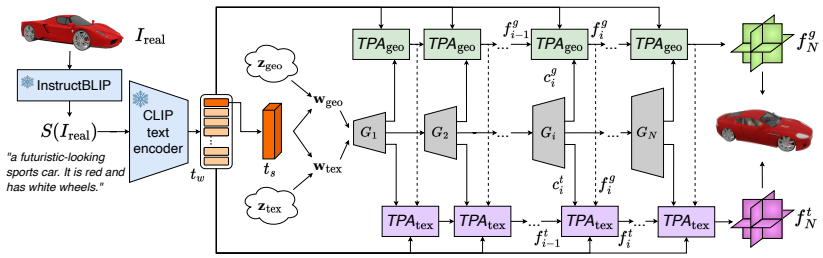



**Fig. 1: Overview of TPA3D for fast text-guided 3D generation.** By taking sentence and word-level features $t_s$ and $t_w$ as the inputs, TPA3D utilizes generator $G$ and triplane attention (TPA) modules to predict the associated triplane features for 3D textured mesh generation, with 3D content information properly observed. Following GET3D [15], each $G$ contains branches for geometry and texture synthesis. Note that InstructBLIP [11] is applied to produce pseudo captions from rendered images during training, while CLIP [38] extracts the resulting text features. And, $c_i$ and $f_i$ denote the sentence and word-level triplane features at each layer $i$, respectively.

## 3 Preliminary

For the sake of clarity, we provide a brief review of GET3D [15], which generates textured 3D shapes and serves as the generator backbone for our method. The model of GET3D is a GAN-based single-class unconditional 3D generator. Adapted from the generator of StyleGAN2 [22], GET3D first maps random noises $\mathbf{z}_{\text{geo}} \in \mathcal{N}(0, I)$ and $\mathbf{z}_{\text{tex}} \in \mathcal{N}(0, I)$ to latent vectors $\mathbf{w}_{\text{geo}} \in \mathbb{R}^{512}$ $\mathbf{w}_{\text{tex}} \in \mathbb{R}^{512}$. Subsequently, for the $i$-th layer of the generator, $\mathbf{w}_{\text{geo}}$ and $\mathbf{w}_{\text{tex}}$ are utilized to control the generation of the geometry triplane $c_i^g \in \mathbb{R}^{H_i \times W_i \times (3d)}$ and the texture triplane $c_i^t \in \mathbb{R}^{H_i \times W_i \times (3d)}$, where $d$ is the feature dimension, $H_i$ and $W_i$ denote the size of the triplane at that layer. After summing up all $c_i^g$ and $c_i^t$, the final triplanes $f_N^g = \sum_{i=1}^{N}(c_i^g)$ and $f_N^t = \sum_{i=1}^{N}(c_i^t)$ are fed to DMTet [47] to generate the output 3D textured mesh. In order to sum up all triplanes, a bilinear upsampling strategy is applied here (except for the triplanes with a size equal to $H_N \times W_N$) to align the resolution of all triplanes. Please refer to the supplementary materials for the details.

To train the generator of GET3D via 2D supervision, a differentiable renderer [24] is utilized to render the generated 3D textured mesh into 2D RGB image $I_{\text{fake}}$ and silhouette mask $M_{\text{fake}}$. Following the discriminator architecture of StyleGAN [21], GET3D applies two separate discriminators conditioned on the camera pose for the RGB and the mask images, respectively.

With the advantage of instant generation of high-fidelity textured shapes, our method is built on top of GET3D with the modification for text-guided 3D object generation. Specifically, to generate a textured mesh that aligns with the detailed description from the input text, our approach performs text-guided refinement on triplanes $c_i^g$ and $c_i^t$ by applying word-level information. With this modification, our model is able to generate high-fidelity 3D textured meshes while ensuring precise correspondences to the fine-grained textual conditions.

# 4 Method

## 4.1 Problem Formulation and Model Overview

We first define the problem definition and the notation used in this paper. Given an input text prompt $S$ describing desirable information of an object, our goal is to generate a 3D textured mesh that matches $S$ *without* reliance on human-defined text-3D pairs for training supervision. To achieve this goal, we propose a novel deep learning framework of TriPlane Attention 3D Generator (TPA3D).

As depicted in Figure 1, our TPA3D contains two network modules. First, we have *sentence-level triplane generators* $G = \{G_1, ..., G_N\}$ for generating sentence-level triplanes $c_i^g$ and $c_i^t$, with latent vectors $\mathbf{w}_{\text{geo}}$ and $\mathbf{w}_{\text{tex}}$ derived from sentence features $t_s$. The other module is *TriPlane Attention block* ($\text{TPA}_{\text{geo}}$ and $\text{TPA}_{\text{tex}}$), refining $c_i^g$ and $c_i^t$ into word-level triplanes $f_i^g$ and $f_i^t$ with word features $t_w$. Given an input text $S$, we apply a pre-trained CLIP text encoder [38] to convert $S$ into a sentence feature $t_s \in \mathbb{R}^{512}$ and word features $t_w \in \mathbb{R}^{77\times512}$ as global and detailed text information, respectively. We concatenate $t_s$ to randomly sampled noises $\mathbf{z}_{\text{geo}}$ and $\mathbf{z}_{\text{tex}}$ as conditions to generate latent vectors $\mathbf{w}_{\text{geo}}$ and $\mathbf{w}_{\text{tex}}$ for $G$ to produce sentence-level geometry triplane $c_i^g$ and texture triplane $c_i^t$ respectively from each layer $G_i$. To further enhance the consistency and connectivity of the generated triplanes and incorporate word-level information to capture geometric and textured details, the proposed novel TPA blocks are adopted to refine $c_i^g$ and $c_i^t$ and derive word-level geometry triplane $f_i^g$ and texture triplane $f_i^t$. Finally, we follow GET3D [15] to produce the 3D textured mesh from $f_N^g$ and $f_N^t$ via DMTet [47] as noted in Sect. 3. We now provide a detailed explanation of our TPA3D in the following subsections.

