EAI Endorsed Transactions
on AI and Robotics
Review Article EAI.EU

Methods and Strategies for 3D Content Creation Based on 3D Native Methods

Shun Fang[1,2,*], Xing Feng[2], and Yanna Lv[2]

[1] Peking University, Haidian, Beijing, 100091, China
[2] Lumverse Inc., Shijingshan, Beijing, 100043, China

Abstract

The paper provides a comprehensive overview of three neural network models, namely Point-E, 3DGen and Shap-E, with a focus on their overall processes, network structures, loss functions, as well as their strengths, weaknesses, and potential future research opportunities. Point-E, an efficient framework, generates 3D point clouds from complex text prompts, leveraging a text-to-image diffusion model followed by 3D point cloud creation. 3DGen, a novel architecture, integrates a Variational Autoencoder with a diffusion model to produce triplane features for conditional and unconditional 3D object generation. Shap-E, a conditional generative model, directly generates parameters of implicit functions, enabling the creation of textured meshes and neural radiance fields. While these models demonstrate significant advancements in 3D generation, areas for improvement include enhancing sample quality, optimizing computational efficiency, and handling more complex scenes. Future research could explore further integration of these models with other techniques and extend their capabilities to address these challenges.

Keywords: 3D Content Creation, Point-E, 3DGen, Shap-E, 3D Generation.

Received on 11 01 2024, accepted on 15 05 2024, published on 27 05 2024

doi: 10.4108/airo.5320

1. Introduction

The recent surge in the development of text-to-image generative models has revolutionized the process of creating and altering high-resolution images [21-30]. Nowadays, one can transform natural language descriptions into stunning visual representations within a matter of seconds [1-12]. Encouraged by these groundbreaking achievements, recent research efforts have delved into text-conditional generation across various other mediums, encompassing video and 3D objects alike [13-20]. In this study, we concentrate primarily on the intriguing problem of text-to-3D generation, which holds immense promise in democratizing the creation of 3D content for a diverse array of applications, ranging from immersive virtual reality experiences to captivating gaming worlds and intricate industrial designs.

Point-E, 3DGen, and Shap-E are three exemplary native 3D methods typically utilized for creating 3D content. These methods represent innovative approaches in the field of 3D generation, each with its unique characteristics and capabilities.

Point-E [1] emerges as an efficient framework for generating 3D point clouds from intricate text prompts. This method stands out through its ability to produce 3D models in a rapid timeframe of 1-2 minutes using a single GPU. The core of Point-E lies in its two-step generation process. Initially, it generates a synthetic view leveraging a text-to-image diffusion model. Subsequently, this generated image serves as the basis for creating a 3D point cloud, thus bridging the gap between textual descriptions and three-dimensional representations. While Point-E offers remarkable efficiency,

*Corresponding author. Email: fangshun@pku.org.cn

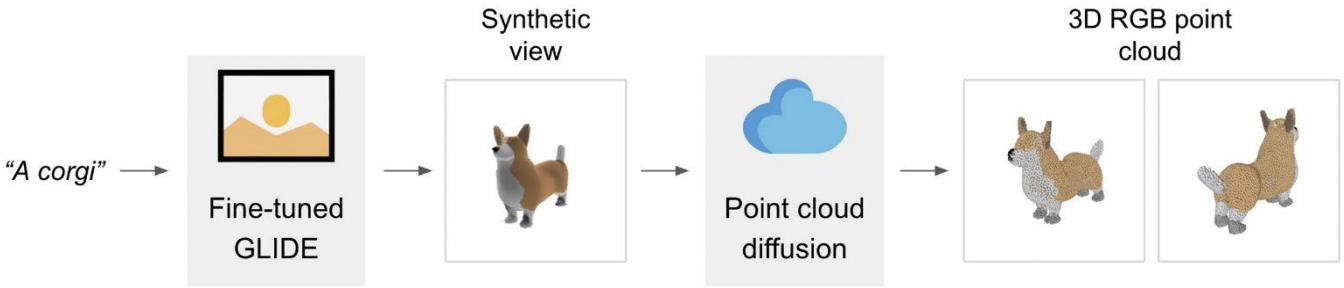

Figure 1. Overview of Point-E [1]

its sample quality might not yet match the state-of-the-art, presenting an area for future improvements.

3DGen [2] introduces a novel architecture that integrates a Variational Autoencoder (VAE) with a diffusion model. This integration enables the generation of triplane features, which are crucial for image-conditioned, text-conditioned, and unconditional 3D object generation. The combination of these techniques allows 3DGen to capture a wide range of 3D shapes and textures, offering greater flexibility and diversity in the generated outputs. However, the complexity of the architecture might pose challenges in terms of training and computational requirements.

Shap-E [3] represents a conditional generative model tailored for creating 3D assets. Its key innovation lies in the direct generation of the parameters of implicit functions. This approach enables the creation of textured meshes and Neural Radiance Fields (NeRF) [34], significantly enhancing the realism and detail of the generated 3D objects. Shap-E offers a promising alternative to traditional 3D modeling techniques, particularly in scenarios where textural and geometric complexity are paramount. Nevertheless, the method might still face challenges in handling extremely complex compositions or detailed textures.

Collectively, these three models represent significant advancements in the field of 3D generation from text prompts. Each model offers unique strengths and addresses specific challenges, opening up new research opportunities. Future work could explore enhancing the sample quality of Point-E, optimizing the training and computational efficiency of 3DGen, and extending the capabilities of Shap-E to handle more complex scenes and textures. Additionally, there is potential for further integration of these models with other techniques, such as reinforcement learning or adversarial training, to further enhance their performance and applicability.

2. Native Methods

The three Native Methods of 3D Content Creation are Point-e, 3DGen, and Shap-E. This section will analyze the workflow, neural network structure, loss function, and advantages of these three neural networks one by one.

2.1. Point-E: A Streamlined Framework for Text-to-3D Generation

Point-E [1] is a system designed to generate 3D point clouds from complex textual prompts. The method is significantly faster than state-of-the-art techniques, producing 3D models in just 1-2 minutes on a single GPU, compared to multiple GPU-hours required by leading approaches. Figure 1 presents a schematic summary of the Point-E workflow that transforms textual cues into 3D point cloud formations. The diagram likely incorporates exemplars of the pipeline's outcomes, juxtaposing the initial text query, the artificial image synthesised by GLIDE, and the corresponding 3D point cloud manifestation. This visual elucidation underscores the system's ability to convert textual inputs into intricate three-dimensional depictions of objects.

a) **Verbal Description Input Stage:** The first phase commences with a written description or directive that characterizes the sought-after 3D entity. This descriptive text serves as the compass for the subsequent 3D point cloud construction.

b) **Text-to-Visual Translation (GLIDE):** The textual instruction is fed into the text-to-image model known as GLIDE. This model assumes the task of producing a simulated visualization of the 3D object based on the provided written details. GLIDE, having undergone extensive pre-training and fine-tuning on 3D-rendered samples, aligns its output closely with the target distribution of the dataset.

c) **Image-to-3D Conversion Layer:** The artificial rendering delivered by GLIDE then serves as the conditional input to the succeeding segment of the pipeline. Here, a point cloud diffusion network takes this 2D image and evolves it into a 3D RGB point cloud representation. This specialized model is adept at interpreting the 2D imagery and reifying it into a point cloud encapsulating both the structural and chromatic properties of the object.

d) **Formation of 3D Point Cloud:** The culmination of the image-to-3D model's operation is a 3D point cloud that faithfully mirrors the content of the original textual hint.

