International Journal of Computer Vision
https://doi.org/10.1007/s11263-024-02298-y

Hyper-3DG: Text-to-3D Gaussian Generation via Hypergraph

Donglin Di[1,2,4] · Jiahui Yang[1,4] · Chaofan Luo[1,3] · Zhou Xue[2] · Wei Chen[1] · Xun Yang[3] · Yue Gao[2]

Received: 13 March 2024 / Accepted: 3 November 2024

Abstract
Text-to-3D generation represents an exciting field that has seen rapid advancements, facilitating the transformation of textual descriptions into detailed 3D models. However, current progress often neglects the intricate high-order correlation of geometry and texture within 3D objects, leading to challenges such as over-smoothness, over-saturation and the Janus problem. In this work, we propose a method named "3D Gaussian Generation via Hypergraph (Hyper-3DG)", designed to capture the sophisticated high-order correlations present within 3D objects. Our framework is anchored by a well-established mainflow and an essential module, named "Geometry and Texture Hypergraph Refiner (HGRefiner)". This module not only refines the representation of 3D Gaussians but also accelerates the update process of these 3D Gaussians by conducting the Patch-3DGS Hypergraph Learning on both explicit attributes and latent visual features. Our framework allows for the production of finely generated 3D objects within a cohesive optimization, effectively circumventing degradation. Extensive experimentation has shown that our proposed method significantly enhances the quality of 3D generation while incurring no additional computational overhead for the underlying framework. (Project code: https://github.com/yjhboy/Hyper3DG).

Keywords Text-to-3D generation · 3D Gaussian Splatting · Hypergraph learning

1 Introduction

The field of text-to-3D generation (Ma et al., 2023; Shi et al., 2023; Yu et al., 2024; Huang et al., 2024; Sun et al., 2024; Poole et al., 2023; Wang et al., 2024; Luo et al., 2024; Yi et al., 2023) represents a frontier in computational creativity, where converting textual descriptions into three-dimensional models is no longer a far-fetched possibility. This burgeoning task holds the potential to revolutionize a myriad of applications, from virtual reality and gaming to architectural design (Zhang, 2019), by enabling the creation of intricate and tangible 3D representations directly from textual input. As shown in Fig. 1, we present several cases generated by our text-to-3D model.

Despite these advances, there remains a notable oversight in addressing the intricate correlations of geometry and texture in 3D objects, leading to issues such as over-smoothness, over-saturation, incoherence, and the Janus Problem (Wang et al., 2024; Hong et al., 2024; Armandpour et al., 2023). For instance, as depicted in Fig. 2, refers to a phenomenon in 3D generation where the rendered objects exhibit multiple heads or faces. This issue is particularly problematic as it leads to inconsistent and unrealistic outputs in text-to-3D pipelines. It arises when the optimization process fails to

Communicated by Shengfeng He.

✉ Zhou Xue
xuezhou08@gmail.com

✉ Yue Gao
gaoyue@tsinghua.edu.cn

Donglin Di
didonglin@lixiang.com

Jiahui Yang
yangjiahui1@lixiang.com

Chaofan Luo
luochaofan@lixiang.com

Wei Chen
chenwei10@lixiang.com

Xun Yang
xyang21@ustc.edu.cn

1 Space AI, Li Auto, Beijing 101399, China

2 School of Software, Tsinghua University, Beijing 100084, China

3 School of Information Science and Technology, University of Science and Technology of China, Hefei 230026, Anhui, China

4 Harbin Institute of Technology, Harbin 150001, Heilongjiang, China

Published online: 16 December 2024

Fig. 1 Examples showcase the capability of text-to-3D content generations with our framework "3D Gaussian Generation via Hypergraph (Hyper-3DG)", which achieves creating high-fidelity 3D objects from text input. Please zoom in for more geometry and textural details

properly constrain the geometric consistency across different views, resulting in a visually disjointed final model (Hu et al., 2024; Huang et al., 2024). Specifically, as illustrated in the left cases of Fig. 2, when the 3D representation of "a bald eagle" (or "A pristine white wedding gown") undergoes rotation, the emergence of multiple faces and beaks becomes apparent. Upon rotating the depth map of the generated 3D object, it becomes evident that there are several points of disconnection between the flower's stem and the blossom (or the frying pan), leading to the appearance of several discontinuities.

Existing methods have predominantly relied on global geometry guidance such as point cloud diffusion (Chen et al., 2024) or multi-view diffusion (Shi et al., 2023; Hong et al., 2024; Armandpour et al., 2023; Liu et al., 2023) to maintain a global structural consistency. Specifically, these text-to-3D methods (Liang et al., 2024; Chen et al., 2024) typically updating the parameters of the entire 3D Gaussian Splatting (3DGS) independently. However, these methods fail to straightway capture the high-order correlations within various aspects of 3D objects, such as the texture of symmetrical or correlated parts. This oversight ultimately compromises the fidelity and usability of the generated models, as symmetrical structures (as local parts) are crucial for consistent updates to maintain coherence (Lai et al., 2024; Ho et al., 2024).

Therefore, we treat the local components of a 3D asset, such as an ear or a foot of an animal, as individual nodes. A 3D asset is then composed of these local nodes, meaning it undergoes optimization through them. We can further promote or restrict the degree of parameter optimization for these local nodes. With this approach, adjusting the number of nodes enables more constrained updates at different levels of 3DGS. When the number of nodes is sufficiently large, equal to the total number of points in the 3DGS, it allows for personalized parameter updates for each point. Given this, we propose a

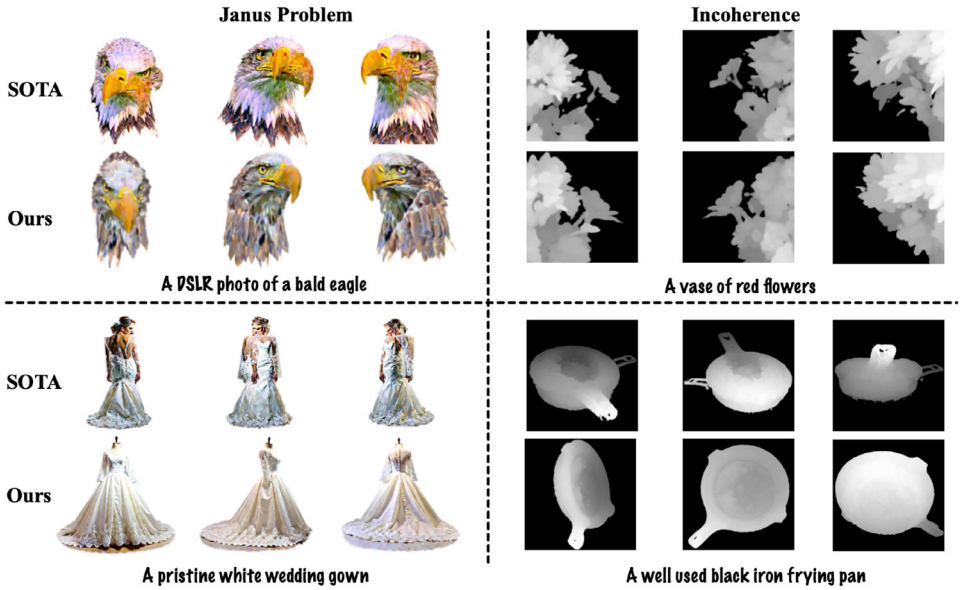

Fig. 2 Illustration the challenges of the Janus Problem and Incoherence issues. We compare the current state-of-the-art method, LucidDreamer (Liang et al., 2024) (referred to as "SOTA"), with our proposed approach, "Hyper-3DG". We zoom in the depth image (right part) to show the details of incoherence. From these cases, we can observe that our proposed method effectively addresses the Janus Problem and the incoherence issue, while significantly enhancing texture quality. Please zoom in to see more geometric and textural details

designed framework, "3D Gaussian Generation via Hypergraph (Hyper-3DG)", effectively engineered to address the complex, high-order correlations that underpin the geometry and texture of 3D objects. Our approach encompasses a primary workflow ("Mainflow") alongside a critical module, the "Geometry and Texture Hypergraph Refiner (HGRefiner)". At its core, our methodology adheres to a well-established pipeline, leveraging the capabilities of the pre-trained 3D generators and 2D diffusion models to initial and optimize the representations of 3D objects. Specifically, guided by the pre-trained 2D diffusion models, Denoising Diffusion Implicit Model (DDIM) (Ho et al., 2020; Song et al., 2020), our process begins with a "Warm-Up" phase, where we harness the power of a pre-trained 3D generator (*e.g.*, Point-E Nichol et al. 2022, Shap-E Jun and Nichol 2023) to generate a preliminary 3D object from textual descriptions. Upon obtaining this initial 3D object, our uniquely designed "HGRefiner" embarks on processing the 3D Gaussians by Peebles and Xie (2023) them into smaller, more manageable patch-level 3D Gaussian clusters, subsequently rendering them into 2D images. We then apply a pre-trained 2D image feature extractor (*e.g.*, ResNet He et al. 2016, ResNeXt Xie et al. 2017, ViT Dosovitskiy et al. 2021, Swin-T Liu et al. 2021, Dino Dino Caron et al. 2021) to capture the latent visual features of these 2D images, thereby enriching the 3D object's representation. This process not only improves patch-level semantic visual comprehension but also retains the rendering speed advantage inherent in 3D Gaussian Splatting. Through this innovative process, our HGRefiner adeptly establishes high-order correlations within the physical spatial space as well as the latent visual space of the 3D objects at the patch level, facilitated by hypergraph learning (Feng et al., 2019; Gao et al., 2022), named as "High-Order Refine" phase. In this way, the HGRefiner module refines the spatial and latent representations of 3D objects in a high-order, correlative manner, focusing on each individual part of the 3D objects. Furthermore, our methodology ensures consistency in the initialization process and hypergraph refinement by employing the same evaluation metric, the Interval Score Matching (ISM) loss (Liang et al., 2024). By maintaining the same loss, we effectively prevent the deterioration of these fundamental characteristics throughout the refinement process, ensuring the integrity and fidelity of the 3D objects remain intact. This meticulous refinement culminates in the generation of finely detailed 3D objects in response to textual prompts.

The key contributions of our work are summarized as follows:

1. We propose a designed framework to address high-order correlations within 3D objects, aiming to optimize both the geometry and texture of the generated 3D objects. To our knowledge, this represents the first trial of its kind in tackling these intricate correlations in the task of 3D generation;
2. The high-order correlative optimizing approach refines 3D Gaussian by fine-tuning both explicit attributes and latent visual features at a manageable, patch-level scale.
3. Our proposed method is designed for low coupling and is capable of significantly improving the performance of 3D generation without adding to the computational load for the various backbone models.

2 Related Work

In this section, we provide a concise overview of recent progress in the field of 3D representations, text-to-3D generation, and hypergraph learning.

2.1 3D Representations

Differentiable 3D representations play a pivotal role in optimization-based 3D generation.

One of the most commonly used representations is Neural Radiance Fields (NeRF) (Mildenhall et al., 2020), which represents a 3D scene as a continuous function mapping 5D coordinates (3D spatial coordinates and 2D viewing direction) to volume density and view-dependent emitted radiance. NeuS (Wang et al., 2021) adopts a set of signed distance functions (SDFs) to represent the surface of 3D objects. Plenoxels (plenoptic voxels) (Fridovich-Keil et al., 2022) represent 3D scenes via a 3D grid of spherical harmonics, enabling faster optimization than NeRF.

Recently, 3D Gaussian Splatting (Kerbl et al., 2023) has emerged as a promising approach for balancing optimization speed, rendering quality, and rendering speed. This method utilizes clusters of 3D Gaussians to explicitly represent a 3D scene, offering a wide range of flexible options for scene manipulation and rendering. Mip-Splatting (Yu et al., 2024) examined the aliasing effects caused by the dilation operation in 3D Gaussian Splatting and introduced a 3D smoothing filter to address this issue. AbsGS (Ye et al., 2024) focused on the adaptive density control mechanism of 3D Gaussians and proposed using the absolute value of the gradients to prevent gradient collisions. 2D Gaussian Splatting (Huang et al., 2024) replaced the ellipsoid primitives in 3D Gaussian Splatting with 2D ellipses to achieve a more accurate geometry. GaussianCube (Zhang et al., 2024) proposed a structured 3D Gaussian representation via optimal transport, allowing for efficient generative modeling. In our work, we focus on generating high-quality 3D Gaussians, aligning with the current popular choices and recent mainstream approaches.