## 4.2 Pseudo Caption Generation

As discussed in Sect. 2, traditional text-guided 3D generation approaches [7, 30, 34] require human-annotated text-3D pairs to enable supervised training. To mitigate the reliance on human-annotated text-3D pairs, we leverage a pre-trained image captioning model of InstructBLIP [11] to produce the detailed pseudo caption $S(I_{\text{real}})$ for a rendered image $I_{\text{real}}$ of its 3D version, providing pseudo text-3D pairs for training. As suggested by [46], it is necessary to remove redundant phrases from generated captions to mitigate potential distraction for 3D generation (e.g., *"in the image"*, *"This is a 3D model"*, and *"black background"*). Please refer to supplementary materials for details of this filtering step. With such pseudo caption $S(I_{\text{real}})$ and rendered image $I_{\text{real}}$ pairs obtained, our TPA3D is subsequently designed to accommodate and leverage this detailed description.

## 4.3 Triplane Attention 3D Generator

We now detail how TPA3D realizes text-guided 3D generation, with the ability to produce 3D content with desirable shape and texture information. Given the input pseudo caption $S(I_{\text{real}})$, we apply the CLIP text encoder [38] to extract

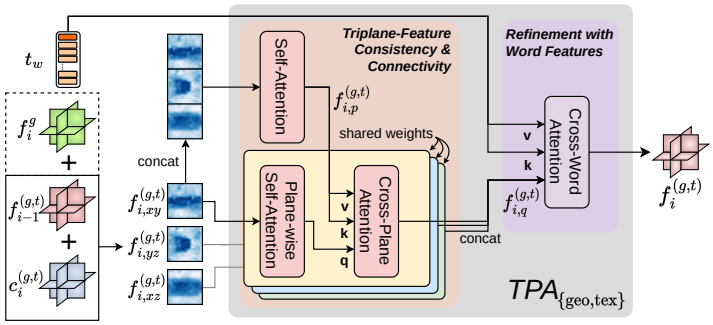



**Fig. 2: Design of TriPlane Attention (TPA).** TPA first performs plane-wise self-attention and cross-plane attention to 3D triplane features to enforce intra-plane consistency and 3D spatial connectivity, respectively. Cross-word attention is subsequently performed to exploit word-level features for incorporating detailed information. Note that for $\text{TPA}_{\text{tex}}$, additional geometry triplane features $f_i^g$ are included to incorporate geometry information for texture generation.

sentence features $t_s$ and word features $t_w$. For $t_s$, we directly use the output CLIP text embeddings [39]. As for $t_w$, we follow [20, 42] and extract the features from the second-last layer of the CLIP text encoder.

**Sentence-Level Triplane Generator.** As illustrated in Figure 1, the sentence-level triplane generator $G$ generates sentence-level triplanes $c_i^g \in \mathbb{R}^{H_i \times W_i \times (3d)}$ and $c_i^t \in \mathbb{R}^{H_i \times W_i \times (3d)}$ with latent vectors $\mathbf{w}_{\text{geo}}$ and $\mathbf{w}_{\text{tex}}$, which are conditioned on sentence features $t_s$, at each layer $G_i$. Following the generator architecture of GET3D [15], the sentence-level generator $G$ takes geometry latent vector $\mathbf{w}_{\text{geo}}$ and texture latent vector $\mathbf{w}_{\text{tex}}$ as inputs to generate the geometry triplane $c_i^g$ and the texture triplane $c_i^t$ at each layer $G_i$. For each layer, $G_i$ takes the layer features of $G_{i-1}$ as the input, and uses modulated convolution layers [22] with $\mathbf{w}_{\text{geo}}$ as style information to generate the layer features, which will be propagated to the next layer $G_{i+1}$ for the subsequent generation. To generate triplanes at each layer $G_i$, two additional modulated convolution layers are applied to layer features and take $\mathbf{w}_{\text{geo}}$ and $\mathbf{w}_{\text{tex}}$ as styles to generate the sentence-level geometry triplane $c_i^g$ and the texture triplane $c_i^t$ respectively. Because $\mathbf{w}_{\text{geo}}$ and $\mathbf{w}_{\text{tex}}$ are conditioned on the sentence features $t_s$, $c_i^g$ and $c_i^t$ only contain sentence-level information for textured mesh generation.