Each individual point in the cloud carries a distinct RGB color attribute, thereby enabling a simultaneous representation of the object's hue and form.

e) **Pipeline Synopsis:** Figure 1 encapsulates the dual-core operations within the Point-E methodology: first, deriving a fabricated 2D perspective from a textual command leveraging GLIDE, followed by the transformation of this 2D representation into a 3D point cloud employing a diffusion algorithm. This two-step strategy has been crafted for efficiency, empowering the swift conversion of intricate text-based instructions into tangible 3D point cloud constructs.

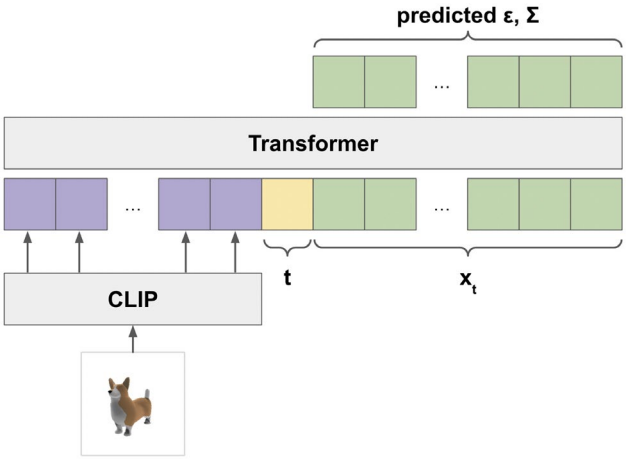

Figure 2. Overview of the Point-E system's architecture for generating 3D point clouds using a diffusion model [1].

Figure 2 demonstrates the mechanism whereby the Point-E system leverages a pre-trained Contrastive Language-Image Pre-training (CLIP) model to draw out characteristics from a 2D image and subsequently applies a Transformer-driven design to generate a 3D point cloud that adheres to these features. This system is engineered to estimate the noise distribution and mean value for each point within the cloud, thus incrementally reconstructing a 3D form from the noisy input data.

a) **Image Input:** The entry point for the system is a 2D image that acts as the blueprint for the ensuing 3D object synthesis. Typically, this image is synthesized by a text-to-image model, responding to the specific textual command issued by the user.

b) **Employment of a Pre-trained CLIP Module:** The input 2D image undergoes processing via a pre-trained Vision Transformer (ViT) component, specifically the ViT-L/14 configuration within CLIP paradigm. The CLIP module distills essential visual attributes from the image, which serve as guiding principles during the creation of the 3D point cloud.

c) **Extraction of Feature Vectors:** The outcome from the CLIP model includes high-level feature vectors extracted from its terminal layer, presenting dimensions of 256 by D', where D' signifies the attribute vector size. These vectors are further linearly projected into a tensor of shape 256 by D, with D being the dimension utilized within the Transformer architecture.

d) **Core Transformer Mechanism:** Central to the Point-E framework is a Transformer-based model. This Transformer ingests the projected feature vectors emanating from the CLIP model, complemented by ancillary contextual data, such as the current time step $\mathbf{t}$ and the noised version of the input $\boldsymbol{x_t}$, both of which are tokenized and introduced into the model.

e) **Processing of Input Parameters:** Each point in the point cloud is subjected to a linear layer to produce a K × D dimensional input matrix, where K denotes the total number of points. Concurrently, the time step $\mathbf{t}$ is channelled through a miniaturized Multi-Layer Perceptron (MLP) to generate a D-dimensional vector, which is prefixed to the Transformer's context array.

f) **Predictive Outputs:** The Transformer model yields a succession of tokens, from which the conclusive K tokens are extracted and mapped to infer the predicted epsilon ϵ and sigma Σ values pertinent to the K input points. These variables facilitate the denoising of the point cloud and ultimately lead to the formation of the 3D structure.

g) **Invariance to Permutations:** Significantly, the absence of positional encodings makes the model permutation-invariant concerning the input point clouds. This characteristic enables the model to process unordered point clouds while maintaining consistency with the output sequence, though the ordering of the output follows the input sequence.

Loss Function

The development of the loss function centers around training a diffusion model to closely approximate the conditional distribution $\mathbf{q}(\boldsymbol{x_{t-1}}|\boldsymbol{x_t})$ through the utilization of a neural network $\boldsymbol{p_\theta}(\boldsymbol{x_{t-1}}|\boldsymbol{x_t})$. The sampling procedure initiates with random Gaussian noise $\boldsymbol{x_T}$ and gradually reverses the noising process, culminating in the acquisition of a noiseless sample $\boldsymbol{x_0}$. The conditional distribution's mean is parameterized by predicting the effective noise added to a sample $\boldsymbol{x_t}$ denoted as ϵ. Although the variance Σ of $\boldsymbol{p_\theta}(\boldsymbol{x_{t-1}}|\boldsymbol{x_t})$ can be set to a heuristic value, superior results are achieved by predicting both the mean and variance. The diffusion sampling process can be interpreted through the framework of differential equations, enabling the employment of diverse solvers for sampling from these models.

Furthermore, the loss function incorporates guidance strategies such as classifier guidance and classifier-free guidance to strike a balance between sample diversity and fidelity in diffusion models. Classifier guidance involves the

utilization of gradients from a noise-aware classifier to perturb each sampling step, whereas classifier-free guidance conditions the model on class labels during sampling. The guidance scale parameter s regulates the influence of conditional predictions on the model's output.

In summary, the loss function strives to optimize the diffusion model by predicting the mean and variance of the conditional distribution, incorporating guidance strategies to enhance generation fidelity, and leveraging differential equations for sampling efficiency.

Experiment

Table 1 juxtaposes Point-E with other 3D generative models, demonstrating that Point-E constitutes a considerably speedier option for creating 3D point clouds from text prompts, albeit conceding somewhat on quality as quantified by the CLIP R-Precision [32] metric.

Table 1: comparison of Point-E with other 3D generative models [1].