2.2 Text-to-3D Generation

Early attempts at text-to-3D generation primarily utilized CLIP (Radford et al., 2021) as a guidance mechanism for optimization, often producing suboptimal results. To harness the powerful generative capabilities of diffusion models, Zero-1-to-3 (Liu et al., 2023) fine-tuned a pre-trained 2D diffusion model conditioned on camera parameters to elicit 3D priors from the 2D diffusion model. 3D assets were then reconstructed from the generated multi-view images. MVDream (Shi et al., 2023) proposed a multi-view diffusion framework to generate consistent multi-view images for 3D object synthesis. Wonder3D (Long et al., 2024) adapted a pre-trained 2D diffusion model into a cross-domain diffusion model to produce paired RGB images and normal maps, subsequently fusing them into textured meshes.

In addition to fine-tuning diffusion models on 3D datasets to generate explicit 3D guidance, another line of research has focused on utilizing a pre-trained 2D diffusion model to directly optimize 3D representations. These approaches usually incorporate differentiable 3D representations such as NeRF (Mildenhall et al., 2020), NeuS (Wang et al., 2021), and optimize their parameters through backpropagation. DreamFusion (Poole et al., 2023) proposed Score Distillation Sampling (SDS) to sample 3D parameters by optimizing a distillation loss. Score Jacobian Chaining (Wang et al., 2023) offered an alternative formulation and arrived at similar parametrizations as SDS. ProlificDreamer (Wang et al., 2024) analyzed the objective function of SDS and proposed a particle-based variational framework named Variational Score Distillation (VSD) that significantly improved the quality of generated content.

Recent works have incorporated SDS with Gaussian Splatting (Kerbl et al., 2023) to achieve faster optimization. DreamGaussian (Tang et al., 2023) proposed a multi-stage framework that optimizes coarse 3D Gaussians via SDS in the first stage, with meshes and UV maps extracted and refined subsequently. GSGEN (Chen et al., 2024) utilized the explicit representation of 3D Gaussians and applied a point cloud diffusion model for global geometric guidance. GaussianDreamer (Yi et al., 2023) focused on the initialization stage and proposed an augmentation strategy to improve performance. LucidDreamer (Liang et al., 2024) analyzed the SDS loss and proposed Interval Score Matching (ISM) to tackle the over-smoothness and inconsistency issues of the original SDS method.

In this work, we empirically follow the well-established mainstream architectural approaches (Wang et al., 2024; Liang et al., 2024; Chen et al., 2024) for 3D Gaussian generation as the primary workflow of our method. Building upon this mainstream pipeline, we further optimize the refiner component by employing a specially designed patch-level 3D Gaussian hypergraph neural network.

2.3 Hypergraph Learning

Hypergraph Learning (Gao et al., 2022, 2020; Di et al., 2022; Bai et al., 2021) has emerged as an effective approach for modeling complex relational data. Traditional graph learning methods (Kipf & Welling, 2017; Veličković et al., 2018) are limited to pairwise relationships, while hypergraphs provide a natural way to represent higher-order interactions among multiple entities. Hypergraph neural networks (HGNNs) (Feng et al., 2019) generalize graph convolution operations to hypergraph structures, allowing for the propagation of information along hyperedges. Several works have explored

HGNNs for tasks such as node classification (Yu et al., 2012; Ma et al., 2021), regression (Di et al., 2021, 2022), link prediction (Yadati et al., 2020; Li et al., 2013; Fan et al., 2021), matching (Liao et al., 2021), 3D retrieval (Gao et al., 2011; Feng et al., 2023), and clustering (Purkait et al., 2016; Li & Milenkovic, 2017). For 3D data, hypergraph learning offers a promising direction for modeling the higher-order relationships inherent (Gao et al., 2012; Zhang et al., 2020; Nong et al., 2022; Jiang et al., 2022). By representing objects and their relationships as hypergraphs, it is capable of capturing complex interactions and generate coherent 3D scenes. In this work, we further investigate the integration of hypergraph representations with 3D generation techniques, which remains an unexplored area of research.

3 Method

Our objective is the creation of 3D content that boasts both precise geometry and rich detailing. To achieve this, our approach, 3D Gaussian Generation via Hypergraph (Hyper-3DG), as illustrated in Fig. 3, leverages the versatility of 3D Gaussian (Kerbl et al., 2023) as a representational form. This allows for the integration of geometric priors and the depiction of intricate high-frequency details. Our method consists of two primary stages, namely "Mainflow: 3D Gaussian Generation via Hypergraph" and "Geometry and Texture Hypergraph Refiner (HGRefiner)". Specifically, the pseudo codes are depicted in Algorithm 1 and Algorithm 2, respectively.

3.1 Mainflow: 3D Gaussian Generation via Hypergraph

In this section, we elaborate on the main-flow of our method. In the initial phase, named as "Warm-up" process in Fig. 3, the objective is to employ a frozen pre-trained 3D generative model and a 2D pre-trained Diffusion model to establish the preliminary geometry and texture of the 3D object from the specified text prompt. The initial establishment of the 3D objects, as described, lays the groundwork for subsequent refinement and enhancement of details. This trunk process is widely embraced as an empirical practice within the field of text-to-3D object generation (*e.g.*, Ma et al. 2023; Sun et al. 2024; Wang et al. 2024; Liang et al. 2024).

Beginning with a textual prompt y, we first obtain a rough version of the 3D Gaussian from scratch using a frozen, pre-trained point cloud Generator (*e.g.*, Point-E Nichol et al. 2022, Shap-E Jun and Nichol 2023). We denote this initialized 3D Gaussian Splatting (3DGS) as $\theta_0 = \{\mu, \alpha, \Sigma, c\}$, where μ, α, Σ, c respectively represent the mean (*i.e.*, center position (x, y, z)), opacity, covariance, and view-dependent color of each corresponding 3D Gaussian distribution.

Algorithm 1: Mainflow: 3D Gaussian Generation via Hypergraph

Input : y, N_0, N_1, CM
Output : $\boldsymbol{\theta}$
Initialization: The adjacent time step interval Δt and the step size Δs for predicting the noise trajectory in DDIM inversion.

$\theta_0 \leftarrow$ Frozen-Pre-trained-3D-Generator (y)
// Initialized Coarse 3DGS
for $\theta_i \leftarrow \theta_0$ **to** $i = N_0$ **do**
  // Update the 3DGS θ_i via pre-trained 2D diffusion model
  $\theta_{i+1} \leftarrow$ DDIM-Update(y, θ_i, CM)
$\theta_i \leftarrow \theta_{N_0}$ // Feed the obtained coarse 3DGS θ_{N_0} to HGRefiner
while θ_i *is not converged* **do**
  $\theta_j \leftarrow \theta_i$
  // Keep the same Hypergraph sturcture for N_1 steps
  for $j \leftarrow 0$ **to** $j = N_1$ **do**
    $\widetilde{\theta}_j \leftarrow \text{HGRefiner}(\theta_j, j, N_1)$ // Update θ_j by HGRefiner
    $\widehat{\theta}_j \leftarrow$ DDIM-Update$(y, \widetilde{\theta}_j, \text{CM})$
  $\theta_i \leftarrow \widehat{\theta}_j$
$\boldsymbol{\theta} \leftarrow \theta_i$ // Obtain the final Fine-generated 3D Object
Function DDIM-Update(y, θ_i, CM)**:**
  $X_{\theta_i} \leftarrow \boldsymbol{g}(\theta_i, \text{CM})$ // Render 2D images from the 3DGS
  $t \sim \mathcal{U}(1, \lambda), \lambda \in \mathbf{R}^+, s = t - \Delta t$ // Timestep t and the pre-timestep s
  $X^t_{\theta_i}, X^s_{\theta_i} \leftarrow \text{DDIM}(y, X_{\theta_i}, t, s, \Delta s)$
  // $X^t_{\theta_i}, X^s_{\theta_i}$ denote the noisy latent vector at t, s, respectively
  $\nabla_\theta \mathcal{L}_{\text{ISM}}(\theta_i) \leftarrow \mathcal{L}_{\text{ISM}}(\epsilon_\phi(X^t_{\theta_i}, t, y), \epsilon_\phi(X^s_{\theta_i}, s, \emptyset))$
  // Gradient Descent
  if θ_i *is not converged* **then**
    $\widehat{\theta}_i \leftarrow \nabla_\theta \mathcal{L}_{\text{ISM}}(\theta_i)$ // Update θ_i with loss function $\mathcal{L}_{\text{ISM}}$
  return $\widehat{\theta}_i$

Subsequently, we introduce a module named "DDIM-Update", which employs a pre-trained 2D Diffusion Model (*e.g.*, Denoising Diffusion Implicit Model Ho et al., 2020; Song et al., 2020) to optimize and refine the 3D Gaussian distribution. This development is inspired by and derived from the methodologies outlined in LucidDreamer (Liang et al., 2024). The module DDIM-Update takes the text prompt (y), 3D Gaussian distributions (θ_0) and the camera poses (CM) as inputs to deliver a more reliable and consistent trajectory for the latent state. The initial 2D images $X_{\theta_0} = \{\boldsymbol{x}_0, \boldsymbol{x}_1, \cdots\} \in \mathbb{R}^{\|\text{CM}\|}$ are rendered by $X_0 = \boldsymbol{g}(\theta_0, \text{CM})$ with the rendering function $\boldsymbol{g}(\cdot)$ as well as the random camera poses $\text{CM} = \{cm_0, cm_1, \dots\}$. Then, the DDIM (Song et al., 2021) inversion transforms the 2D images into a sequence of

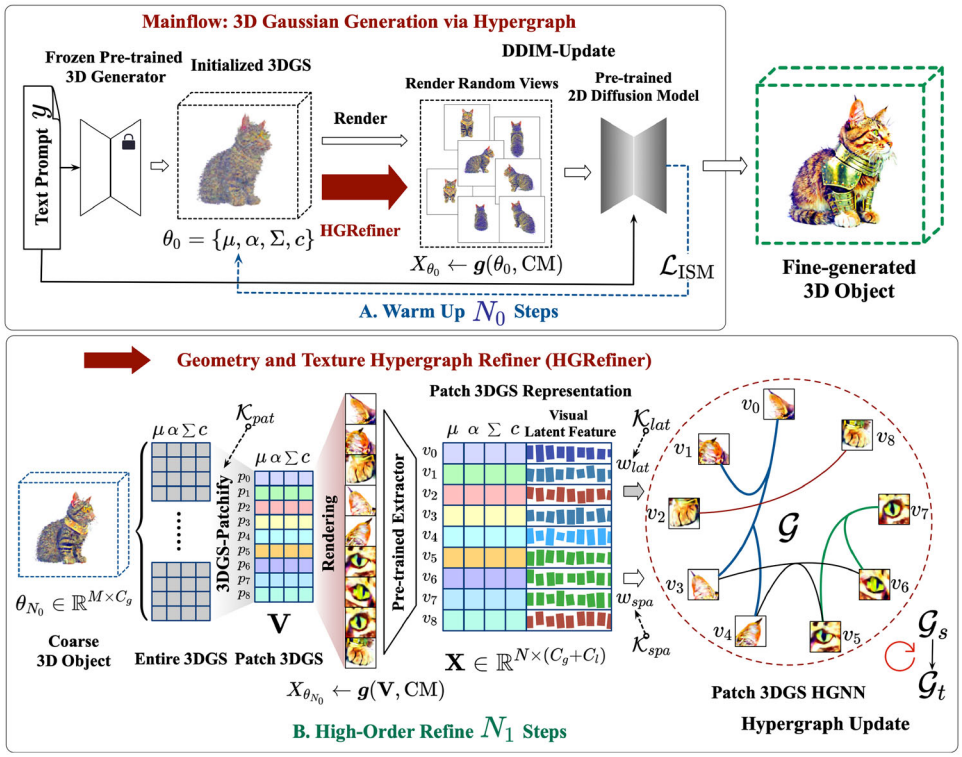

Fig. 3 Illustration of the proposed method, 3D Gaussian Generation via Hypergraph (Hyper-3DG). Our method comprises a main flow as well as a designed hypergraph refiner module (Geometry and Texture Hypergraph Refiner). Given the text prompt as input, the "Warm up" stage can yield coarse 3D Gaussian by a pre-trained 3D generator and a 2D diffusion model. After N_0 steps of initialization, the "HGRefiner" further refines the geometry and texture of the coarse 3D Gaussian at the patch level, with an adjustable updated hypergraph structure. Following N_1 steps of high-order refinement, the final fine-generated 3D Object is obtained

unconditional noisy latent trajectories $\{X_{\theta_i}^{\Delta t}, X_{\theta_i}^{2\Delta t}, \ldots, X_{\theta_i}^{s}, X_{\theta_i}^{t}\}$, $X_{\theta_i}^{s} = X_{\theta_i}^{t-\Delta t}$ is the noise sequence at step s derived from the input X_{θ_i}, where Δt is the DDIM inversion step size and the notation X_{θ_i} for $i \in [0, N_0 - 1]$ represents the $i-th$ update during the warmup phase across N_0 steps, including diverse views rendered from different camera pose cm_j within the range $j \in [0, \|\mathrm{CM}\| - 1]$. Considering $\epsilon_\phi(\cdot)$ as the predicted noise by the 2D diffusion, the iterative prediction can be formulated as:

$$\begin{aligned} &\boldsymbol{x}_t = \sqrt{\bar{\alpha}_t}\bar{\boldsymbol{x}}_0^s + \sqrt{1-\bar{\alpha}_t}\epsilon_\phi(\boldsymbol{x}_s, s, y); \\ &\boldsymbol{x}_t \in X_{\theta_i}^t,\ \boldsymbol{x}_s \in X_{\theta_i}^s;\ y = \emptyset \end{aligned} \tag{1}$$

where $s = t - \Delta t$, $\bar{\boldsymbol{x}}_0^s = \bar{\alpha}_s^{-\frac{1}{2}}\boldsymbol{x}_s - \gamma(s)\epsilon_\phi(\boldsymbol{x}_s, s, y = \emptyset)$, the condition y here is emptyset $\emptyset$, and $\bar{\alpha}_t$ is a function of the diffusion schedule. During the update phase, we leverage the Interval Score Matching (ISM) loss $\mathcal{L}_{\mathrm{ISM}}$ (Liang et al., 2024) to reduce the difference between the denoising directions at two distinct intervals within the diffusion trajectory, which is mathematically expressed as:

$$\mathcal{L} \triangleq \mathbb{E}_{t,c}\left[\omega(t)||\epsilon_\phi(\boldsymbol{x}_t, t, y) - \epsilon_\phi(\boldsymbol{x}_s, s, \emptyset)||^2\right] \tag{2}$$

$$\nabla_\theta \mathcal{L}_{\mathrm{ISM}}(\theta) = \mathbb{E}_{t,c}$$

$$\left[\omega(t)(\underbrace{\epsilon_\phi(\boldsymbol{x}_t, t, y) - \epsilon_\phi(\boldsymbol{x}_s, s, \emptyset)}_{\text{ISM opt direction}})\frac{\partial \boldsymbol{g}(\theta, c)}{\partial \theta}\right] \tag{3}$$

Specifically in practice, we adopt the approach of enhancing efficiency by forecasting $X_{\theta_i}^s$ with large step size Δs in the multi-step DDIM denoising process. The ISM loss guides the optimization of the 3D model's parameters θ to produce detailed and realistic 3D objects, effectively overcoming the over-smoothing problem associated with traditional SDS methods.

Upon completion of N_0 iterations, the "Warm Up" phase yields the parameter set θ_{N_0}, which is subsequently passed to the HGRefiner stage for additional optimization.

3.2 Geometry and Texture Hypergraph Refiner

During this stage, the "HGRefiner" module takes as its input the coarse 3D Gaussian distribution generated in the previous "Warm Up" stage and improves its quality, specifically the geometry and texture, through the designed "Patch 3DGS Hypergraph Learning". We represent the entire 3D Gaussian of the coarse 3D object as $\theta_{N_0} \in \mathbb{R}^{M \times C_g}$, where M and C_g represent the number of Gaussian distributions and the attributes of the 3DGS (μ, α, Σ, c), respectively. The employment of 3D Gaussian distribution as a method for 3D object representation involves handling extensive data volumes, which complicates the extraction of latent semantic visual representations. To address this challenge, we introduce a mechanism termed "3DGS-Patchify", designed to compress and reduce the 3D Gaussian to patch-level affordable dimensions, in spatial space.

Algorithm 2: Geometry and Texture Hypergraph Refiner (HGRefiner)

Input : $\theta_j \in \mathbb{R}^{M\times C_g}, \mathcal{K}_{pat}, \mathcal{K}_{spa}, \mathcal{K}_{lat}, j, N_1$
Output: $\widetilde{\theta}_j \in \mathbb{R}^{M\times C_g}$

1 $\mathbf{V}_j \leftarrow$ 3DGS-Patchify $\left(\theta_j, \mathcal{K}_{pat}\right)$ // Yield the patch-level 3DGS
2 $X_j \leftarrow \boldsymbol{g}\left(\mathbf{V}_j, \mathrm{CM}\right)$ // Render the 2D images from patch-level 3DGS
3 $\mathbf{F}_j \leftarrow$ 2D-Img-Extractor $\left(X_j\right)$ // Extract the latent visual feature
4 $\overline{\mathbf{V}}_j \leftarrow$ Mean-in-Patch $(\mathbf{V})$ // Mean vector of each patch-level 3DGS
5 $\mathbf{X}_j \leftarrow \overline{\mathbf{V}}_j || \mathbf{F}_j$ // Concatenate explicit and latent representation
6 **if** $j == N_1$ **then**
// Update the structure of Patch-3DGS Hypergraph
7 $\mathcal{G}_j^{spa} = \langle \mathbf{X}_j, \mathbf{H}_j^{spa}, \mathbf{W}\rangle \leftarrow$ Construct-Patch-3DGS-Hypergraph $\left(\mathbf{X}_j, \mathcal{K}_{spa}\right)$
8 $\mathcal{G}_j^{lat} = \langle \mathbf{X}_j, \mathbf{H}_j^{lat}, \mathbf{W}\rangle \leftarrow$ Construct-Patch-3DGS-Hypergraph $\left(\mathbf{X}_j, \mathcal{K}_{lat}\right)$
9 **else**
// Not update the structure of Patch-3DGS Hypergraph
10 $\mathcal{G}_j^{spa} \leftarrow \mathcal{G}_{j-1}^{spa}$ // Get the previous hypergraph from last step
11 $\mathcal{G}_j^{lat} \leftarrow \mathcal{G}_{j-1}^{lat}$ // Get the previous hypergraph from last step
12 $\widetilde{\mathbf{X}}_j \leftarrow$ Patch-3DGS-HGNN $\left(\mathbf{X}_j, \mathcal{G}_j^{spa}, \mathcal{G}_j^{lat}, w_{spa}, w_{lat}\right)$
13 $\Delta\theta_j \leftarrow$ 3DGS-Recover $\left(\widetilde{\mathbf{X}}_j - \mathbf{X}_j\right)$ // Calculate updates, recover shape
14 $\widetilde{\theta}_j \leftarrow \theta_j + \Delta\theta_j$ // Obtain the Optimized entire 3DGS

In this context, we employ the K-Means clustering algorithm (Krishna & Murty, 1999) as the implementation mechanism for 3DGS-Patchify. This approach segments the entire 3DGS ($\mathbb{R}^{M\times C_g}$) into N clusters, represented as $\mathbf{V} \in \mathbb{R}^{\left(N\cdot\frac{M}{N}\right)\times C_g}$, where $N \ll M$ signifies the number of clustered patch-level 3DGS. Note that the scale of patchify number N is governed by a hyper-parameter $\mathcal{K}_{pat}$ (where $N \Leftarrow \mathcal{K}_{pat}$ in K-Means). Each patch-level 3DGS represents a small cluster of 3D Gaussian and can be rendered into a patch-level 2D image by the render function ($\boldsymbol{g}(\cdot)$) using specified camera parameters CM. We denote these patch-level 2D images as $X_{\theta_{N_0}} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_N\} \in \mathbb{R}^N$.

Upon acquiring the patch-level 2D images ($X_{\theta_{N_0}}$) and their corresponding patch-level 3DGS, we subsequently treat them as vertices and construct the Patch 3DGS Hypergraph. The tensor representation of these vertices ($\mathbf{X} = \overline{\mathbf{V}}||\mathbf{F}$) is obtained by concatenating the mean vector of each explicit attribute of the patch-level 3DGS ($\overline{\mathbf{V}} \in \mathbb{R}^{N\times C_g}$) with the latent visual features ($\mathbf{F} \in \mathbb{R}^{N\times C_l}$) derived from a pre-trained 2D image feature extractor (*e.g.*, ResNet He et al. 2016, ResNeXt Xie et al. 2017, ViT Dosovitskiy et al. 2021, Swin-T Liu et al. 2021). We denote this tensor representation as $\mathbf{X} \in \mathbb{R}^{N\times(C_g+C_l)}$, where C_g and C_l denote the dimension of 3DGS attributes and the latent visual features, respectively. Considering the representation contains the explicit spatial information and the latent semantic features, we consequently construct the spatial hypergraphs ($\mathcal{G}^{spa}$) as well as semantic hypergraphs ($\mathcal{G}^{lat}$) with corresponding dynamic weights w_{spa} and w_{lat}. The construction of the hypergraph can be implemented as the K-nearest neighbors (KNN) algorithm (Peterson, 2009), applied separately ($\mathcal{K}_{spa}, \mathcal{K}_{lat}$) in both spatial ($\mu \Leftrightarrow (x, y, z)$) and latent spaces ($\mathcal{F}$), through calculating the Euclidean distance. In this way, the spatial and latent Patch 3DGS hypergraphs are constructed and respectively denoted as $\mathcal{G}^{spa} = \langle \mathbf{X}, \mathbf{H}^{spa}, \mathbf{W}\rangle$ and $\mathcal{G}^{lat} = \langle \mathbf{X}, \mathbf{H}^{lat}, \mathbf{W}\rangle$. We denote $\mathbf{H}^{(\cdot)} \in \mathbb{R}^{N\times E}$ and $\mathbf{W} = \mathbf{1} \in \mathbb{R}^{E\times E}$ respectively as the incidence matrix and the vertices weight matrix (*e.g.*, all-ones matrix $\mathbf{1}$), where E represents the number of hyperedges in the hypergraph. The "Patch-3DGS-HGNN" referred to the line 12 of Algorithm 2 is formulated as follows:

$$\begin{cases} \mathbf{H} = \mathbf{H}^{spa} \| \mathbf{H}^{lat} \\ \widetilde{\mathbf{X}} = \sigma\left(\mathbf{D}_v^{-1/2}\mathbf{H}\mathbf{W}\mathbf{D}_e^{-1}\mathbf{H}^{\top}\mathbf{D}_v^{-1/2}\mathbf{X}\boldsymbol{\Theta}\right) \end{cases} \tag{4}$$

where $\mathbf{D}_e \in \mathbb{R}^{E\times E}$, $\mathbf{D}_v \in \mathbb{R}^{N\times N}$ and $\mathbf{W} \in \mathbb{R}^{E\times E}$ denote the diagonal degree matrix of hyperedges, the degree matrix of vertices and weight matrix of hyperedges, respectively. $\sigma(\cdot)$ denotes the nonlinear activation function (*e.g.*, LeakyReLU$(\cdot)$). $\boldsymbol{\Theta} \in \mathbb{R}^{(C_g+C_l)\times(C_g+C_l)}$ is a diagonal matrix representing the learnable parameters updated by the ISM loss function in the outer loop (*i.e.*, Mainflow). It functions similarly to a multilayer perceptron (MLP) layer. By calculating the difference between the original representation tensor ($\mathbf{X} \in \mathbb{R}^{N\times(C_g+C_l)}$) and the updated representation tensor ($\widetilde{\mathbf{X}} \in \mathbb{R}^{N\times(C_g+C_l)}$), and by employing the "3DGS-Recover" function, we generate the update amounts for patch-level 3DGS ($\Delta\theta \in \mathbb{R}^{M\times C_g}$). The function "3DGS-Recover" simply drops the latent visual features ($\mathbf{F}$) from the representation tensor ($\mathbf{X}, \widetilde{\mathbf{X}}$) and recovers the patch-level 3DGS to the original shape by replicating augmentation ($\left(\bigotimes \mathbb{R}^N \rightarrow \mathbb{R}^M\right)$). The final updated 3DGS is produced by adding the updating increments ($\Delta\theta \in \mathbb{R}^{M\times C_g}$) to the original 3DGS ($\theta \in \mathbb{R}^{M\times C_g}$).

Upon completing N_1 steps of this High-Order Patch-3DGS refinement, coupled with the main flow, and ensuring the entire 3DGS ($\boldsymbol{\theta}$) has converged, the final finely detailed 3D object is generated.

4 Experiments

In this section, we elaborate our experiments conducted to validate the effectiveness of the proposed 3DGHG approach. Specifically, we benchmark 3DGHG against previous state-of-the-art methods in the domain of text-to-3D generation. Moreover, we conduct a series of ablation studies to evaluate the significance of crucial components within our method, encompassing each hyper-parameters, loss functions, pre-trained 2D and 3D models, and other relevant factors. The comprehensive results of these investigations are presented hereafter.