**Word-Level Triplane Refinement via TPA.** To further refine the above sentence-level triplanes with detailed information matching the text input, we propose the TriPlane Attention (TPA) block that performs word-level refinement to generate word-level geometry triplane $f_i^g \in \mathbb{R}^{H_i \times W_i \times (3d)}$ and texture triplane $f_i^t \in \mathbb{R}^{H_i \times W_i \times (3d)}$ accordingly. As depicted in Figure 2, for a $\text{TPA}_{\text{geo}}$ at layer $i$, we take the sentence-level geometry triplane $c_i^g$ generated by $G_i$ and the output

$f_{i-1}^g$ of the $\text{TPA}_{\text{geo}}$ in the previous layer to obtain three plane input features $f_{i,xy}^g, f_{i,yz}^g, f_{i,xz}^g \in \mathbb{R}^{H_i \times W_i \times d}$. For better understanding, we use $\langle \cdot \rangle$ as an operator to stack the feature channel (i.e., $\langle f_{i,xy}^g \cdot f_{i,yz}^g \cdot f_{i,xz}^g \rangle \in \mathbb{R}^{H_i \times W_i \times (3d)}$). Therefore, $\langle f_{i,xy}^g \cdot f_{i,yz}^g \cdot f_{i,xz}^g \rangle$, the input of the $\text{TPA}_{\text{geo}}$ is defined as,

$$\langle f_{i,xy}^g \cdot f_{i,yz}^g \cdot f_{i,xz}^g \rangle = f_{i-1}^g + c_i^g. \tag{1}$$

As for $\text{TPA}_{\text{tex}}$, we additionally include the output $f_i^g$ from $\text{TPA}_{\text{geo}}$ with a weight $\alpha = 0.5$ as the input of $\text{TPA}_{\text{tex}}$ to generate texture matches the corresponding geometry. Therefore, the input of $\text{TPA}_{\text{tex}}$ is defined as:

$$\langle f_{i,xy}^t \cdot f_{i,yz}^t \cdot f_{i,xz}^t \rangle = f_{i-1}^t + c_i^t + \alpha * f_i^g. \tag{2}$$

Note that the model architectures of $\text{TPA}_{\text{geo}}$ and $\text{TPA}_{\text{tex}}$ are the same except for their inputs.

**Triplane-Feature Consistency and Connectivity.** To ensure proper modeling of sentence-level object information through triplane features before incorporating word-level fine-grained details, our TPA (taking $\text{TPA}_{\text{geo}}$ for example) in Figure 2 initiates by prioritizing the acquisition of intra-plane consistency. Subsequently, additional efforts can be placed on fostering inter-plane 3D spatial connectivity across all planes, establishing a foundation for subsequent refinement processes. To inject sentence-level content pertaining to the desired object, we start from representation learning for each triplane. As depicted in the lower middle of Figure 2, this is achieved by observing intra-plane consistency for each triplane feature. That is, we choose to perform plane-wise self-attention on each plane feature to extract plane-wise content features. To further enhance the 3D spatial information inherent in our plane-wise content features and ensure comprehensive multi-aspect correspondence across different planes, we then employ a cross-plane attention mechanism to establish inter-plane connectivity. This involves treating the fused triplane feature $f_{i,p}^g$ (i.e., the output of applying self-attention on the concatenation of triplane features $f_{i,xy}^g, f_{i,yz}^g$, and $f_{i,xz}^g$) as both key and value, while employing the three plane-wise content features as queries to execute attention operations. We note that to decrease the number of parameters, the weights of cross-plane attention and plane-wise self-attention are shared between each plane feature. Finally, we concatenate three output plane features as self-refined features $f_{i,q}^g$ for later fine-grained word-level refinement.

**Refinement with Word Features.** To incorporate word-level information into triplane features for 3D generation, we perform the word-level refinement by cross-word attention. As shown in the right of TPA in Figure 2, the cross-word attention takes self-refined features $f_{i,q}^g$ as query, word features $t_w$ as key and value, to refine $f_{i,q}^g$ to the output word-level triplane $f_i^g$. Therefore, $f_i^g$ will include 3D spatial information and word-level information. With the refinement by $\text{TPA}_{\text{geo}}$ blocks and $\text{TPA}_{\text{tex}}$ blocks, we obtain final word-level triplanes $f_N^g$ and $f_N^t$, which are used to generate the 3D textured mesh that matches the detailed description. Different from DiffTF [3], our TPA blocks have additional

cross-word attention to utilize word features, so the generated triplanes contain fine-grained information corresponding to the detailed text prompt.

### 4.4 Text-Guided Discriminators

To train our TPA3D, we deploy and train the discriminators conditioned on the text inputs. Following GET3D [15], we use the same architectures of two discriminators $D_{\text{rgb}}$ and $D_{\text{mask}}$ for RGB images and masks, respectively. To properly design text-guided discriminators, we concatenate the sentence features $t_s$ to the camera pose condition as a new condition. In this case, the discriminators not only need to know whether the input rendered image is real or fake, but also have to judge whether the image matches the given detailed caption. For $D_{\text{rgb}}$ and $D_{\text{mask}}$, the adversarial objective is formulated as,

$$\begin{aligned}\mathcal{L}(D_{\text{rgb}}, G) = &\mathbb{E}_{t_s \in T}\ g(D_{\text{rgb}}(I_{\text{fake}}, t_s)) \\ &+ \mathbb{E}_{t_s \in T, I_{\text{real}} \in p_{\text{rgb}}}\ (g(-D_{\text{rgb}}(I_{\text{real}}, t_s)) \\ &+ \lambda \|\nabla D_{\text{rgb}}(I_{\text{real}})\|_2^2),\end{aligned} \tag{3}$$

$$\begin{aligned}\mathcal{L}(D_{\text{mask}}, G) = &\mathbb{E}_{t_s \in T}\ g(D_{\text{mask}}(M_{\text{fake}}, t_s)) \\ &+ \mathbb{E}_{t_s \in T, M_{\text{real}} \in p_{\text{mask}}}\ (g(-D_{\text{mask}}(M_{\text{real}}, t_s)) \\ &+ \lambda \|\nabla D_{\text{mask}}(M_{\text{real}})\|_2^2),\end{aligned} \tag{4}$$

where $g(x) = -\log(1 + \exp(-x))$. Note that $p_{\text{rgb}}$ and $p_{\text{mask}}$ represent the distributions of real rendered RGB images and silhouette masks, and $\lambda$ is a hyperparameter.