Method	ViT-B/32	ViT-L/14	Latency
DreamFields [17]	78.6%	82.9%	~200 V100-hr†
CLIP-Mesh [31]	67.8%	74.5%	~17 V100-min*
DreamFusion [16]	75.1%	79.7%	~12 V100-hr†
Point-E (40M, text-only)	15.4%	16.2%	16 V100-sec
Point-E (40M)	35.5%	38.8%	1.0 V100-min
Point-E (300M)	40.3%	45.6%	1.2 V100-min
Point-E (1B)	41.1%	46.8%	1.5 V100-min
Conditioning Images	69.6%	86.6%	-

Table 1 furnishes a comparative examination of the Point-E system vis-à-vis alternative 3D generative methodologies, grounded on the CLIP R-Precision measure. The tabular presentation aims to illustrate the effectiveness of diverse methods in crafting 3D content from textual commands, coupled with their respective computational efficiency.

a) **Technique Enumeration: T**his column enumerates the numerous methodologies and systems that have undergone scrutiny for their capacity to generate 3D structures prompted by text.

b) **CLIP R-Precision (ViT-B/32): T**his column showcases the CLIP R-Precision scores attained by each method upon evaluation using a ViT-B/32 variant of the CLIP model. This metric gauges the efficacy of text-to-visual (specifically, text-to-3D in this scenario) generation models by juxtaposing the synthesized outputs against a benchmark collection of reference images.

c) **CLIP R-Precision (ViT-L/14):** Analogous to the preceding column, this section presents the CLIP R-Precision score yet resorts to a more sophisticated ViT-L/14 CLIP model, offering a potentially more refined appraisal of the generated material.

d) **Computational Latency:** This column delineates the processing duration or latency incurred by every method to produce a solitary sample. Reported times vary in minutes for certain methods, while others are converted to V100-minute units—a standard of computational time pegged to the performance of NVIDIA V100 GPUs—to facilitate uniform comparisons.

e) **Methodological Comparison:** The table juxtaposes Point-E against other cutting-edge approaches, including DreamFields [17], CLIP-Mesh [31], and DreamFusion [16]. Despite displaying marginally lower CLIP R-Precision scores, indicative of a potential inferiority in the quality of generated 3D models compared to top performers, Point-E boasts significantly shorter latencies, rendering it vastly swifter.

f) **Point-E Model Variants:** The table further dissects the performance profiles of differing Point-E iterations, differentiated by the scale of their neural network architectures (for instance, 40M, 300M, 1B denote parameter counts in millions). As model complexity escalates, CLIP R-Precision tends to improve, but this enhancement comes concurrent with a rise in computational time requirements.

g) **Efficiency-Quality Trade-off:** The table accentuates the balance between speed and accuracy. Methods like DreamFields indeed attain superior CLIP R-Precision scores, but they necessitate substantial computational resources and extended processing times, thereby compromising practicality in situations where expediency is paramount.

Advantages

The method exhibits several notable advantages:

a) **Efficiency:** With the utilization of a single GPU, the method is capable of generating 3D models in a remarkable timeframe of just 1-2 minutes. This represents a significant acceleration of one to two orders of magnitude compared to current state-of-the-art techniques.

b) **Balanced Performance:** Although the method may not achieve the pinnacle of sample quality compared to existing methods, its sampling speed offers a practical compromise for certain applications, making it a suitable choice for specific use cases.

c) **Reliable and High-quality Outcomes:** The model demonstrates remarkable consistency and produces high-quality 3D shapes in response to complex prompts. This demonstrates its proficiency in inferring a diverse range of shapes while accurately associating colors with pertinent aspects of the shapes.

d) **Guidance Strategies:** The method incorporates sophisticated guidance strategies, such as classifier guidance and classifier-free guidance, to strike a

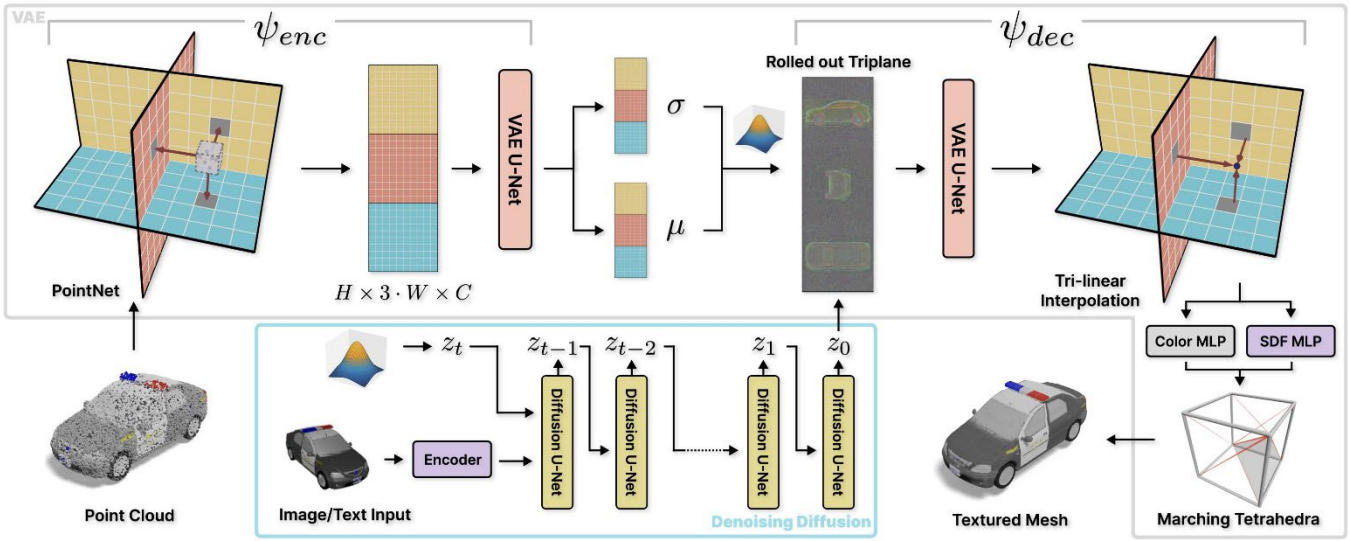

Figure 3. Two-stage Pipeline of 3DGen Architecture [2].

harmonious balance between sample diversity and fidelity in diffusion models. This integration has the potential to enhance the overall quality of generated content.

e) **Availability of Pre-trained Models:** The authors have generously shared their pre-trained point cloud diffusion models, along with evaluation code and models, to facilitate further research and development within the community.

In conclusion, the method shines in terms of its remarkable efficiency, practicality, capability to produce high-quality results, seamless integration of guidance strategies, and the generous release of pre-trained models for wider utilization.