4.1 Comparison Experiment

4.1.1 Comparative Methods and Settings

Comparative experiments are conducted in the field of text-to-3D generation, with 3D Gaussian Splatting (Kerbl et al., 2023) serving as the chosen representation for 3D objects. Additionally, we compare our method with other text-to-3D NeRF-based generation methods for a comprehensive evaluation.

To ensure a fair assessment, we employ identical textual prompts and consistent settings across all methods. Specifically, the Classifier-Free Guidance (CFG) parameter (Ho & Salimans, 2022) is set to 100 for the methods based on SDS (Poole et al., 2023; Chen et al., 2024; Tang et al., 2023), and to 7.5 for the method employing the ISM loss (Liang et al., 2024). Other parameters are adjusted in line with the respective official methodologies (Poole et al., 2023; Liang et al., 2024; Chen et al., 2024; Tang et al., 2023) to ensure apples-to-apples comparisons. In our comparative analysis, we benchmarked our method against a series of state-of-the-art 3DGS-based (*i.e.*, GSGEN Chen et al. 2024, LGM Tang et al. 2024, LucidDreamer Liang et al. 2024) as well as NeRF-based (*i.e.*, DreamFusion Poole et al. 2023, DreamGaussian Tang et al. 2023, MVDream Shi et al. 2023, Magic3D Lin et al. 2023, ProlificDreamer Wang et al. 2024) approaches. For more details on the implementation of the comparison methods, please refer to Sect. 1 in the appendix.

4.1.2 Implementation Details

All experiments are conducted using the stable diffusion model 2.1 for distillation purposes, and to ensure consistency and fairness in comparison, NVIDIA 4090 GPUs were used across all trials. We utilize the official implementation of Point-E (Nichol et al., 2022) to generate coarse 3D assets and integrate them into differentiable 3D representations. After the initialization phase, we proceed to the "Warm-Up" phase and then apply our Hyper-3DG refinement (HGRefiner) stage. The basic experimental setup is as follows: iterations = 4,000, N_0 = 1,500, N_1 = 150, $\mathcal{K}_{pat}$ = 50, $\mathcal{K}_{lat}$ = 13, $\mathcal{K}_{spa}$ = 13, the 3DGS position learning rate is set to 1.6×10^{-6}, and 2D image latent feature extractor is ViT. We primarily tune N_1, $\mathcal{K}pat$, $\mathcal{K}lat$, and $\mathcal{K}spa$ for higher quality results. During each N_1 iteration where our Hyper-3DG method is employed, the GPU memory consumption is approximately 3,570 MiB, and the process takes about 1.2 min to complete. The processes of patchify and hypergraph construction are each completed in less than 10 s. The image rendering and feature extraction for the patchify 3DGS step typically require between 30 to 50 s. Subsequently, the final update of the hypergraph generally takes about 10 s to complete.

In the process of 3DGS patch rendering, before executing K-means clustering, we first save the parameters of the overall 3DGS. This involves binding the attributes and spatial positions of each 3DGS patch to ensure that the true attributes of each patch can be restored after executing K-means. When rendering the patch 3DGS into images, we create eight uniformly spaced horizontal views centered around each Gaussian point to serve as camera parameters. Additionally, we provide eight surrounding views with a 45 degree tilt angle as camera parameters. Specifically, the parameters used to create the camera poses are as follows: the horizontal and vertical angle ranges are set to [-180, 180] degrees and [45, 105] degrees, respectively. The camera-to-target distance ranges from 5.2 to 5.5 units, with a default of 3.5 units. The field of view is set to 0.55, ranging between 0.32 and 0.60.

During the rendering process, we utilize images with the object centered, randomly selecting four views for 3DGS image rendering to extract features from the resulting local images. Figure 4 illustrates an example of an overall 3D Gaussian rendering with the number of patches set to 5, alongside the images obtained from the patchified 3DGS. The visualization shows that during the warmup phase, the bird still exhibits a notable Janus problem, through our subsequent optimization, this issue is significantly reduced. It is noteworthy that the automatic patch cropping shown in Fig. 4 is achieved through Gaussian clustering to obtain patch-level 3DGS. By incorporating previous camera parameters, this process enables multi-view patch-level images, which serve as one of the hypergraph's high-order correlation features in the HGRefiner module, utilizing the DINO extractor to leverage these patch-level image features. Experimental results demonstrate that Gaussian clustering based on distance is capable of representing patch-level 3DGS, serving as the vertices of the hypergraph, and obtaining high-order correlations through hypergraph learning. Our implementation is based on the sparse-to-fine framework (Ma et al., 2023; Liang et al., 2024; Chen et al., 2024), with all parameters configured to match those of compared methods to ensure the efficacy of hypergraph integration.

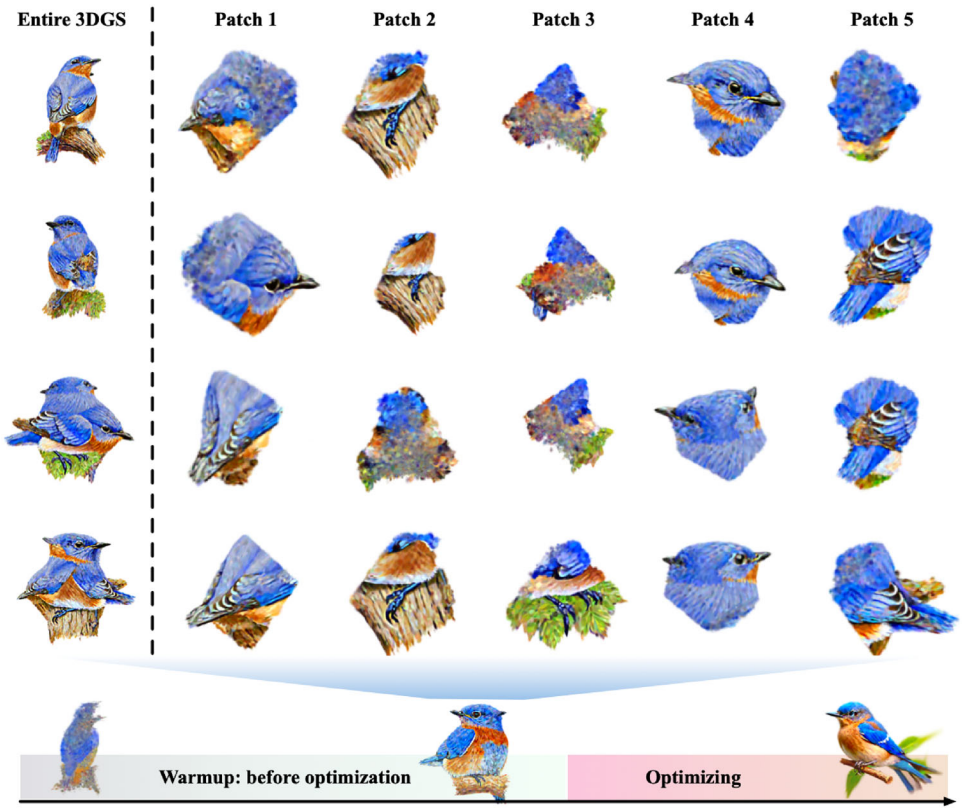

Fig. 4 Illustration of the rendered images from the entire 3DGS and five patchified 3DGS models, generated using four random camera poses. The case we presented exhibit the Janus Problem during the initial warmup phase, as our method continues to optimize, this issue is significantly reduced in the later stages

4.1.3 Quantitative Analysis

We evaluated all comparison methods, including our proposed, using CLIP-score metrics (Radford et al., 2021) for a comprehensive assessment. We compute CLIP-scores across different CLIP retrieval models such as ViT-B/32 and ViT-L/14 (Dosovitskiy et al., 2021). For CLIP-Score evaluation, we utilized a dataset comprising 30 3D assets. For each 3D sample, four uniformly distributed images were extracted from a horizontal frontal view to conduct the CLIP-Score assessment. We also calculated the variance and the *p-value* for each baseline comparison against our method. Additionally, we quantified the time cost of each method and determined the average ranking of each method based on a user study. More details of the user study can be found in Sect. 4.3. The quantitative results are shown in Table 1. Based on these results, we can draw the following key observations:

- Overall, the quantitative results in Table 1 demonstrate that our method, Hyper-3DG, outperforms other state-of-the-art methods across the board in both ViT-B/32 and ViT-B/32 Clip Scores. Moreover, compared to all other methods, the results show that the *p-value* is less than 0.05, demonstrating statistically significant differences for our method.
- Our method introduces an approximate 20% increase in time overhead compared to the best baseline method (*i.e.*, LucidDreamer Liang et al. 2024) due to the appended hypergraph optimization process, raising the average processing time from $\sim$ 90 minutes to $\sim$ 105 minutes. In particular, since the LGM method is not based on optimization and only requires a single inference phase, it can achieve fast inference. However, its generated 3DGS models are very sparse and exhibit holes and distortions in both texture and geometry, as can be observed in the visualization results in the qualitative experiments (refer to Sect. 4.1.4), which is also reflected in the user study (US) score. Although our method does not demonstrate superiority in terms of time overhead, it surpasses other methods in other performance metrics. Enhancing time efficiency will be a focal point of our future research endeavors.
- Due to the discrepancy between the CLIP-Score rankings and the actual visual quality of the generated images, we observe that the 3D generation performance of LGM is not particularly impressive though it achieves a higher ViT-B/32 CLIP-Score than several compared methods (*i.e.*, Magic3D Lin et al. 2023, ProlificDreamer Wang et al. 2024, MVDream Shi et al. 2023, DreamFusion Poole et al. 2023, GSGEN Chen et al. 2024). This discrepancy arises because the CLIP metric is inherently based on 2D evaluations, even though we assess it from multiple perspectives. Consequently, the inherent limitations of CLIP in accurately evaluating 3D content remain a

challenge. Therefore, considering both the CLIP Score and user studies (including visualization and ratings by human voting) provides a rigorous, scientific, and comprehensive evaluation.

4.1.4 Qualitative Analysis

Following the implementation outlined in the previous sections, we present several comparative results, as depicted in Figs. 5, 6, 7 and 8. In these experiments, we maintain consistent parameters with LucidDreamer to ensure a fair comparison. In Fig. 8, we specifically compare two cases from LucidDreamer. It is important to note that, due to the numerous adjustable parameters in LucidDreamer, we can only ensure comparisons under the same experimental settings, making it challenging to precisely replicate the optimal quality shown in the original LucidDreamer presentation. Based on these results, we can derive the following key observations:

- *Enhanced cross-view consistency.* Our method achieves a higher level of view consistency in the generated objects, as demonstrated in the examples of the strawberry and flowers shown in Fig. 5, the frying pan and elephant in Fig. 6, the guitar in Fig. 7, the warrior and saber in Fig. 8. Our method outperforms others by exhibiting greater fidelity when viewed from different angles, particularly in the wedding gown, guitar, frying pan, and warrior examples. Moreover, our approach surpasses other techniques in preserving consistency between the front and back views, effectively addressing the Janus Problem. In contrast, comparative methods such as LucidDreamer (Liang et al., 2024), DreamGaussian (Tang et al., 2023), and LGM (Tang et al., 2024) may result in inconsistencies, broken geometry, and poor texture. Other methods like DreamFusion (Poole et al., 2023), MVDream (Shi et al., 2023), and GSGEN (Chen et al., 2024) struggle to consistently generate the back view, yielding less satisfactory results.
- *Advanced color and texture.* Our method excels in generating 3D assets with highly natural and detailed color and texture. For instance, the authentic texture of the strawberry and white wedding gown in Fig. 5 exhibit a more refined and lifelike appearance, particularly in the smooth and realistic depiction of the strap. The sunflowers and bagel in Fig. 7 showcase more precise structural and textural details, including realistic lighting, environmental ambiance, and pleasing tones. The ice cream sundae and frying pan in Fig. 6 are rendered with more realistic ice cream texture and authentic rust. This stands in contrast to other methods (Poole et al., 2023; Wang et al., 2024; Liang et al., 2024; Chen et al., 2024; Tang et al.,

Table 1 The CLIP-Score results of various methods across different CLIP retrieval models (ViT-B and ViT-L Dosovitskiy et al. 2021), along with the ranking scores from user study (US) and the time cost associated with each method