To introduce additional discriminative ability during training, we use additional negative pairs in the mismatching objective $\mathcal{L}_{\text{mis}}$ to make the model more sensitive to mismatched text conditions. Therefore, the mismatching objective is formulated as,

$$\begin{aligned}\mathcal{L}_{\text{mis}} = &\mathbb{E}_{t'_s \in T'}\ g(D_{\text{rgb}}(I_{\text{fake}}, t'_s)) \\ &+ \mathbb{E}_{t'_s \in T', I_{\text{real}} \in p_{\text{rgb}}}\ g(D_{\text{rgb}}(I_{\text{real}}, t'_s)) \\ &+ \mathbb{E}_{t'_s \in T'}\ g(D_{\text{mask}}(M_{\text{fake}}, t'_s)) \\ &+ \mathbb{E}_{t'_s \in T', M_{\text{real}} \in p_{\text{mask}}}\ g(D_{\text{mask}}(M_{\text{real}}, t'_s)),\end{aligned} \tag{5}$$

where $T'$ denotes the set of mismatched sentence features.

### 4.5 Training and Inference

**Training.** Since we use InstructBLIP [11] to generate detailed pseudo captions, we only require rendered image $I_{\text{real}}$ of the 3D object as our training data. As a result, the pseudo caption $S(I_{\text{real}})$, sentence features $t_s$, and word features $t_w$ can be produced as described above. The generator is trained to generate $I_{\text{fake}}$ and feed $I_{\text{real}}$ and $I_{\text{fake}}$ into the discriminators. To stabilize the training process, we use an additional CLIP similarity score for $I_{\text{fake}}$ and $t_s$ as a training objective $\mathcal{L}_{\text{clip}}$. Therefore, the overall training objective is defined as:

$$\mathcal{L} = \mathcal{L}(D_{\text{rgb}}, G) + \mathcal{L}(D_{\text{mask}}, G) + \mathcal{L}_{\text{mis}} + \mathcal{L}_{\text{clip}}. \tag{6}$$

**Table 1: Quantitative results in terms of (a) FID↓ and (b) CLIP R-Precision@5↑.** Compared to TAPS3D with only sentence-level features, our TPA3D performs additional word-level refinement and results in better visual quality and improved alignment between generated shapes and given text prompts. Note that *Acc.* represents *Accessory* in tables.

| Method | ShapeNet | | | OmniObject3D | |
|---|---|---|---|---|---|
| | Car | Chair | Motorbike | Vehicle | Acc. |
| GET3D [15] | **11.50** | **22.75** | **49.98** | 98.15 | 145.66 |
| TAPS3D [52] | 26.37 | 44.70 | 84.83 | 152.34 | 172.14 |
| Ours (TPA3D) | 18.50 | 38.11 | 77.69 | **68.80** | **83.31** |

**(a)** FID↓

| Method | ShapeNet | | | OmniObject3D | |
|---|---|---|---|---|---|
| | Car | Chair | Motorbike | Vehicle | Acc. |
| TAPS3D [52] | 12.55 | 7.52 | 5.00 | 9.47 | 6.67 |
| Ours (TPA3D) | **80.94** | **38.58** | **24.76** | **65.26** | **64.44** |

**(b)** CLIP R-Precision@5↑

**Inference.** For inference, one can replace the generated pseudo caption directly with the desirable input text prompt, fed into the generator for synthesizing the fine-grained 3D textured mesh that matches the input text prompt.

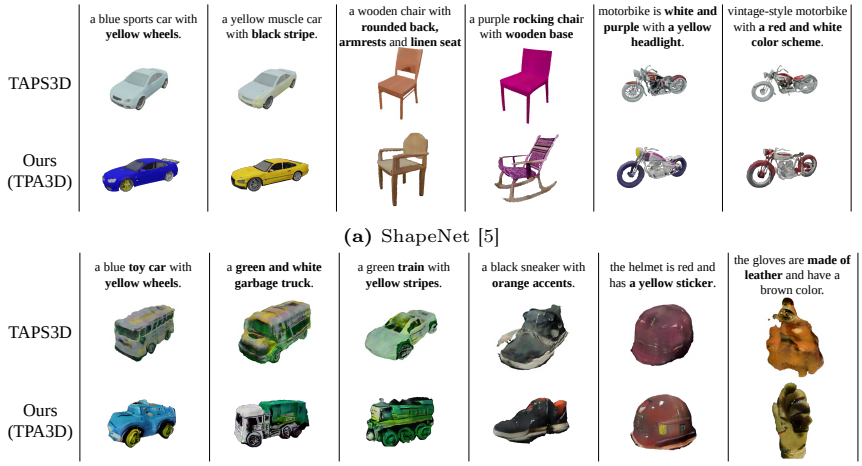
## 5 Experiments

### 5.1 Dataset

We train and evaluate our models on the synthetic 3D ShapeNet [5] dataset following [10, 15, 52, 54] and further include real-scanned 3D dataset OmniObject3D [5] to demonstrate its applicability on versatile real-world data. Specifically, we choose categories with diverse geometric and textural details, including *Car, Chair*, and *Motorbike* for ShapeNet. Since the number of objects for a single class in OmniObject3D is much smaller than ShapeNet, we combine *Toy Bus, Toy Car, Toy Truck,* and *Toy Train* as *Vehicle*, and combine *Gloves, Hat, Helmet,* and *Shoes* as *Accessory* for OmniObject3D. In our experiments, we generate images with a high resolution of 1024×1024 by rendering each textured shape from 24 randomly sampled camera angles. For categories with fewer shapes, like *Motorbike, Vehicle,* and *Accessory*, we increase the view count to 100 to ensure a comparable volume of rendered images. Finally, for fair comparison purposes, we generate *one* pseudo caption for each rendered image for quantitative and qualitative evaluation.