2.2. 3DGen: Integrating VAEs and Diffusion for Conditional 3D Object Creation

The 3DGen [2] model demonstrates the ability to generate high-quality textured or untextured 3D meshes across multiple categories quickly and efficiently, outperforming previous methods in both geometry and texture generation.

a) **Introduction and Background:** Despite significant strides made in 3D creation technologies, including Point Cloud VAEs, triplane representations, neural implicit surfaces, and differentiable rendering, a scalable and efficacious solution to jointly produce top-notch textured and untextured 3D meshes remains unresolved.

b) **3DGen Framework Composition:** The 3DGen model operates through two core phases (Figure 3):

- Phase One - Triplane VAE Construction: An optimised VAE mechanism is employed to transform an optionally color-infused point cloud derived from the source mesh into a triplane Gaussian latent domain. It subsequently masters the process of reassembling a seamless, textured 3D mesh from this encoded data.
- Phase Two - Conditional Diffusion Process: A diffusion model undergoes training to forge triplane characteristics, which can be calibrated based on a provided image-text embedding. This functionality empowers image-guided, text-guided, and unconditional generation tasks.

c) **Approach:**

- Employment of Neural Fields: Leveraging the triplane representation to implicitly depict 3D entities or environments, the method uses three orthogonal feature planes for resourceful rendering purposes.
- Triplane VAE Processing: The encoder transposes point clouds into a distribution of triplane-encoded features, whereupon the decoder reconstructs the mesh by employing differentiable rendering techniques.
- Texture Forecasting: The VAE's capabilities are augmented to accommodate color-laden point clouds and anticipate surface hues as a texture map.
- Diffusion on Triplanes: A diffusion model is tutored on flattened triplanes—essentially 2D images that facilitate the application of existing image diffusion methodologies. This model integrates 3D-aware convolutions to bolster spatial consistency.

f) **Experimental Procedures:**

- Datasets such as ShapeNetCore and Objaverse are utilised for training and scalability assessments.
- The VAE training incorporates a rendering-based recovery loss, while the diffusion model is educated

with a predictive goal and a cosine-based noise scheduling.

- Experiments encompass both unconditional and conditional generations, demonstrating that the model significantly surpasses prior cutting-edge methods concerning geometric accuracy and texture synthesis.

g) **Evaluation and Insights:**

- The study underscores the advantages of incorporating render losses and the progressive training strategy of the VAE decoder to enhance mesh refinement.
- It emphasizes the criticality of selecting a conditioning encoder capable of capturing subtle variations in inputs.

Loss Function

The loss function described in this paper comprises multiple components, aiming to effectively train both the Tri-plane VAE and the Triplane Diffusion model. Regarding the Triplane VAE, the loss function incorporates a rendering-based reconstruction loss. This loss is facilitated by a differentiable renderer and the differentiable marching tetrahedra algorithm, enabling the preservation of intricate mesh details without resorting to pre-processing steps. Furthermore, a KL divergence loss is introduced to ensure that the encoder learns a distribution of triplane features that closely resembles a Gaussian prior. Additionally, a Laplacian mesh smoothing loss is integrated to enhance the smoothness of the reconstructed meshes. When training a textured VAE for color prediction, an additional loss term, referred to as $\boldsymbol{\mathcal{L}_{color}}$, is included. This loss compares the predicted surface colors with the ground truth surface colors, assisting the model in gener-ating semantically consistent textures alongside shapes.

The loss function $\boldsymbol{\mathcal{L}_{vae}}$ is used to train the VAE component of the model. It is a combined loss that includes several terms to ensure the model learns to reconstruct the input mesh accurately and maintain a certain distribution for the latent space.

$$\boldsymbol{\mathcal{L}_{vae}} = \left\|\boldsymbol{m_x} - \boldsymbol{m_x^{gt}}\right\|_2^2 + \left\|\boldsymbol{d_x} - \boldsymbol{d_x^{gt}}\right\|_1 + \boldsymbol{\lambda}\boldsymbol{\mathcal{L}_{smooth}} - \boldsymbol{\gamma D_{KL}}\left(\boldsymbol{q_{\psi_{enc}}(z|x)}\middle|\boldsymbol{p(h)}\right) \quad (1)$$

- $\left\|\boldsymbol{m_x} - \boldsymbol{m_x^{gt}}\right\|_2^2$: This term represents the mean squared error between the predicted mask silhouette $\boldsymbol{m_x}$ and the ground truth mask silhouette $\boldsymbol{m_x^{gt}}$. It measures the difference in the silhouettes of the reconstructed mesh and the original mesh.
- $\left\|\boldsymbol{d_x} - \boldsymbol{d_x^{gt}}\right\|_1$: This term represents the mean absolute error (L1 loss) between the predicted depth map $\boldsymbol{d_x}$ and the ground truth depth map $\boldsymbol{d_x^{gt}}$. It measures the difference in depth information between the reconstructed mesh and the original mesh.
- $\boldsymbol{\lambda\mathcal{L}_{smooth}}$: This is a smoothing loss that is multiplied by a scalar $\boldsymbol{\lambda}$. The smoothing loss encourages the generated mesh to be smooth, which is important for the quality of the final 3D model.
- $\boldsymbol{\gamma D_{KL}}\left(\boldsymbol{q_{\psi_{enc}}(z|x)}\middle|\boldsymbol{p(h)}\right)$: This term is the Kullback-Leibler divergence between the learned latent distribution $\boldsymbol{q_{\psi_{enc}}(z|x)}$ and a Gaussian prior $\boldsymbol{p(h)}$. The KL divergence measures how one probability distribution diverges from a second, expected probability distribution. The scalar $\boldsymbol{\gamma}$ weights this term. The goal is to keep the learned latent representation close to a Gaussian distribution.

The loss function $\mathcal{L}_{color}$ is used when training the model to predict textures on the mesh surface. It measures the difference between the predicted texture colors and the ground truth texture colors.

$$\mathcal{L}_{color} = \left\|c_x - c_x^{gt}\right\|_1 + \left\|c_x - c_x^{gt}\right\|_2^2 \quad (2)$$

- $\left\|c_x - c_x^{gt}\right\|_1$: This term represents the mean absolute error (L1 loss) between the predicted texture color c_x and the ground truth texture color c_x^{gt}. It captures the difference in color values between the predicted and actual texture.
- $\left\|c_x - c_x^{gt}\right\|_2^2$: This term represents the mean squared error (L2 loss) between the predicted texture color c_x and the ground truth texture color c_x^{gt}. It is another measure of the color difference, but it squares the differences, penalizing larger errors more heavily than the L1 loss.

Advantages

Table 2 presents a comparative analysis of different models on the task of image-conditioned mesh generation. The table is structured to show the performance of various models in terms of two key metrics: Chamfer-L1 distance and Shading Fréchet Inception Distance (FiD), for both head categories and tail categories. The table demonstrates that the 3DGen model, both with and without pretraining, outperforms the other models in terms of both geometric accuracy (lower Chamfer-L1 distances) and shading quality (lower FiD scores), especially for the head categories. The pretraining further improves the 3DGen model's performance, with notable reductions in both metrics compared to the non-pretrained version.

Table 2: Comparison between 3DGen and 3DILG [2].

Model	Head Categories		Tail Categories	
	Chamfer-L1 (↓)	Shading-FiD (↓)	Chamfer-L1 (↓)	Shading-FiD (↓)
CLIP-Forge† [14]	0.244	188.89	-	-
3DILG [33]	0.219	93.56	0.244	100.59
3DGen	0.181	80.31	0.192	94.88
3DGen+ pretraining	0.172	76.44	0.184	85.92

- Model: This column lists the names of the models being compared.
- Chamfer-L1 (↓): This column shows the Chamfer-L1 distance for each model. The Chamfer-L1 distance is a measure of the similarity between two point sets, and in this context, it's used to quantify the geometric similarity between the generated meshes and the ground truth meshes. Lower values are better, indicating a closer match to the ground truth.
- Shading-FiD (↓): This column displays the Shading FiD scores for each model. FiD is a measure of the similarity between two probability distributions and is often used to evaluate the quality of generated images or, in this case, the quality of the shading on generated meshes when rendered. Again, lower values are better, signifying more realistic and higher quality shading.