Methods		ViT-B/32 ↑		ViT-L/14 ↑		US ↓	Time (min) ↓
Magic3D (Lin et al., 2023)	(*std, p-value*)	0.5688	*±0.019,4.2e−18*	0.3787	*± 0.012,7.4e−21*	8.54	50
ProlificDreamer (Wang et al., 2024)	(*std, p-value*)	0.7192	*±0.015,5.4e−8*	0.6042	*±0.012,4.1e−10*	8.43	200
MVDream (Shi et al., 2023)	(*std, p-value*)	0.7785	*±0.008,3.5e−3*	0.6539	*±0.010,2.4e−4*	2.60	40
DreamGaussian (Tang et al., 2023)	(*std, p-value*)	0.7165	*±0.008,5.5e−7*	0.608	*±0.011,4.9e−8*	5.94	40
DreamFusion (Poole et al., 2023)	(*std, p-value*)	0.7602	*±0.011,9.4e−4*	0.588	*±0.015,4.4e−7*	5.98	90
GSGEN (Chen et al., 2024)	(*std, p-value*)	0.7775	*±0.012,3.8e−4*	0.6604	*±0.012,1.5e−3*	6.00	60
LGM (Tang et al., 2024)	(*std, p-value*)	0.7803	*±0.009,1.2e−3*	0.6432	*±0.012,2.4e−5*	2.68	**1**
LucidDreamer (Liang et al., 2024)	(*std, p-value*)	0.7809	*±0.012,9.0e−4*	0.6658	*±0.013,7.3e−3*	2.54	90
Hyper-3DG (ours)	(*std,* –)	**0.7971**	*±0.009,* –	**0.6815**	*±0.010,* –	**2.31**	105

A higher ViT score indicates better performance
For the CLIP-Score, we conducted variance (*std*) and paired *p-value* tests, with $p < 0.05$ indicating a significant difference

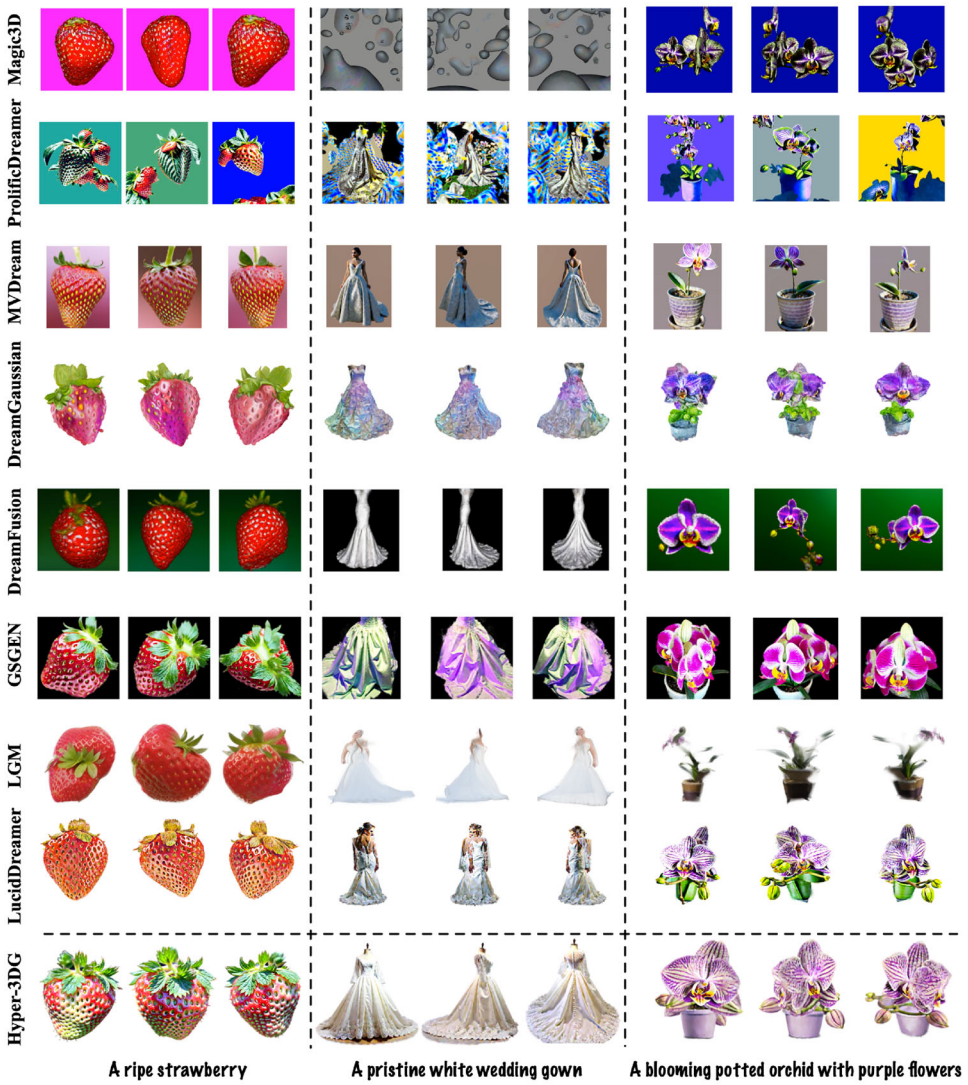

Fig. 5 Three examples of Magic3D (Lin et al., 2023), ProlificDreamer (Wang et al., 2024), MVDream (Shi et al., 2023), DreamGaussian (Tang et al., 2023), DreamFusion (Poole et al., 2023), GSGEN (Chen et al., 2024), LGM (Tang et al., 2024), LucidDreamer (Liang et al., 2024), and Hyper-3DG (Ours). The images in each column represent rendering results from the identical perspective. A few methods could not generate the back view known as the Janus Problem. The results demonstrate the superiority of our approach in synthesizing highly realistic content, replete with intricate details. Please zoom in for the finer intricacies

2023, 2024; Lin et al., 2023), which may suffer from over-smoothing or over-saturation, leading to unrealistic colors or the inability to produce valid colors, as seen in the comparison examples.

- *Improved structural integrity.* Our method successfully addresses the challenge of structural incoherence by effectively filling gaps between correlated structures. This is exemplified in Fig. 7, where our method produces a guitar with superior structural integrity. In comparison, the crank of the guitar generated by the LucidDreamer (Liang et al., 2024) method appears incomplete. This improvement is attributable to the capability of HGRefiner to refine and optimize geometric information, resulting in a complete and coherent structure. Other methods, as demonstrated, may fail to form a normal and complete geometry of the guitar and white wedding gown examples, highlighting the superiority of our approach in maintaining structural integrity.
- *Stylizing observation.* During our experiments, we occasionally observed unnatural white highlights and a cartoon-like style in the generated 3D assets. This phenomenon can be attributed to two main factors. Firstly, the inherent randomness in the stable diffusion distillation process, as well as our experimental parameter settings, contribute to this effect. Through extensive experimentation with various parameters, we found that the results can range from cartoon-like to realistic styles. The example of the strawberry in Fig. 5 serves as a notable illustration of this variability. Secondly, this can also be attributed to the primary 3D data of the pre-trained base model (*i.e.*, Point-E Nichol et al. 2022), which consequently generates 3D objects in the same style.

Fig. 6 A comparison of experimental results among state-of-the-art methods and our approach under identical settings. The images in each column represent rendering results from the same perspective. Some methods fail to generate the back view, a limitation known as the Janus Problem. Please zoom in for the finer intricacies

4.2 Ablation Study

This section employs the foundational parameters outlined in Sect. 4.1.2 as the constants for the ablation study. Constrained by limited manpower and time, we only sample the intermediate rendering state display rather than the final results of the 4,000 iterations. For more ablation studies, please refer to Sect. 1 in the appendix.

4.2.1 Ablation Study on Loss Function

To assess the effectiveness of various loss functions within our proposed framework, we conducted a comparative analysis with consistent settings across all experiments. Several loss functions are commonly used for the task of text-to-3D generation. Focusing on 3D Gaussian Splatting generation, we introduce and compare the experimental performance of three prominent loss functions: Score Distillation Sampling (SDS) (Poole et al., 2023; Chen et al., 2024), Variational Score Distillation (VSD) (Wang et al., 2024), and Interval Score Matching (ISM) (Liang et al., 2024), which have been proposed and widely adopted in state-of-the-art methods.

- **SDS**, introduced in DreamFusion (Poole et al., 2023), also known as Score Jacobian Chaining (Wang et al., 2023), is a popular optimization-based sampling method for 3D asset generation.
- **VSD** (Wang et al., 2024) builds upon SDS by incorporating a variational framework and simulating a Wasserstein gradient flow ODE to generate samples. It offers higher-quality samples but at a greater computational cost.
- **ISM** (Liang et al., 2024) improves upon SDS by replacing the noise term and noise prediction term with noise predictions from DDIM-inversed latents, providing bet-

Fig. 7 A comparison of experimental results among state-of-the-art methods and our approach under identical settings. The images in each column represent rendering results from the same perspective. Some methods fail to generate the back view, a limitation known as the Janus Problem. Please zoom in for the finer intricacies

Fig. 8 Visual comparison of LucidDreamer and our method. We present two cases from LucidDreamer, showing results obtained using identical parameters. Our results demonstrate superior texture details and less of the Janus Problem, as evidenced in cases like the legs of Saber

Fig. 9 Loss Function. The comparative results of employing different loss functions (*i.e.*, SDS Chen et al. 2024, VSD Wang et al. 2024, ISM Liang et al. 2024) within our proposed framework, with identical settings maintained across all experiments

ter sample quality than SDS and lower computation cost than VSD.

The results presented in Fig. 9 indicate that ISM achieves superior texture quality and detail while maintaining computational efficiency. For instance, the lion figure produced using SDS exhibits inaccurate color representation, with the red blood color appearing purple, whereas the colors of samples generated by ISM are more accurate. Additionally, the bus figure produced with ISM displays more detailed and realistic textures compared to those generated by VSD and SDS. Based on this empirical evidence, we recommend adopting the ISM loss function in our framework, as it is likely to yield the best results with a high probability.

4.2.2 Ablation Study on 3DGS-Patchify

To assess the effectiveness of various implementations of "3DGS-Patchify" and to determine the optimal hyper-parameter ($\mathcal{K}_{pat}$), we performed a comparative analysis under consistent experimental conditions. As depicted in Fig. 10, both DBSCAN (Ester et al., 1996) and GMM (Zhuang et al., 1996) yield less satisfactory results compared to K-Means (Hamerly & Elkan, 2003) in the function of "3DGS-Patchify". For instance, DBSCAN and GMM may produce incomplete strawberry shapes due to erroneous clustering outcomes. Similarly, in the cases of the headphone and fire truck examples, these methods do not achieve the level of detail and realism offered by K-Means. We further investigate the impact of the $\mathcal{K}_{pat}$ hyper-parameter of K-Means, by conducting an ablation experiment across a range of values from 1 to 200. Our observations indicate that artifacts resembling tire shapes emerge on the body of car at the extreme values of $\mathcal{K}_{pat}$, specifically when $\mathcal{K}_{pat}$ is set to 1 or 200. Furthermore, a consistent pattern of artifacts appearing in the same locations is noticeable when $\mathcal{K}_{pat}$ is set to 80. These findings suggest that the optimal range for the $\mathcal{K}_{pat}$ parameter may lie within the vicinity of 80, as both lower and higher values can lead to the emergence of unwanted artifacts.

4.2.3 Ablation Study on Hypergraph Construction

To evaluate the effectiveness of various hyperparameters of KNN within our Hyper-3DG framework, the results of the ablation study are presented in Fig. 11. Our approach utilizes two specific KNN parameters: $\mathcal{K}_{lat}$ for the image feature space and $\mathcal{K}_{spa}$ for the 3DGS parameter space. For both KNN parameters, we conducted ablation experiments over a consistent interval with identical variables. With $\mathcal{K}_{pat}$ set at 50, the results in Fig. 11 suggest that the overall image rendering performance, as influenced by the KNN parameters, is comparatively favorable within a middle interval (specifically, 13–33). For $\mathcal{K}_{lat}$, when $\mathcal{K}_{lat} = 23$, the representation of rear tire is superior, while the front tire is average. Conversely, when $\mathcal{K}_{lat} = 43$, the representation of the rear tire is markedly deficient, whereas that of the front tire surpasses the rest. This provides guidance for parameter tuning, suggesting that a balanced representation of the tires might be achieved by targeting parameters between these two extremes. For $\mathcal{K}_{spa}$, the representation of front tire is relatively consistent across different values of $\mathcal{K}_{spa}$, but the representation of rear tire shows a clear preference for middle values (*e.g.*, 23) and a degradation at both extremes (*e.g.*, 3 and 43). The rendered image quality improves with an increase in $\mathcal{K}_{spa}$, although there is a slight decline at higher values, such as $\mathcal{K}_{spa} = 43$. These findings indicate that the optimal values for $\mathcal{K}_{lat}$ and $\mathcal{K}_{spa}$ lie within specific ranges, and they provide a basis for further refinement of the hyper-parameters to enhance the quality of the generated 3D assets.