### 5.2 Quantitative Results

To quantitatively evaluate the capability of our method, we compare our proposed TPA3D with several existing state-of-the-art works, including GET3D [15] and TAPS3D [52], with the following evaluation metrics. To evaluate the fidelity and quality of generated shapes, we render 3D textured shapes into RGB images from 24 random camera views and compute Fréchet inception distances (FID) [17] of rendered images. As for the evaluation of consistency between text




**(a)** ShapeNet [5]

**(b)** OmniObject3D [53]

**Fig. 3: Qualitative comparisons with TAPS3D on (a) ShapeNet and (b) OmniObject3D.** Given detailed input prompts, our TPA3D generates accurate shapes aligned to prompts, while TAPS3D only realizes general classes and simple colors.

prompts and generated objects, we adopt the CLIP R-precision@5 [35] as our main metric. Lastly, all the models are trained and evaluated with pseudo captions generated by InstuctBLIP [11]. The results of FID, as presented in Table 1a, reveal that our TPA3D achieves comparable scores in FID with state-of-the-art 3D generator GET3D (as the performance upper bound for ShapeNet [5]) and outperforms text-guided 3D generation method TAPS3D in all classes. We note that, while GET3D excels in producing high-quality shapes, it is limited to unconditional shape generation and is hard to deal with high-diversity datasets composed of multiple classes without further guidance, and thus our TPA3D achieves higher scores in FID than GET3D on OmniObject3D [53]. As for the correspondence between input texts and generated objects, Table 1b demonstrates our TPA3D achieves higher CLIP R-precision across all classes compared to TAPS3D, which only utilizes sentence features. The result indicates that our generated shape better aligns with specified input text conditions. This validates the effectiveness of our approach, which leverages word features to enhance details in triplane features, resulting in our generated shape and texture retrieving detailed requirements specified in the text prompt.

### 5.3 Qualitative Results

To qualitatively evaluate the ability to deal with detailed descriptions, we first compare our TPA3D with TAPS3D [52]. The results in Figure 3 demonstrate that our model produces shapes accurately aligned to the text prompt while TAPS3D only comprehends simple modifiers and affects the output shape with

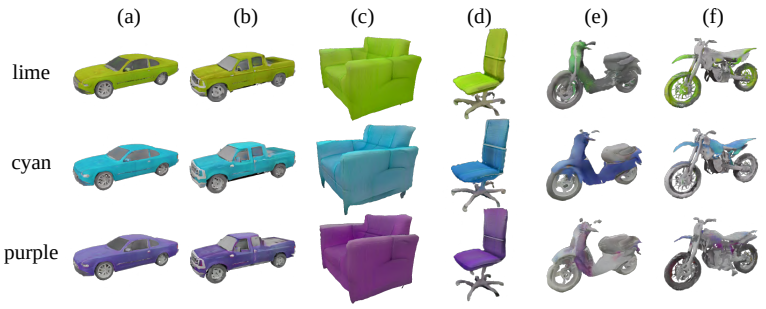



**Fig. 4: Example text-guided 3D generation results of TPA3D.** We consider input prompts of "a {color} {object}" with multiple colors and sub-classes for generation. Each column stands for a different color, while each row stands for a unique sub-class: (a)*"muscle car"* (b)*"pickup truck"* (c)*"sofa"* (d)*"office chair"* (e)*"scooter"* (f)*"dirt bike"*. Note that the same seeds are applied for sampling $\mathbf{z}_{\text{geo}}$ and $\mathbf{z}_{\text{tex}}$ for each row.

different colors. For ShapeNet [5] (see the third column of Figure 3a), TAPS3D only generates a wooden chair and ignores further details provided in the text prompt. In contrast, our TPA3D accurately captures all details such as *"rounded back"*, *"armrests"*, and *"linen seat"*. For OmniObject3D [53] (see the second column of Figure 3b), TAPS3D mixes the colors and misunderstands the accurate sub-class. In contrast, our TPA3D separates the colors *"green"* and *"white"*, and constructs the shape of *"garbage truck"*. This qualitatively verifies the effectiveness of our design of incorporating word-level triplane refinement in our TPA blocks. Furthermore, we present additional qualitative results for text-guided 3D generation in Figure 4 with different combinations of color and subclass. In this figure, we observe that our TPA3D exhibits impressive precision in generating textured shapes aligned with various combinations of subclass and color provided in text prompts. By fixing the random seed and subclass in each column, we can also observe that TPA3D only modifies textures with the changed color and maintains nearly identical shapes. This further verifies the effectiveness of word-level refinement in TPA for disentangling geometry and texture information.

### 5.4 Further Analysis

**Text-Guided Manipulation.** With the proper separation of geometry and texture triplane features, our TPA3D is able to manipulate the generated objects by simply changing the input text description and fixing the same random seed for sampling the initial noises $\mathbf{z}_{\text{geo}}$ and $\mathbf{z}_{\text{tex}}$. As shown in Figure 5, we first generate a chair object via the input text *"a wooden chair"*. By adding different text descriptions to the original one, our TPA3D manipulates the original chair accordingly without changing details unrelated to the additional descriptions. Such a manipulation property may improve its practicability as a 3D content creation tool for users to control the output incrementally.