The method exhibits numerous advantages:

a) **Exceptional Mesh Generation Quality:** The utilization of the Triplane VAE and Triplane Diffusion models ensures the production of 3D meshes that are not only of high quality but also possess intricate details, realistic textures, and semantic consistency.

b) **Unique Differentiable Rendering Loss:** By opting for a rendering-based reconstruction loss, rather than relying on SDF or occupancy regression losses, the method maintains fine details in the mesh output without the need for preparatory steps like watertighting. This approach significantly enhances the quality of reconstruction.

c) **Versatile Texture Prediction:** The framework's adaptability extends to texture prediction, enabling the concurrent generation of semantically coherent textures alongside shapes.

d) **Efficient Model Training:** The models are trained efficiently, utilizing a comprehensive loss function that encompasses rendering-based reconstruction loss, KL divergence loss, Laplacian mesh smoothing loss, and color prediction loss. This approach ensures that all relevant aspects of mesh generation are addressed during training.

e) **Compatibility with Advanced Image Diffusion Models:** The latent space learned by the VAE, specifically the triplane format, aligns well with diffusion models, enabling seamless integration with cutting-edge image diffusion models.

f) **Staged VAE Training:** The method incorporates a staged training approach for the VAE, where the decoder undergoes separate fine-tuning to enhance fine details and mesh smoothness. This approach avoids the computational overhead associated with training directly at a higher tetrahedral resolution.

g) **Potent Conditioning Model:** The choice of conditioning encoder is paramount during diffusion training as it significantly affects the ability to capture subtle nuances in the input, leading to improved mesh alignment.

In summary, the method excels in its capability to produce high-quality 3D meshes, train models efficiently, preserve intricate details, enable texture prediction, and maintain compatibility with state-of-the-art image diffusion models.

2.3. Shap-E: Direct Generation of Implicit Function Parameters for High-Quality 3D Rendering

The neural network architecture of Shap-E [3] is comprised of an encoder that takes as input both point clouds and rendered views of a 3D asset. The encoder processes the input data via cross-attention and a transformer backbone, generating latent representations. These latent representations are subsequently passed through a latent bottleneck and projection layer, resulting in the determination of the MLP's weights. During the training phase, the MLP is queried, and its outputs are utilized in either an image reconstruction loss or a distillation loss. The encoder is pre-trained using the Adam optimizer, with specific hyperparameters and training iterations. The implicit neural representations are represented by 6-layer MLPs, employing specific activation functions and input coordinate expansions. Furthermore, the density head of the MLPs can be influenced by ray direction embeddings, ensuring view-consistency during testing. Figure 4 presents an elucidation of the encoding architecture adopted within the Shap-E model. This structure accepts dual forms of input to embody a 3D entity: point cloud data and several rendered perspectives. The illustration underscores the intricacy involved in transforming 3D assets into a format amenable to the generative prowess of the Shap-E model, thereby enabling the production of premium-grade 3D renditions.

a) **Input from Point Cloud:** The encoding process initiates with the receipt of a point cloud portrayal of the 3D object. This particular point cloud, featuring 16,384 points accompanied by RGB values, is subjected to preprocessing through a point convolutional layer, reducing its dimensionality to a cluster of 1,000 embeddings.

b) **Multi-perspective Point Cloud Input:** Beyond the point cloud, the encoder also ingests numerous rendered viewpoints of the same 3D asset. These snapshots are captured at randomly selected camera angles, with each foreground pixel enriched by surface coordinate information, culminating in a 256x256x7 image. An 8x8 patch embedding technique is applied to these rendered visuals, generating a series that encapsulates the multi-perspective point cloud essence.

c) **Cross-Attention Operation:** Thereafter, the encoder engages in cross-attention operations over the processed point cloud embeddings and the multiview point cloud sequence. This strategic mechanism

empowers the model to selectively attend to various aspects of the input while calculating the latent representation.