4.2.4 Ablation Study on Steps of Warm Up and High-Order Refine

In our proposed framework Hyper-3DG, there are two distinct stages: "Mainflow" and "High-Order Refine". Each of these stages is governed by two control parameters, N_0 and N_1, which we have investigated experimentally to understand their respective impacts. As depicted in Fig. 12, the impact of the initial warmup phase was investigated by varying N_0 from 0 to 1,000. Notably, $N_0 = 0$ indicates no warmup phase,

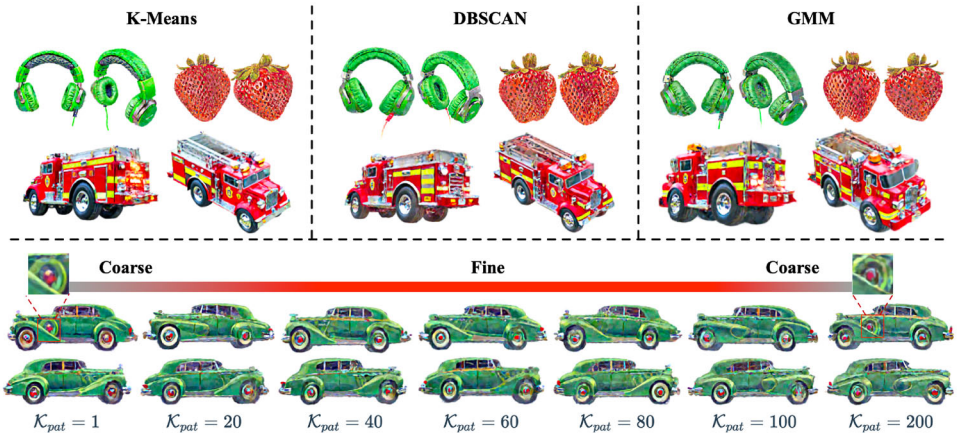

Fig. 10 3DGS-Patchify. The comparative results of employing different 3DGS-Patchify functions (*i.e.*, K-Means Hamerly and Elkan 2003, DBSCAN Ester et al. 1996, GMM Zhuang et al. 1996) and the different hyper-parameter of K-Means (denoted as $\mathcal{K}_{pat}$) within our proposed framework, with identical settings maintained across all experiments. Here, the prompts are respectively "A pair of green headphones", "A ripe strawberry", "A classic fire truck", and "A classic Packard car" (Color figure online)

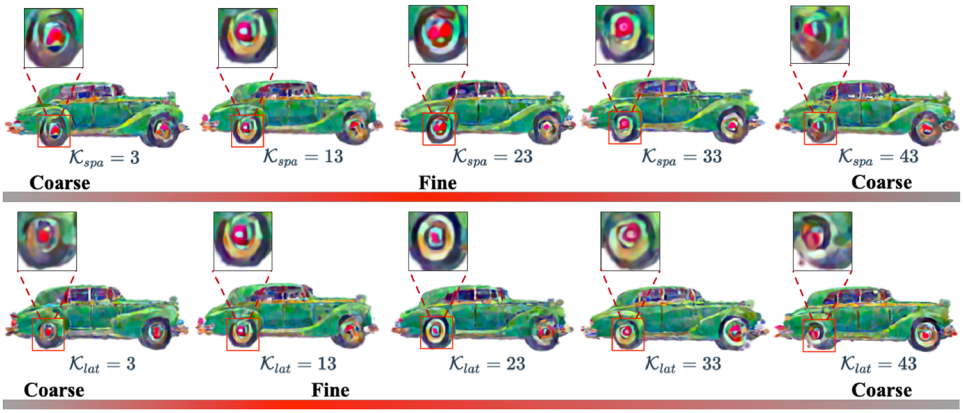

Fig. 11 Hypergraph Construction. The comparative results of employing different hyper-parameters of constructing hypergraphs (*i.e.*, $\mathcal{K}_{spa}$ and $\mathcal{K}_{lat}$) within our proposed framework, with identical settings maintained across all experiments

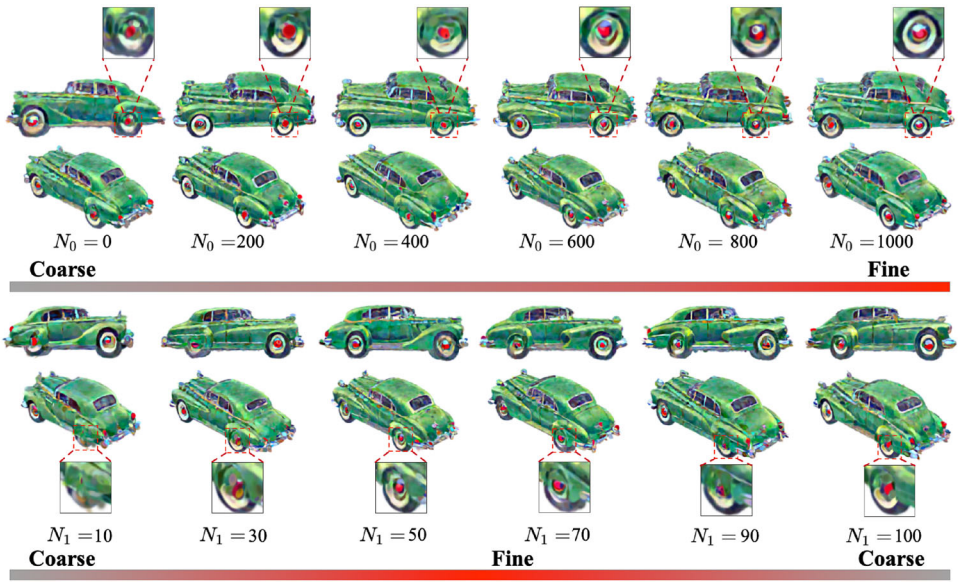

Fig. 12 Steps of Warm Up and Refine. The comparative results of employing different hyper-parameters of "Warm Up" and "Refine" (*i.e.*, N_0 and N_1) within our proposed framework, with identical settings maintained across all experiments

with the generation process starting directly from the initialization provided by Point-E. The quality of the generated Packard car improved incrementally with the increase in N_0. This suggests that a longer warmup phase leads to better

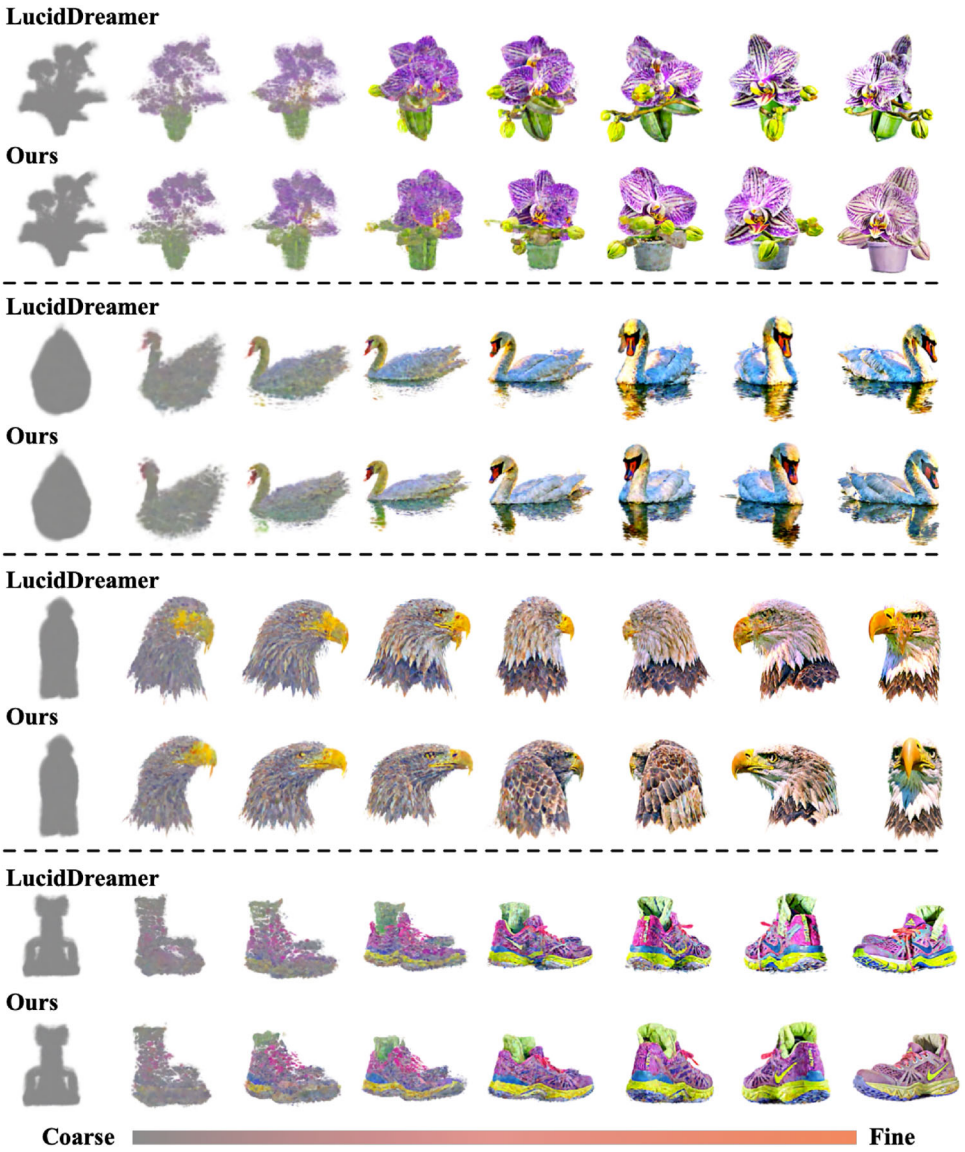

Fig. 13 Illustration of the Coarse-to-Fine process in the Hypergraph Refiner. The results indicate the progression and differences from initialization to the final outcomes. It is evident that LucidDreamer gradually exhibits the Janus Problem during the intermediate stages. In contrast, our approach effectively mitigates this issue by maintaining consistency in geometry and texture tone, resulting in higher quality 3D assets

initialization and preparation for the subsequent refinement steps. In the refinement phase following the warmup, we noticed that the quality of the output continued to improve as N_1 increased from 10 to 70. However, beyond a certain point, *i.e.*, N_1 exceeding 70 and increasing to 100, the quality began to decline. This indicates that there is an optimal range for N_1 that balances the refinement process without overfitting or introducing artifacts. These findings highlight the importance of carefully selecting these hyper-parameters to achieve the best balance between computational efficiency and the quality of the generated 3D assets.

4.2.5 Ablation Study on Effectiveness of Hypergraph Refiner

To demonstrate the effectiveness of the optimization of the proposed hypergraph refiner, we conducted visualization experiments. As shown in Fig. 13, we visualize each step of the optimization process for one state-of-the-art method (*i.e.*, LucidDreamer Liang et al. 2024) and our method. Overall, the visualization results demonstrate the improvements achieved under the same configuration parameters. These examples in Fig. 13 illustrate that the HGRefiner module effectively mitigates issues such as geometry inconsistencies and color oversaturation in 3D generation results. This substantiates the effectiveness and utility of our method.

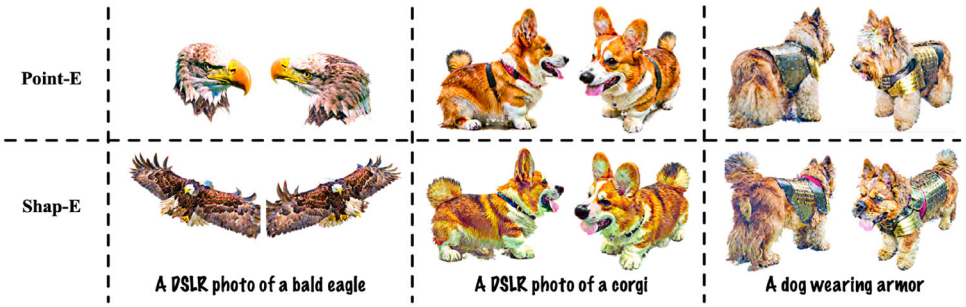

Fig. 14 Pre-trained 3D Generator. The comparative results of employing different pre-trained 3D generator models (*i.e.*, Point-E Nichol et al. 2022 and Shap-E Jun and Nichol 2023) within our proposed framework, with identical settings maintained across all experiments

Notably, the initialization method does not affect our module's ability to process high-order information. The Janus Problem typically emerges during the early to mid-stages of the original optimization process. Our approach, by leveraging high-order correlations, further promotes or inhibits the optimization of 3DGS parameters at the patch level, effectively preventing such unreasonable changes and resulting in higher-quality 3D assets. From the experiments we provided, which emphasize realism and consistency, it is evident that hypergraphs have a positive influence.