**Fig. 5: Examples of chair manipulation by adding different detailed text descriptions.** The left shows a chair generated from the input text *"a wooden chair"*. With the same random seed for sampling $\mathbf{z}_{\text{geo}}$ and $\mathbf{z}_{\text{tex}}$, five distinct manipulations are produced by adding different detailed text descriptions.

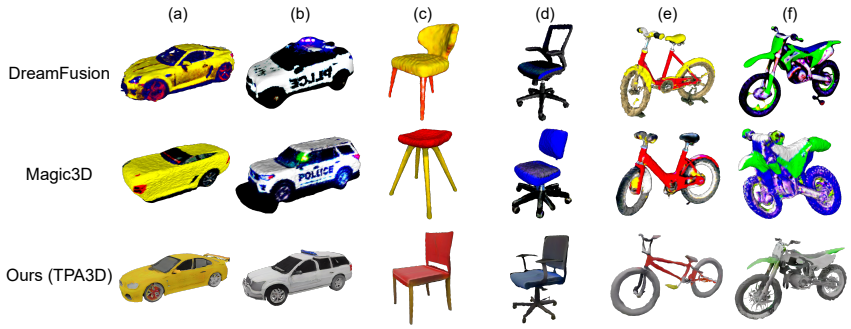



**Fig. 6: Qualitative comparisons with SDS-based methods.** Each column takes a unique text prompt of (a)*"a yellow sports car with red wheel and tinted window"*, (b)*"a white SUV with a blue police light on top of it"*, (c)*"the chair has a red seat and yellow legs"*, (d)*"a black office chair with a blue seat"*, (e)*"a red bicycle with yellow pedals"*, and (f)*"a green and white dirt bike"*.

**Comparison with SDS-Based Methods.** Since score distillation sampling (SDS) has shown significant performance in high-fidelity text-to-3D generation, we also compare TPA3D with SDS-based methods of DreamFusion [37] and Magic3D [29]. As shown in Figure 6, our method exhibits higher correspondence between complex input texts and generated objects. For example, in the fourth column of Figure 6, our TPA3D accurately separates the colors of *"black office chair"* and *"blue seat"*, while SDS-based methods either mix the colors or mismatch the colors to the parts of chairs. This is because, SDS-based methods heavily rely on pre-trained 2D diffusion models (e.g., Stable Diffusion [41]), and thus they are not able to generate 3D objects directly from complex textual descriptions. Such limitations (i.e., dependence on 2D diffusion models) have been discussed in previous works such as StructureDiffusion [13] and Attend-and-excite [6].

**Ablation Study on TPA.** Since the proposed TPA module serves as the major technical component in TPA3D, we conduct several ablation studies to

**Table 2: Runtime comparisons for diffusion/GAN-based generative models.** We compare the inference time reported in [29, 52]. For TAPS3D and TPA3D, we calculate the average time by generating 1000 samples with different text prompts.

| Method | Device | Output | Time |
|---|---|---|---|
| DreamFusion [37] | TPUv4 machine | Rendering | 90 min |
| Magic3D [29] | NVIDIA A100 x8 | Rendering | 40 min |
| TITG3SG [30] | Telsa V100-32G | Voxel | 2.21 sec |
| TAPS3D [52] | Telsa V100-32G | Rendering | 0.05 sec |
| TAPS3D [52] | Telsa V100-32G | Mesh | 1.03 sec |
| Ours (TPA3D) | Telsa V100-32G | Rendering | 0.09 sec |
| Ours (TPA3D ) | Telsa V100-32G | Mesh | 2.87 sec |

verify the design of TPA. In particular, we assess the design of TPA in three aspects: functions of cross-plane and cross-word attention in TPA, performances with different numbers of TPA blocks, and the improvement on the shape quality and the textual alignment with only $\text{TPA}_{\text{geo}}$ or $\text{TPA}_{\text{tex}}$. Due to page limitations, please refer to Sect. B, Table A1, and Table A2 in the supplementary materials for the complete ablation study results.

**Inference Speed Comparison.** To assess the real-time performance of each text-guided 3D generative model, we present the inference time of existing methods (reported from [29,52]) in Table 2. Notably, SDS-loss optimization-based approaches [29,37] require tens of minutes to complete the inference time optimization for each text input. In contrast, our proposed method maintains an instant inference speed comparable to GAN-based networks. Our approach achieves high-resolution image renderings at 1024×1024 in just tens of milliseconds and generates textured meshes within three seconds, similar to the performance of other GAN-based generators such as TAPS3D [52] and TITG3SG [30].

## 6 Conclusion

In this paper, we proposed TPA3D, a GAN-based deep learning framework for fast text-guided 3D object generation. With only access to 3D shape data and their rendered 2D images, we utilized a pre-trained image captioning model and text encoder to generate detailed pseudo captions from the above visual data as the text condition. By observing the text condition, our TPA3D is able to extract geometry and texture triplane features for generating textured 3D meshes. Taking the sentence feature of the text description as input, the sentence-level generator of our TPA3D derives sentence-level triplane features. To enforce fine-grained details from the word-level descriptions, the introduced TPA block further performs word-level refinement during generation. From the experiments, we demonstrate the ability of TPA3D in matching the generated textured mesh to the detailed description while retaining sufficient fidelity.