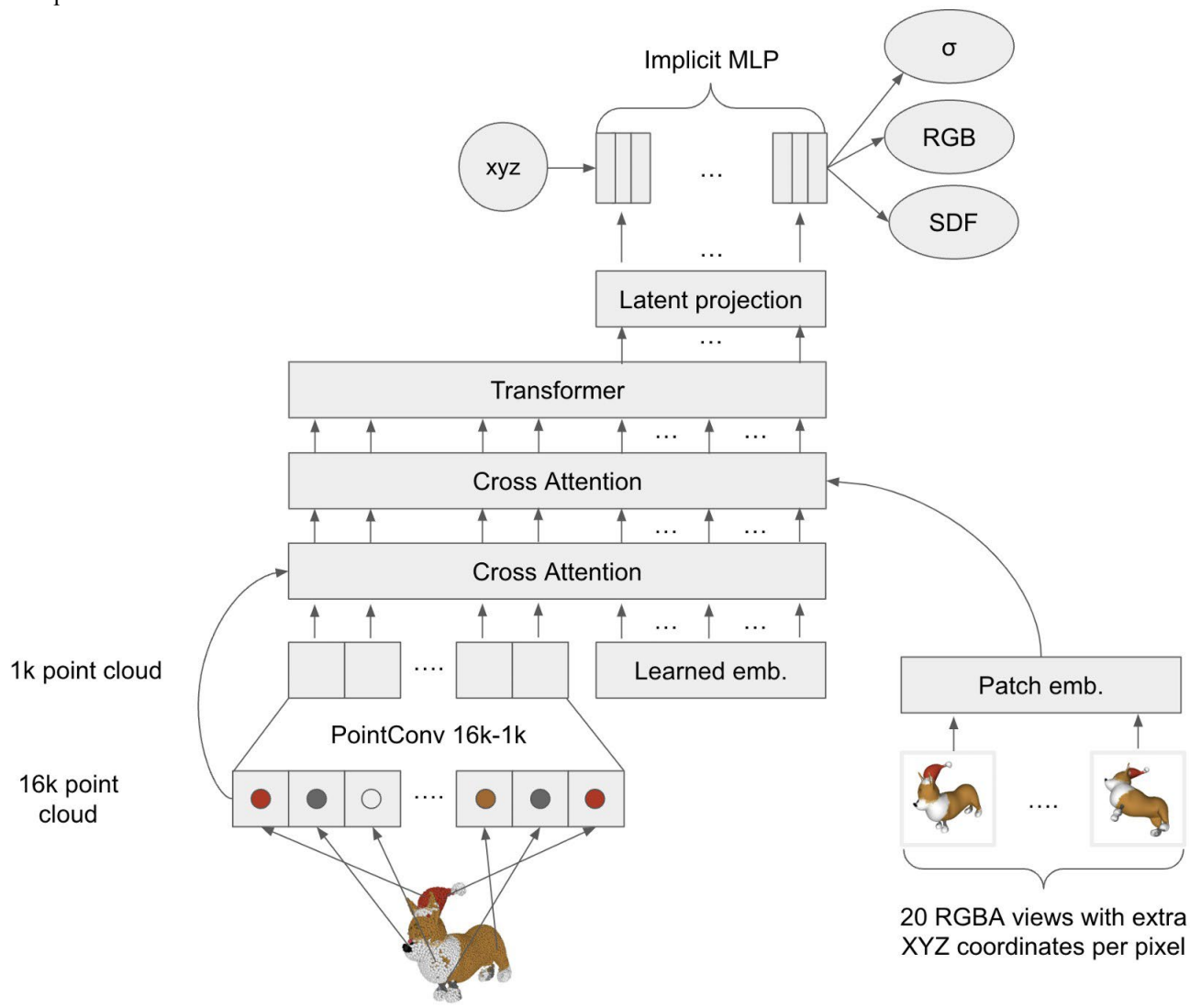

Figure 4. Overview of Shap-E Encoder Architecture [3]

d) **Application of Transformer Core:** Once the cross-attention layers have been traversed, a transformer core is brought into play to refine the now-updated embeddings. This transformer iteratively manipulates the sequence of vectors to yield a corresponding sequence of latent descriptors.

e) **Latent Compression and Projection Phase:** Every vector in the sequence of latent representations undergoes a latent compression and projection stage. The end product of this layer serves as an individual row in the ensuing MLP weight matrices assembly.

f) **Derivation of MLP Parameters:** The ultimate output from the encoder constitutes the parameters for an MLP that interprets the 3D asset as an implicit function. This MLP thereafter becomes instrumental in rendering the 3D asset in the form of both NeRF and a Signed Texture Field (STF).

g) **Rendering Execution:** The MLP parameters are deployed in the actual rendering of the 3D asset. During NeRF rendering, the model interrogates the MLP along camera ray paths to derive density and color estimates. Conversely, for STF rendering, the MLP-predicted SDF values and texture hues are harnessed to fabricate a mesh, which is then rendered utilizing a differentiable rendering engine.

Loss Function

The loss function employed during training encompasses either an image reconstruction loss or a distillation loss. The encoder generates latent representations of 3D assets, which are subsequently utilized to interrogate the MLP and obtain outputs. These outputs are subsequently contrasted with the ground truth data, utilizing the designated loss functions, to refine the model parameters. The loss function serves as a

pivotal component in directing the training procedure of the neural network architecture, ensuring that the implicit representations generated by the encoder align meticulously with the input data and desired outputs. Eq.3 is used to measure the difference between the predicted RGB colors and the actual RGB colors of the rendered 3D asset. The goal is to minimize the discrepancy between the generated image and the ground truth. The loss is calculated using the L1 norm, which is the sum of the absolute differences between the predicted and actual colors for both coarse and fine renderings.

$$L_{\text{RGB}} = E_{\text{r}\in R}\left[\left\|\hat{C}_c(\text{r}) - C(\text{r})\right\|_1 + \left\|\hat{C}_f(\text{r}) - C(\text{r})\right\|_1\right] \quad (3)$$

Where,

- r: Represents a ray in the rendering process.
- $\hat{C}_c(\text{r})$: The predicted RGB color for a coarse rendering of ray r.
- $\hat{C}_f(\text{r})$: The predicted RGB color for a fine rendering of ray r.
- $C(\text{r})$: The actual RGB color of the rendered 3D asset for ray r.

Eq.4 evaluates how well the model predicts the transmittance, which is related to the density of the volume along a ray and affects how much light passes through it.

$$L_T = E_{\text{r}\in R}\left[\left\|\hat{T}_c(\text{r}) - T(\text{r})\right\|_1 + \left\|\hat{T}_f(\text{r}) - T(\text{r})\right\|_1\right] \quad (4)$$

Where,

- r: A ray in the rendering process.
- $\hat{T}_c(\text{r})$: The predicted transmittance for a coarse rendering of ray r.
- $\hat{T}_f(\text{r})$: The predicted transmittance for a fine rendering of ray r.
- $T(\text{r})$: The actual transmittance value for ray r, derived from the alpha channel of the ground truth renderings.

Eq5. is specific to the rendering of 3D assets using SDF representation. It compares the rendered images of the reconstructed mesh generated by the SDF with the target RGBA renderings. The loss is computed using the L2 norm, which is the sum of the squared differences between the rendered mesh images and the target images, scaled by the square of the image resolution and the number of images.

$$\boldsymbol{L = \frac{1}{N \cdot s^2}\sum_{i=1}^{N}\|\text{Render}(\text{Mesh}_i) - \text{Image}_i\|_2^2 + L_{\text{RGB}} + L_{\text{T}}} \quad (5)$$

Where,

- N: The number of images in the dataset used for training.
- s: The image resolution, which is used to scale the loss.
- Mesh_i: The mesh constructed from the SDF for sample i.
- Image_i: The target RGBA rendering for image i.
- $\text{Render}(x)$: A function that renders a mesh x using a differentiable renderer.

Advantages

Table 3 furnishes an appraisal of the encoder's performance across distinct epochs of the training regimen. It scrutinizes two key indicators: the Peak Signal-to-Noise Ratio (PSNR) and the CLIP R-Precision [32] measure. These metrics serve to gauge the fidelity of 3D asset reconstructions generated by the encoder vis-à-vis the genuine reference renders. As Table 1 exhibits, the efficacy of the encoder in crafting high-fidelity 3D reconstructions progresses incrementally throughout the entire learning cycle.

Table 3: Evaluating encoder [3]

Stage	NeRF PSNR/dB	STF PSNR/dB	NeRF Point-E CLIP R-Precision	STF Point-E CLIP R-Precision
Pre-training (300K)	33.2	-	44.3%	-
Pre-training (600K)	34.5	-	45.2%	-
Distillation	32.9	23.9	42.6%	41.1%
Fine-tuning	35.4	31.3	45.3%	44.0%

a) **Stage:** This column delineates the specific juncture of the training progression:
   - Pre-training (300K): The preliminary phase where the encoder commences its education over 300,000 iterative steps.
   - Pre-training (600K): A subsequent extension of the preparatory training up to 600,000 iterations.
   - Distillation Phase: A stage where the encoder assimilates distillation strategies to hone its predictive prowess.
   - Fine-tuning: The conclusive adjustment phase wherein the encoder parameters are meticulously adjusted for optimal compatibility with both NeRF and STF rendering techniques.

b) **NeRF PSNR/dB:** Reflects the Peak Signal-to-Noise Ratio, expressed in decibels, pertaining to the NeRF rendering methodology. PSNR, a widely recognized standard, gauges the restoration quality of images or, in this context, synthesized 3D scenes; higher PSNR readings denote superior reconstruction.

c) **STF PSNR/dB:** Denotes the PSNR relevant to the Signed Distance Function rendering approach, assessing how closely the regenerated 3D structures

align with the original regarding both form and textural detail.

d) **NeRF Point·E CLIP R-Precision:** Presents the CLIP R-Precision score for the 3D assets once rendered using the NeRF method. Based on the CLIP model, this metric quantifies the extent to which the rendered visuals correspond to the textual characterizations of the 3D assets.