4.2.6 Ablation Study on Pre-trained 3D Generator

We conduct an empirical comparison to assess the impact of different pre-trained 3D generator models used for initialization, recognizing the sensitivity of 3D Gaussians to initial conditions. The models we compared are as follows:

- **Point-E** (Nichol et al., 2022) is a diffusion model tailored for rapid point cloud generation. It features a transformer architecture and is capable of generating point clouds in response to text or image inputs.
- **Shap-E** (Jun & Nichol, 2023) is designed to generate parameters for implicit 3D representations, such as NeRF (Mildenhall et al., 2020) or DMTet (Shen et al., 2021). By modeling a higher-dimensional multi-representation output space, Shap-E can quickly produce high-quality 3D assets.

As depicted in Fig. 14, the bald eagle, corgi, and a dog wearing armor, all of which indicate that samples initialized with Point-E (Nichol et al., 2022) are more aligned with the prompt and display fewer rough surface textures compared to those initialized using Shap-E (Jun & Nichol, 2023). This finding suggests that Point-E (Nichol et al., 2022) offers a more stable and precise starting point for the text-to-3D Gaussian Splatting generation process, contributing to superior overall results.

4.3 Details of User Study

We provide additional details of the settings for the user study experiment in this section. In the absence of standardized evaluation metrics for 3D generation, we conducted a user-centric assessment to gauge model performance. A dedicated evaluation set was constructed, encompassing 22 prompts across 9 distinct methodologies. Participants were presented with a rendered video of a particular 3D asset alongside its corresponding input text prompt. 20 participants independently assessed each item in the set. Evaluations focused on the asset's alignment with the prompt and the quality of the generated details, employing a ranking system ranging from 1 to 9. The average ranking scores of Magic3D (Lin et al., 2023), ProlificDreamer (Wang et al., 2024), MVDream (Shi et al., 2023), DreamGaussian (Tang et al., 2023), DreamFusion (Poole et al., 2023), GSGEN (Chen et al., 2024), LGM (Tang et al., 2024), LucidDreamer (Liang et al., 2024), and our proposed method Hyper-3DG are *8.54, 8.43, 2.60, 5.94, 5.98, 6.00, 2.68, 2.54, 2.31*, respectively, as depicted in Table 1. These results highlight the marked superiority of our proposed method. Further details about the user study are as follows:

Firstly, regarding the selection criteria for examples, we adopted a random selection method and chose a total of 22 random samples. This approach ensures the diversity and representativeness of the examples, thereby providing a more comprehensive evaluation of our model's performance. Secondly, concerning the statistical distribution of participants. We conducted interviews through randomly distributed questionnaires with 10 young men and 10 young women, asking them to rank the quality of 3D assets generated from 22 randomly selected prompts. Their experience in the AIGC field ranged from less than 1 year, 1–3 years, to more than 3 years. The user study involved participants ranking each method (lower ranks are better). Each participant's average score across the 22 samples is presented in Table 2, and the final averaged results are summarized in Table 1. We can observe from the user study results that our method outperforms other state-of-the-art methods according to most participants.

Table 2 The average ranking results from a user study conducted on 22 samples, using the exact same settings and text prompts for all compared methods and ours

User	#. 1	#. 2	#. 3	#. 4	#. 5	#. 6	#. 7	#. 8	#. 9	#. 10
Magic3D (Lin et al., 2023)	8.64	8.50	8.55	8.45	8.32	8.73	8.50	8.45	8.55	8.59
ProlificDreamer (Wang et al., 2024)	8.18	8.50	8.45	8.55	8.68	8.27	8.50	8.55	8.45	8.41
MVDream (Shi et al., 2023)	3.64	2.41	2.86	**2.27**	**2.36**	2.82	2.59	2.50	2.68	2.45
DreamGaussian (Tang et al., 2023)	4.82	6.14	5.86	6.09	5.77	5.91	5.77	6.45	6.00	6.05
DreamFusion (Poole et al., 2023)	5.73	5.64	6.14	5.91	6.23	6.27	6.23	5.91	5.82	5.68
GSGEN (Chen et al., 2024)	5.64	6.23	6.00	6.00	6.00	5.82	6.00	5.64	6.18	6.27
LGM (Tang et al., 2024)	3.59	2.73	2.50	2.59	2.64	2.59	2.55	2.50	2.41	3.23
LucidDreamer (Liang et al., 2024)	3.00	2.68	2.35	2.50	2.55	2.36	2.50	2.82	2.64	2.18
Hyper-3DG (Ours)	**1.77**	**2.18**	**2.32**	2.64	2.45	**2.23**	**2.36**	**2.18**	**2.27**	**2.14**
User	#. 11	#. 12	#. 13	#. 14	#. 15	#. 16	#. 17	#. 18	#. 19	#. 20
Magic3D (Lin et al., 2023)	8.50	8.55	8.45	8.50	8.73	8.41	8.59	8.59	8.68	8.50
ProlificDreamer (Wang et al., 2024)	8.50	8.45	8.41	8.41	8.27	8.59	8.41	8.41	8.18	8.45
MVDream (Shi et al., 2023)	2.95	**2.23**	2.45	2.68	2.55	**2.41**	2.64	2.68	**2.32**	2.45
DreamGaussian (Tang et al., 2023)	5.91	5.82	5.64	6.14	6.32	5.95	6.05	5.86	5.95	6.27
DreamFusion (Poole et al., 2023)	6.23	6.00	6.41	5.68	5.68	6.09	6.27	5.91	5.91	5.91
GSGEN (Chen et al., 2024)	5.86	6.18	6.05	6.27	6.00	5.95	5.68	6.05	6.18	5.91
LGM (Tang et al., 2024)	2.36	2.77	2.68	2.27	2.73	2.36	2.45	3.09	2.77	2.68
LucidDreamer (Liang et al., 2024)	2.50	2.32	2.55	2.82	2.41	2.64	2.59	**2.14**	2.55	2.82
Hyper-3DG (Ours)	**2.18**	2.68	**2.36**	**2.23**	**2.32**	2.59	**2.32**	2.27	2.45	**2.18**

A lower ranking score from the user study indicates better performance
A lower ranking indicates better performance

4.4 Limitations and Broader Impact

In this section, we explore the limitations and broader implications associated with our proposed Hyper-3DG method. The Hyper-3DG approach may yield less than optimal outcomes when faced with text prompts that contain complex scene descriptions or intricate logical structures. This shortcoming stems from the limited language comprehension abilities of the Point-E (Nichol et al., 2022) and the CLIP text encoder (Radford et al., 2021) integrated within the StableDiffusion framework. Furthermore, although our introduced 3DGS hypergraph refiner, which incorporates 3D priors, mitigates the Janus Problem, it does not entirely obviate the risk of degeneration, particularly when the textual prompt significantly influences the diffusion models. Beyond technical challenges, the content produced by generative models could have negative implications for the labor market. Moreover, like other generative systems, there is a risk that our method could be exploited to generate fraudulent or harmful content, underscoring the need for heightened vigilance and ethical considerations in its application.

5 Conclusion and Future Work

In summary, our research introduces Hyper-3DG, a framework that seamlessly integrates differentiable rendering and text-to-image advancements to efficiently generate high-quality 3D assets. Central to our approach is the Geometry and Texture Hypergraph Refiner (HGRefiner), which effectively overcomes the Janus Problem and the inherent incoherence issue in generation processes. Hyper-3DG can be applied to various differentiable 3D representations, generally enhancing the quality and reducing the time consumption of existing 3D generation methods. This work not only advances the quality and diversity of 3D assets but also sets a precedent for future innovations in 3D modeling, with far-reaching implications for virtual reality and gaming industries. In future work, we will focus on generating more sophisticated 3D objects as well as intricate scenes together by improving the ability to leverage the capabilities of pre-trained 2D and 3D generation models.

Appendix

In this section, we include details of comparative methods and settings, as well as ablation studies on graph vs. hypergraph and random render views, to provide more insights into our method.

Details of Comparative Methods and Settings

In our comparative analysis, we benchmarked our method against a series of state-of-the-art 3DGS-based (*i.e.*, GSGEN Chen et al. 2024, LGM Tang et al. 2024, LucidDreamer Liang et al. 2024) as well as NeRF-based (*i.e.*, DreamFusion Poole et al. 2023, DreamGaussian Tang et al. 2023, MVDream Shi et al. 2023, Magic3D Lin et al. 2023, ProlificDreamer Wang et al. 2024) approaches.

A: 3DGS Compared Methods

1. **GSGEN** (Chen et al., 2024) integrates SDS with 3D Gaussians, leveraging the explicit nature of 3D Gaussians and applying a point cloud diffusion model as global geometric guidance for the generated 3D objects. In these experiments, we followed the base configuration by initializing the 3DGS with Point-E and setting the maximum training steps to 15,000. The Adam optimizer was employed with an epsilon of 1.0×10^{-15} for numerical stability. Learning rates were tailored for various components: mean rates at 0.005 and 3.0×10^{-5} with exponential decay; svec rates at 0.003 and 0.001, also decaying exponentially; and fixed rates for qvec, color, alpha, and background at 0.003, 0.01, 0.003, and 0.003, respectively. Here, svec and qvec represent the scaling and rotation optimization parameters of the 3DGS, respectively.
2. **LGM** (Tang et al., 2024) introduces a framework called the Large Multi-View Gaussian Model, designed for generating high-resolution 3D models from text prompts or single-view images. LGM distinguishes itself through the use of multi-view Gaussian features for efficient 3D representation and employs an asymmetric U-Net as the backbone for high-throughput processing of multi-view images. Unlike optimization-based methods such as SDS, this framework can produce high-fidelity 3D Gaussians in a short time, thereby overcoming the computational limitations faced by existing feed-forward models. In our experiments, we utilized the default text-to-3D model of LGM.
3. **LucidDreamer** (Liang et al., 2024) analyzes the characteristics of the SDS loss and introduces the Interval Score Matching (ISM) loss to address the excessive smoothness and lack of consistency in the original SDS loss. This innovation significantly enhances the quality of the produced 3D Gaussians. In the LucidDreamer experiments, we initialize the 3DGS using Point-E with an initial point count of 100,000. For the 3D Gaussian spheres, we execute the point cloud adjustment algorithm from iteration 100 to 3,000, resetting the opacity every 300 iterations throughout a total of 5,000 iterations. The opacity, scaling, and rotation optimization parameters of the 3DGS are set with learning rates of 0.05, 0.005, and 0.001, respectively.

B. NeRF Compared Methods

1. **DreamFusion** (Poole et al., 2023) introduces an innovative method for text-to-3D synthesis by leveraging a 2D diffusion model. Central to this approach is Score Distillation Sampling (SDS), which creatively employs a pre-trained 2D text-to-image diffusion model, Imagen, to optimize a 3D model represented as a NeRF. In the DreamFusion experiments, we adhered to the default settings, utilizing an Adam optimizer with a learning rate of 0.01 and betas set to 0.9 and 0.99 for stable convergence. We tailored the learning rates for different model components: 0.01 for geometry and 0.001 for background, to finely tune the training process. The training process employed mixed precision (16-bit) for efficient computation.
2. **DreamGaussian** (Tang et al., 2023) employs a three-stage optimization process from coarse to fine detail. Initially, SDS and 3D Gaussians are used to quickly generate a coarse 3D representation. This is followed by the extraction of meshes, which are then used for UV map refinement in the final stage. In our experiments, we configured the learning rates specifically for different model components to optimize performance: the geometry component was assigned a learning rate of 0.0001, while the texture component was set to a higher learning rate of 0.2.
3. **MVDream** (Shi et al., 2023) is a diffusion model designed for consistent multi-view image generation from text prompts. It combines learning from both 2D and 3D data, achieving the generalizability of 2D diffusion models and the consistency of 3D renderings. The model acts as a generalizable 3D prior, independent of specific 3D representations, and enhances 3D generation tasks through Score Distillation Sampling (SDS), thereby improving the consistency and stability of existing 2D lifting methods. In our experiments, we utilized the default text-to-3D configuration of MVDream.
4. **Magic3D** (Lin et al., 2023) advances beyond DreamFusion by employing a two-stage optimization approach. Initially, it creates a low-resolution NeRF model using sparse 3D hash grid structures for efficiency. Then, it refines this model into a high-resolution 3D mesh using a differentiable renderer and a high-resolution latent dif-

Fig. 15 Graph vs. Hypergraph. The comparative results of employing Graph Neural Network (GNN) (Kipf & Welling, 2017) and our proposed hypergraph-based model model within our proposed framework, with identical settings maintained across all experiments

fusion model. This coarse-to-fine strategy accelerates the optimization process and enables the generation of detailed textures and geometries at resolutions up to 512×512.