## Acknowledgement

This work is supported in part by the National Science and Technology Council via grant NSTC 112-2634-F-002-007 and NSTC 113-2640-E-002-003, and the Center of Data Intelligence: Technologies, Applications, and Systems, National Taiwan University (grant nos.113L900902, from the Featured Areas Research Center Program within the framework of the Higher Education Sprout Project by the Ministry of Education (MOE) of Taiwan). We also thank the National Center for High-performance Computing (NCHC) for providing computational and storage resources.

## References

1. Achlioptas, P., Diamanti, O., Mitliagkas, I., Guibas, L.: Learning representations and generative models for 3d point clouds. In: International conference on machine learning. pp. 40–49. PMLR (2018)
2. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein generative adversarial networks. In: International conference on machine learning. pp. 214–223. PMLR (2017)
3. Cao, Z., Hong, F., Wu, T., Pan, L., Liu, Z.: Large-vocabulary 3d diffusion model with transformer. arXiv preprint arXiv:2309.07920 (2023)
4. Chan, E.R., Lin, C.Z., Chan, M.A., Nagano, K., Pan, B., De Mello, S., Gallo, O., Guibas, L.J., Tremblay, J., Khamis, S., et al.: Efficient geometry-aware 3d generative adversarial networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 16123–16133 (2022)
5. Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., et al.: Shapenet: An information-rich 3d model repository. arXiv preprint arXiv:1512.03012 (2015)
6. Chefer, H., Alaluf, Y., Vinker, Y., Wolf, L., Cohen-Or, D.: Attend-and-excite: Attention-based semantic guidance for text-to-image diffusion models. ACM Transactions on Graphics (TOG) **42**(4), 1–10 (2023)
7. Chen, K., Choy, C.B., Savva, M., Chang, A.X., Funkhouser, T., Savarese, S.: Text2shape: Generating shapes from natural language by learning joint embeddings. In: Computer Vision–ACCV 2018: 14th Asian Conference on Computer Vision, Perth, Australia, December 2–6, 2018, Revised Selected Papers, Part III 14. pp. 100–116. Springer (2019)
8. Chen, M., Radford, A., Child, R., Wu, J., Jun, H., Luan, D., Sutskever, I.: Generative pretraining from pixels. In: International conference on machine learning. pp. 1691–1703. PMLR (2020)
9. Chen, Z., Zhang, H.: Learning implicit fields for generative shape modeling. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5939–5948 (2019)
10. Cheng, Y.C., Lee, H.Y., Tulyakov, S., Schwing, A.G., Gui, L.Y.: Sdfusion: Multimodal 3d shape completion, reconstruction, and generation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4456–4465 (2023)
11. Dai, W., Li, J., Li, D., Tiong, A.M.H., Zhao, J., Wang, W., Li, B., Fung, P., Hoi, S.: Instructblip: Towards general-purpose vision-language models with instruction tuning (2023)

12. Ding, M., Yang, Z., Hong, W., Zheng, W., Zhou, C., Yin, D., Lin, J., Zou, X., Shao, Z., Yang, H., et al.: Cogview: Mastering text-to-image generation via transformers. Advances in Neural Information Processing Systems **34**, 19822–19835 (2021)
13. Feng, W., He, X., Fu, T.J., Jampani, V., Akula, A., Narayana, P., Basu, S., Wang, X.E., Wang, W.Y.: Training-free structured diffusion guidance for compositional text-to-image synthesis. arXiv preprint arXiv:2212.05032 (2022)
14. Fu, R., Zhan, X., Chen, Y., Ritchie, D., Sridhar, S.: Shapecrafter: A recursive text-conditioned 3d shape generation model. Advances in Neural Information Processing Systems **35**, 8882–8895 (2022)
15. Gao, J., Shen, T., Wang, Z., Chen, W., Yin, K., Li, D., Litany, O., Gojcic, Z., Fidler, S.: Get3d: A generative model of high quality 3d textured shapes learned from images. Advances In Neural Information Processing Systems **35**, 31841–31854 (2022)
16. Ha, H., Agrawal, S., Song, S.: Fit2form: 3d generative model for robot gripper form design. In: Conference on Robot Learning. pp. 176–187. PMLR (2021)
17. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. Advances in neural information processing systems **30** (2017)
18. Huang, T., Zeng, Y., Dong, B., Xu, H., Xu, S., Lau, R.W., Zuo, W.: Textfield3d: Towards enhancing open-vocabulary 3d generation with noisy text fields. arXiv preprint arXiv:2309.17175 (2023)
19. Jia, C., Yang, Y., Xia, Y., Chen, Y.T., Parekh, Z., Pham, H., Le, Q., Sung, Y.H., Li, Z., Duerig, T.: Scaling up visual and vision-language representation learning with noisy text supervision. In: International conference on machine learning. pp. 4904–4916. PMLR (2021)
20. Kang, M., Zhu, J.Y., Zhang, R., Park, J., Shechtman, E., Paris, S., Park, T.: Scaling up gans for text-to-image synthesis. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10124–10134 (2023)
21. Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 4401–4410 (2019)
22. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of stylegan. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 8110–8119 (2020)
23. Katara, P., Xian, Z., Fragkiadaki, K.: Gen2sim: Scaling up robot learning in simulation with generative models. arXiv preprint arXiv:2310.18308 (2023)
24. Laine, S., Hellsten, J., Karras, T., Seol, Y., Lehtinen, J., Aila, T.: Modular primitives for high-performance differentiable rendering. ACM Transactions on Graphics (TOG) **39**(6), 1–14 (2020)
25. Li, J., Li, D., Savarese, S., Hoi, S.: Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. arXiv preprint arXiv:2301.12597 (2023)
26. Li, J., Li, D., Xiong, C., Hoi, S.: Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In: International Conference on Machine Learning. pp. 12888–12900. PMLR (2022)
27. Li, K., Malik, J.: Implicit maximum likelihood estimation. arXiv preprint arXiv:1809.09087 (2018)
28. Li, M., Duan, Y., Zhou, J., Lu, J.: Diffusion-sdf: Text-to-shape via voxelized diffusion. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12642–12651 (2023)