Table 4. Comparison of Point-E, 3DGen and Shap-E.

	Point-E	3DGen	Shap-E
Method Principle	It creates 3D point clouds by first generating a synthetic view from text and then generating a point cloud based on that image using another model.	A two-step pipeline is used, involving a triplane VAE to learn latent representations of textured meshes and a conditional diffusion model for generating triplane features.	It is a conditional model for 3D assets, generating parameters of implicit functions for textured meshes or neural radiance fields. It employs a two-stage training with an encoder and a conditional diffusion model.
Experiment Effects	It can produce 3D models in 1-2 minutes on a single GPU, offering a practical trade-off between speed and sample quality.	It enables high-quality textured or untextured 3D mesh generation across diverse categories quickly and efficiently.	It demonstrated the ability to generate complex and diverse 3D assets quickly when trained on a large dataset of paired 3D and text data.
Advantages	Significantly faster sampling time compared to state-of-the-art methods, making it more accessible for some use cases.	Outperforms previous work substantially in image-conditioned and unconditional generation on mesh quality and texture generation.	It converges faster and achieves comparable or better sample quality than Point-E, a generative model over point clouds, despite modeling a higher-dimensional output space.
Limitations	The method still falls short of the state-of-the-art in terms of sample quality and requires synthetic renderings.	The generality of the model is not yet on par with state-of-the-art image generation models, indicating room for improvement.	The sample quality of Shap-E still falls short of optimization-based approaches for text-conditional 3D generation.
Application Areas	Suitable for applications where rapid generation of 3D content is more critical than achieving the highest quality.	Game design, AR, and VR are potential application areas.	Potential applications include 3D content creation for gaming, virtual reality, and industrial design.
Future Research Directions	Training 3D generators on real-world images and extending the method to produce high-quality 3D representations like meshes or NeRFs could be future directions.	Closing the gap between the generality of 3DGen and state-of-the-art image generation models, using 2D image datasets as weak supervision, or utilizing 2D generative models to aid 3D generation could be future research focuses.	Combining Shap-E with optimization-based 3D generative techniques could lead to faster convergence and improved sample quality.

e) **STF Point·E CLIP R-Precision:** Offers the CLIP R-Precision value for the 3D assets after rendering them via the STF method. This score speaks to the degree of semantic correspondence between the recreated elements within the 3D shapes.

The tabular data reveals a consistent enhancement in the encoder's proficiency transitioning from the early pre-training to the fine-tuning stages for both NeRF and STF rendering modalities, as manifested by ascending PSNR values and CLIP R-Precision scores. Notably, however, the distillation phase appears to temporarily impede the NeRF reconstruction quality, as evidenced by a dip in both NeRF PSNR and CLIP R-Precision. Nevertheless, this setback is rectified and marginally exceeded during the final fine-tuning stage. This method boasts several advantages. Firstly, it is capable of enhancing sample quality by employing guidance techniques in conditional diffusion models. Secondly, the latent diffusion model proposed by the author demonstrates the ability to generate samples in a continuous latent space, offering a versatile means of producing diverse outputs. Furthermore, the author's text-conditional Shap-E model surpasses comparable models in terms of CLIP R-Precision and exhibits qualitatively distinct behavior for specific text prompts, highlighting the efficacy of their approach. Additionally, Shap-E boasts faster inference compared to Point-E as it does not necessitate an additional upsampling diffusion model, thus enhancing efficiency.

3. Comparative Analysis

The three Models share a commonality in utilizing diffusion models for 3D generation, yet they diverge in their approach to latent space representation and conditioning (Table 4). Shap-E and 3DGen adopt implicit representations, whereas Point-E opts for point clouds. In terms of speed versus quality, Point-E prioritizes speed, making it apt for rapid prototyping or time-sensitive applications. Conversely, Shap-E and 3DGen strive for a balance between the two. Textual conditioning is a focal point for Shap-E and Point-E, while 3DGen extends its capabilities to include image-conditioned generation. Scalability is highlighted by 3DGen, which demonstrates the potential of pre-training on vast datasets to enhance model generalizability.

Looking ahead, all Models suggest that future research should explore the integration of larger datasets, the enhancement of model generality, and the improvement of texture generation. Additionally, the potential of combining these models with other generative techniques or leveraging 2D models for 3D generation is emphasized as a promising direction for further exploration.

4. Conclusion

The three Models present innovative approaches to 3D generative modeling, each leveraging diffusion models to varying extents. Shap-E introduces a two-stage model that generates high-dimensional, multi-representational outputs for 3D assets, excelling in sample quality and convergence speed. Point-E, on the other hand, prioritizes rapid generation of 3D point clouds from textual prompts, offering a practical trade-off between speed and detail. Lastly, 3DGen employs a triplane VAE and conditional diffusion model to achieve efficient and high-quality generation of textured meshes, demonstrating significant improvements over previous methods and showcasing the potential for scalability. Collectively, these works advance the field of 3D content creation, offering solutions for diverse applications ranging from gaming to virtual reality, while also identifying areas for future research to bridge the gap towards the versatility of state-of-the-art image generation models.

References

[1] Nichol, A., Jun, H., Dhariwal, P., Mishkin, P. and Chen, M., 2022. Point-e: A system for generating 3d point clouds from complex prompts. arXiv:2212.08751, DOI: 10.48550/arXiv.2212.08751.

[2] Gupta, A., Xiong, W., Nie, Y., Jones, I. and Oğuz, B., 2023. 3dgen: Triplane latent diffusion for textured mesh generation. arXiv:2303.05371, DOI: DOI:10.48550/arXiv.2303.05371.

[3] Jun, H. and Nichol, A., 2023. Shap-e: Generating conditional 3d implicit functions. arXiv:2305.02463, DOI: DOI:10.48550/arXiv.2305.02463.

[4] Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M. and Sutskever, I., 2021, July. Zero-shot text-to-image generation. In International Conference on Machine Learning (pp. 8821-8831). PMLR, DOI: 10.48550/arXiv.2102.12092.

[5] Balaji, Y., Nah, S., Huang, X., Vahdat, A., Song, J., Kreis, K., Aittala, M., Aila, T., Laine, S., Catanzaro, B. and Karras, T., 2022. eDiffi-I: Text-to-image diffusion models with an ensemble of expert denoisers. arXiv:2211.01324, DOI:10.48550/arXiv.2211.01324.

[6] Feng, Z., Zhang, Z., Yu, X., Fang, Y., Li, L., Chen, X., Lu, Y., Liu, J., Yin, W., Feng, S. and Sun, Y., 2023. ERNIE-ViLG 2.0: Improving text-to-image diffusion model with knowledge-enhanced mixture-of-denoising-experts. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 10135-10145), DOI:10.1109/CVPR52729.2023.00977.