29. Lin, C.H., Gao, J., Tang, L., Takikawa, T., Zeng, X., Huang, X., Kreis, K., Fidler, S., Liu, M.Y., Lin, T.Y.: Magic3d: High-resolution text-to-3d content creation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 300–309 (2023)
30. Liu, Z., Wang, Y., Qi, X., Fu, C.W.: Towards implicit text-guided 3d shape generation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 17896–17906 (2022)
31. Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., Geiger, A.: Occupancy networks: Learning 3d reconstruction in function space. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 4460–4470 (2019)
32. Metzer, G., Richardson, E., Patashnik, O., Giryes, R., Cohen-Or, D.: Latent-nerf for shape-guided generation of 3d shapes and textures. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12663–12673 (2023)
33. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. Communications of the ACM **65**(1), 99–106 (2021)
34. Mittal, P., Cheng, Y.C., Singh, M., Tulsiani, S.: Autosdf: Shape priors for 3d completion, reconstruction and generation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 306–315 (2022)
35. Park, D.H., Azadi, S., Liu, X., Darrell, T., Rohrbach, A.: Benchmark for compositional text-to-image synthesis. In: Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1) (2021)
36. Pavllo, D., Kohler, J., Hofmann, T., Lucchi, A.: Learning generative models of textured 3d meshes from real-world images. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 13879–13889 (2021)
37. Poole, B., Jain, A., Barron, J.T., Mildenhall, B.: Dreamfusion: Text-to-3d using 2d diffusion. arXiv preprint arXiv:2209.14988 (2022)
38. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: International conference on machine learning. pp. 8748–8763. PMLR (2021)
39. Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., Chen, M.: Hierarchical text-conditional image generation with clip latents. arXiv preprint arXiv:2204.06125 **1**(2), 3 (2022)
40. Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., Sutskever, I.: Zero-shot text-to-image generation. In: International Conference on Machine Learning. pp. 8821–8831. PMLR (2021)
41. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 10684–10695 (2022)
42. Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E.L., Ghasemipour, K., Gontijo Lopes, R., Karagol Ayan, B., Salimans, T., et al.: Photorealistic text-to-image diffusion models with deep language understanding. Advances in Neural Information Processing Systems **35**, 36479–36494 (2022)
43. Sanghi, A., Chu, H., Lambourne, J.G., Wang, Y., Cheng, C.Y., Fumero, M., Malekshan, K.R.: Clip-forge: Towards zero-shot text-to-shape generation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 18603–18613 (2022)

44. Sanghi, A., Fu, R., Liu, V., Willis, K.D., Shayani, H., Khasahmadi, A.H., Sridhar, S., Ritchie, D.: Clip-sculptor: Zero-shot generation of high-fidelity and diverse shapes from natural language. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 18339–18348 (2023)
45. Schuhmann, C., Beaumont, R., Vencu, R., Gordon, C., Wightman, R., Cherti, M., Coombes, T., Katta, A., Mullis, C., Wortsman, M., et al.: Laion-5b: An open large-scale dataset for training next generation image-text models. Advances in Neural Information Processing Systems **35**, 25278–25294 (2022)
46. Sharma, P., Ding, N., Goodman, S., Soricut, R.: Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics. pp. 2556–2565 (2018)
47. Shen, T., Gao, J., Yin, K., Liu, M.Y., Fidler, S.: Deep marching tetrahedra: a hybrid representation for high-resolution 3d shape synthesis. Advances in Neural Information Processing Systems **34**, 6087–6101 (2021)
48. Tao, M., Bao, B.K., Tang, H., Xu, C.: Galip: Generative adversarial clips for text-to-image synthesis. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14214–14223 (2023)
49. Van Den Oord, A., Vinyals, O., et al.: Neural discrete representation learning. Advances in neural information processing systems **30** (2017)
50. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)
51. Wang, C., Chai, M., He, M., Chen, D., Liao, J.: Clip-nerf: Text-and-image driven manipulation of neural radiance fields. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3835–3844 (2022)
52. Wei, J., Wang, H., Feng, J., Lin, G., Yap, K.H.: Taps3d: Text-guided 3d textured shape generation from pseudo supervision. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 16805–16815 (2023)
53. Wu, T., Zhang, J., Fu, X., Wang, Y., Ren, J., Pan, L., Wu, W., Yang, L., Wang, J., Qian, C., et al.: Omniobject3d: Large-vocabulary 3d object dataset for realistic perception, reconstruction and generation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 803–814 (2023)
54. Xu, J., Wang, X., Cheng, W., Cao, Y.P., Shan, Y., Qie, X., Gao, S.: Dream3d: Zero-shot text-to-3d synthesis using 3d shape prior and text-to-image diffusion models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 20908–20918 (2023)
55. Yang, G., Huang, X., Hao, Z., Liu, M.Y., Belongie, S., Hariharan, B.: Pointflow: 3d point cloud generation with continuous normalizing flows. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 4541–4550 (2019)