[7] Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E.L., Ghasemipour, K., Gontijo Lopes, R., Karagol Ayan, B., Salimans, T. and Ho, J., 2022. Photorealistic text-to-image diffusion models with deep language understanding. Advances in Neural Information Processing Systems, 35, pp.36479-36494, DOI:10.48550/arXiv.2205.11487.

[8] Yu, J., Xu, Y., Koh, J.Y., Luong, T., Baid, G., Wang, Z., Vasudevan, V., Ku, A., Yang, Y., Ayan, B.K. and Hutchinson, B., 2022. Scaling autoregressive models for content-rich text-to-image generation. arXiv:2206.10789, 2(3), p.5, DOI:10.48550/arXiv.2206.10789.

[9] Gafni, O., Polyak, A., Ashual, O., Sheynin, S., Parikh, D. and Taigman, Y., 2022, October. Make-a-scene: Scene-based text-to-image generation with human priors. In European Conference on Computer Vision (pp. 89-106). Cham: Springer Nature Switzerland, DOI:10.48550/arXiv.2203.13131.

[10] Ramesh, A., Dhariwal, P., Nichol, A., Chu, C. and Chen, M., 2022. Hierarchical text-conditional image generation with clip latents. arXiv:2204.06125, 1(2), p.3, DOI:10.48550/arXiv.2204.06125.

[11] Nichol, A., Dhariwal, P., Ramesh, A., Shyam, P., Mishkin, P., McGrew, B., Sutskever, I. and Chen, M., 2021. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. arXiv:2112.10741, DOI: 10.48550/arXiv.2112.10741.

[12] Ding, M., Yang, Z., Hong, W., Zheng, W., Zhou, C., Yin, D., Lin, J., Zou, X., Shao, Z., Yang, H. and Tang, J., 2021. CogView: Mastering text-to-image generation via transformers. Advances in Neural Information Processing Systems, 34, pp.19822-19835, DOI: 10.48550/arXiv.2105.13290.

[13] Sanghi, A., Fu, R., Liu, V., Willis, K.D., Shayani, H., Khasahmadi, A.H., Sridhar, S. and Ritchie, D., 2023. 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), DOI:10.1109/CVPR52729.2023.01759.

[14] Sanghi, A., Chu, H., Lambourne, J.G., Wang, Y., Cheng, C.Y., Fumero, M. and Malekshan, K.R., 2022. 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), DOI:10.1109/CVPR52688.2022.01805.

[15] Lin, C.H., Gao, J., Tang, L., Takikawa, T., Zeng, X., Huang, X., Kreis, K., Fidler, S., Liu, M.Y. and Lin, T.Y., 2023. Magic3d: High-resolution text-to-3d content creation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 300-309), DOI:10.1109/CVPR52729.2023.00037.

[16] Poole, B., Jain, A., Barron, J.T. and Mildenhall, B., 2022. Dreamfusion: Text-to-3d using 2d diffusion. arXiv:2209.14988, DOI:10.48550/arXiv.2209.14988.

[17] Jain, A., Mildenhall, B., Barron, J.T., Abbeel, P. and Poole, B., 2022. Zero-shot text-guided object generation with dream fields. 2022 IEEE. In CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 857-866), DOI:10.1109/CVPR52688.2022.00094.

[18] Ho, J., Salimans, T., Gritsenko, A., Chan, W., Norouzi, M., and Fleet, D. J. Video diffusion models. arXiv:2204.03458, 2022b, DOI:10.48550/arXiv.2204.03458.

[19] Singer, U., Polyak, A., Hayes, T., Yin, X., An, J., Zhang, S., Hu, Q., Yang, H., Ashual, O., Gafni, O. and Parikh, D., 2022. Make-a-video: Text-to-video generation without text-video data. arXiv:2209.14792, DOI: 10.48550/arXiv.2209.14792.

[20] Hong, W., Ding, M., Zheng, W., Liu, X. and Tang, J., 2022. Cogvideo: Large-scale pretraining for text-to-video generation via transformers. arXiv:2205.15868, DOI:10.48550/arXiv.2205.15868.

[21] Li, Y., Yang, G., Zhu, Y., Ding, X., & Gong, R. (2018). Probability model-based early merge mode decision for dependent views coding in 3D-HEVC. ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM), 14(4), 1-15.

[22] Yang, F., Chen, K., Yu, B., & Fang, D. (2014). A relaxed fixed point method for a mean curvature-based denoising model. Optimization Methods and Software, 29(2), 274-285.

[23] Wang, F., Li, Z. S., & Liao, G. P. (2014). Multifractal detrended fluctuation analysis for image texture feature representation. International Journal of Pattern Recognition and Artificial Intelligence, 28(03), 1455005.

[24] Yang, X., Lei, K., Peng, S., Cao, X., & Gao, X. (2018). Analytical expressions for the probability of false-alarm and decision threshold of Hadamard ratio detector in non-asymptotic scenarios. IEEE Communications Letters, 22(5), 1018-1021.

[25] Li, Y., Yang, G., Zhu, Y., Ding, X., & Gong, R. (2018). Probability model-based early merge mode decision for dependent views coding in 3D-HEVC. ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM), 14(4), 1-15.

[26] Peng, C., & Liao, B. (2023). Heavy-head sampling for fast imitation learning of machine learning based combinatorial auction solver. Neural Processing Letters, 55(1), 631-644.

[27] Jin, J. (2016). Multi-function current differencing cascaded transconductance amplifier (MCDCTA) and its application to current-mode multiphase sinusoidal oscillator. Wireless Personal Communications, 86, 367-383.

[28] Lu, H., Jin, L., Luo, X., Liao, B., Guo, D., & Xiao, L. (2019). RNN for solving perturbed time-varying underdetermined linear system with double bound limits on residual errors and state variables. IEEE Transactions on Industrial Informatics, 15(11), 5931-5942.

[29] Li, Z., Li, S., & Luo, X. (2021). An overview of calibration technology of industrial robots. IEEE/CAA Journal of Automatica Sinica, 8(1), 23-36.

[30] Li, Z., Li, S., Bamasag, O. O., Alhothali, A., & Luo, X. (2022). Diversified regularization enhanced training for effective manipulator calibration. IEEE Transactions on Neural Networks and Learning Systems.

[31] Khalid, N., Xie, T., Belilovsky, E., & Popa, T. (2023). Clip-mesh: Generating textured meshes from text using pretrained image-text models (Doctoral dissertation, Concordia University Montréal, Québec, Canada).

[32] Park, D. H., Azadi, S., Liu, X., Darrell, T., & Rohrbach, A. (2021, June). Benchmark for compositional text-to-image synthesis. In Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1).

[33] Zhang, B., Nießner, M., & Wonka, P. (2022). 3dilg: Irregular latent grids for 3d generative modeling. Advances in Neural Information Processing Systems, 35, 21871-21885.

[34] Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., & Ng, R. (2021). Nerf: Representing scenes as neural radiance fields for view synthesis. Communications of the ACM, 65(1), 99-106.