5. **ProlificDreamer** (Wang et al., 2024) represents a significant advancement in text-to-3D generation by introducing the Variational Score Distillation (VSD) loss, which models the 3D scene as a probabilistic entity rather than a single point as in traditional Score Distillation Sampling (SDS). This approach allows for a more nuanced and diverse generation of 3D content aligned with textual prompts. Key innovations include a high rendering resolution of 512×512, an annealed distillation time schedule, and a sophisticated scene initialization strategy, all contributing to high-fidelity, detailed, and diverse 3D models. Additionally, ProlificDreamer is more adaptable with classifier-free guidance weights.

Additional Ablation Study

Ablation Study on Graph vs. Hypergraph

We extended our comparative analysis to include graph neural networks (GNNs) (Kipf & Welling, 2017; Veličković et al., 2018) alongside our proposed hypergraph-based methods. In these experiments, we replaced the hypergraph convolution layer with a standard graph convolution layer (GCN) while maintaining all other settings constant. The key difference between these two approaches lies in their ability to model relationships: the graph convolution can only capture pairwise interactions due to its inherent data structure, whereas the hypergraph convolution is capable of modeling high-order correlations among the various parts of a 3D object. This capability is particularly advantageous for 3D data and has been widely supported by previous research in the field (Gao et al., 2022, 2012). As depicted in Fig. 15, the superiority of hypergraph-based methods is evident, such as the coherence of the flying superman's cape, the continuity of the superman's body, the symmetry between the face and body of the panda, and the overall coherence of the packard car. These visual cues indicate that the hypergraph-based methods are more effective in processing and generating 3D data. This result is consistent with the theoretical advantages of hypergraphs in capturing high-order correlations and complex relationships within 3D data, which is a critical aspect for achieving more realistic and detailed 3D object representations.

Ablation Study on 2D Images Visual Feature Extractor

Our ablation experiments aimed to assess the performance of various visual feature extractors for the rendered patches. We examined a range of models, each with its distinct characteristics and strengths in image processing.

- **ResNet** (He et al., 2016) is a foundational deep residual network known for its effectiveness in addressing the vanishing gradient problem and stabilizing training, making it a standard in computer vision.
- **ResNext** (Xie et al., 2017) builds upon ResNet by introducing a multi-branch architecture, which typically enhances performance in certain computer vision tasks compared to ResNet.
- **ViT** (Dosovitskiy et al., 2021) is a transformer-based architecture designed for image recognition tasks. It differs from traditional CNNs by using attention mechanisms to capture global dependencies in image data, often yielding superior results.
- **SwinT** (Liu et al., 2021) adapts the transformer approach to computer vision with a sliding-window scheme, extending the transformer architecture to general image recognition tasks and outperforming the original ViT in many cases.
- **DINO** (Caron et al., 2021) is a self-supervised learning approach that leverages the ViT architecture to capture the visual semantics of images effectively, without the need for large-scale labeled datasets.

Our empirical analysis reveal that different feature extractors produce samples of varying qualities. Samples generated with DINO (Caron et al., 2021) were generally more detailed in terms of texture, as observed in the shape and texture

Fig. 16 Pre-trained 2D Images Visual Feature Extractor. The comparative results of employing different pre-trained 2D Images visual feature extractor (*i.e.*, ResNet He et al. 2016, ResNeXt Xie et al. 2017, ViT Dosovitskiy et al. 2021, Swin-T Liu et al. 2021, DINO Caron et al. 2021) within our proposed framework, with identical settings maintained across all experiments. The new prompts utilized here encompass "A sitting panda" and "A basketball"

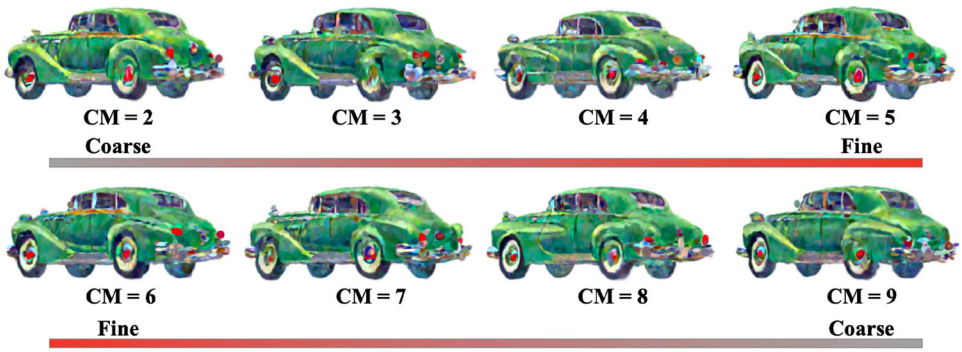

Fig. 17 Random Render Views. The comparative results of employing different hyper-parameters of the random render views (denoted as CM) within our proposed framework, with identical settings maintained across all experiments

of the tire of car and the color of the panda in Fig. 16. However, the differences among the feature extractors were not overwhelmingly significant. Considering the balance between time and computational resources, we typically opt for ResNet or ViT as our implementation method for this part of the framework.

Ablation Study on Random Render Views

The ablation study on Random Render Views explores the effect of varying camera angles on the intermediate rendering process. Using the prompt "a classic Packard car", as shown in Fig. 17, we observe the progression of the generated quality. Initially, at lower Camera Model (CM) values (*e.g.*, 2 to 3), the quality of the Packard car is suboptimal, with notable deficiencies in the clarity of the tires. As the CM values increase to the range of 4 to 7, there is a significant improvement in the generation quality, indicating that the camera angle plays a crucial role in the quality of the rendered 3D object. However, further increasing CM values to 8 to 10 do not lead to an improvement in the quality but rather a decline, suggesting that there is an optimal range for camera angles that maximizes the visual fidelity of the generated 3D assets.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s11263-024-02298-y.

Code Availability The generated results reported in this paper are available in the public repository (https://github.com/yjhboy/Hyper3DG). Additionally, the code used to generate these results will be released in the same repository soon. The repository will be accessible to the

research community, allowing for reproducibility of the experiments and further exploration of the methods presented in this study.

Declarations

Conflict of interest The authors of this research paper declare that there are no conflict of interest regarding the content of this study. None of the authors have any financial, personal, or professional affiliations that could be perceived as having influenced their work on this paper.

Ethical Approval This research study was conducted in accordance with the ethical standards set forth by the relevant institutional review board. All procedures involving human participants were approved by the board, and all participants provided informed consent prior to participating in the study. The authors ensure that all data used in this study are anonymized and treated with confidentiality.

Informed Consent All participants in this study provided informed consent before participating in the research. They were fully informed about the purpose of the study, the procedures involved, and the potential risks and benefits. Participants were also informed about their right to withdraw from the study at any time without consequences. The informed consent forms were signed by the participants and are kept securely by the researchers.

References

Armandpour, M., Zheng, H., Sadeghian, A., Sadeghian, A., & Zhou, M. (2023). Re-imagine the negative prompt algorithm: Transform 2d diffusion into 3d, alleviate janus problem and beyond. arXiv preprint arXiv:2304.04968.

Bai, S., Zhang, F., & Torr, P. H. (2021). Hypergraph convolution and hypergraph attention. *Pattern Recognition, 110*, 107637.

Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., & Joulin, A. (2021). Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 9650–9660).

Chen, Z., Wang, F., Wang, Y., & Liu, H. (2024). Text-to-3d using gaussian splatting. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 21401–21412).

Di, D., Shi, F., Yan, F., Xia, L., Mo, Z., Ding, Z., Shan, F., Song, B., Li, S., Wei, Y., et al. (2021). Hypergraph learning for identification of covid-19 with ct imaging. *Medical Image Analysis, 68*, 101910.

Di, D., Zhang, J., Lei, F., Tian, Q., & Gao, Y. (2022). Big-hypergraph factorization neural network for survival prediction from whole slide image. *IEEE Transactions on Image Processing, 31*, 1149–1160.

Di, D., Zou, C., Feng, Y., Zhou, H., Ji, R., Dai, Q., & Gao, Y. (2022). Generating hypergraph-based high-order representations of whole-slide histopathological images for survival prediction. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 45*(5), 5800–5815.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2021). An image is worth 16x16 words: Transformers for image recognition at scale. https://arxiv.org/abs/2010.11929.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2021). An image is worth 16x16 words: Transformers for image recognition at scale. In *Int. Conf. Learn. Represent*.

Ester, M., Kriegel, H.-P., Sander, J., Xu, X., et al. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *Int. Conf. on Knowl. Discovery and Data Mining.* (vol. 96, pp. 226–231).

Fan, H., Zhang, F., Wei, Y., Li, Z., Zou, C., Gao, Y., & Dai, Q. (2021). Heterogeneous hypergraph variational autoencoder for link prediction. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 44*(8), 4125–4138.

Feng, Y., Ji, S., Liu, Y.-S., Du, S., Dai, Q., & Gao, Y. (2023). Hypergraph-based multi-modal representation for open-set 3d object retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Feng, Y., You, H., Zhang, Z., Ji, R., & Gao, Y. (2019). Hypergraph neural networks. In *Proceedings of the AAAI conference on artificial intelligence,* (vol. 33, pp. 3558–3565).

Fridovich-Keil, S., Yu, A., Tancik, M., Chen, Q., Recht, B., & Kanazawa, A. (2022). Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 5501–5510).

Gao, Y., Feng, Y., Ji, S., & Ji, R. (2022). Hgnn+: General hypergraph neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 45*(3), 3181–3199.

Gao, Y., Tang, J., Hong, R., Yan, S., Dai, Q., Zhang, N., & Chua, T.-S. (2011). Camera constraint-free view-based 3-d object retrieval. *IEEE Transactions on Image Processing, 21*(4), 2269–2281.

Gao, Y., Wang, M., Tao, D., Ji, R., & Dai, Q. (2012). 3-d object retrieval and recognition with hypergraph analysis. *IEEE Transactions on Image Processing, 21*(9), 4290–4303.

Gao, Y., Zhang, Z., Lin, H., Zhao, X., Du, S., & Zou, C. (2020). Hypergraph learning: Methods and practices. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 44*(5), 2548–2566.

Hamerly, G., & Elkan, C. (2003). Learning the k in k-means. *Advances in Neural Information Processing Systems*, *16*.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770–778).

Ho, J., & Salimans, T. (2022). Classifier-free diffusion guidance. arXiv preprint arXiv:2207.12598.

Ho, M.-Y., Wu, C.-M., Wu, M.-S., & Tseng, Y. J. (2024). Every pixel has its moments: Ultra-high-resolution unpaired image-to-image translation via dense normalization. https://arxiv.org/abs/2407.04245.

Ho, J., Jain, A., & Abbeel, P. (2020). Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems, 33*, 6840–6851.

Hong, S., Ahn, D., & Kim, S. (2024). Debiasing scores and prompts of 2d diffusion for view-consistent text-to-3d generation. *Advances in Neural Information Processing Systems*, 36.

Hu, Z., Zhao, M., Zhao, C., Liang, X., Li, L., Zhao, Z., Fan, C., Zhou, X., & Yu, X. (2024). Efficientdreamer: High-fidelity and robust 3d creation via orthogonal-view diffusion priors. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 4949–4958).

Huang, Y., Wang, J., Zeng, A., Cao, H., Qi, X., Shi, Y., Zha, Z. -J., & Zhang, L. (2024). Dreamwaltz: Make a scene with complex 3d animatable avatars. *Advances in Neural Information Processing Systems*, 36.

Huang, B., Yu, Z., Chen, A., Geiger, A., & Gao, S. (2024). 2d gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024 conference papers* (pp. 1–11).

Huang, T., Zeng, Y., Zhang, Z., Xu, W., Xu, H., Xu, S., Lau, R. W., & Zuo, W. (2024). Dreamcontrol: Control-based text-to-3d generation with 3d self-prior. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 5364–5373).

Jiang, P., Deng, X., Wang, L., Chen, Z., & Zhang, S. (2022). Hypergraph representation for detecting 3d objects from noisy point clouds. *IEEE Transactions on Knowledge and Data Engineering*

Jun, H., & Nichol, A. (2023). Shap-e: Generating conditional 3d implicit functions. arXiv preprint arXiv:2305.02463.

Kerbl, B., Kopanas, G., Leimkühler, T., & Drettakis, G. (2023). 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.42*(4).

Kipf, T. N., & Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In *Int. Conf. Learn. Represent*.

Krishna, K., & Murty, M. N. (1999). Genetic k-means algorithm. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 29*(3), 433–439.

Lai, J., Yang, S., Zhou, J., Wu, W., Chen, X., Liu, J., Gao, B.-B., & Wang, C. (2024). Clustered-patch Element Connection for Few-shot Learning. https://arxiv.org/abs/2304.10093.

Li, P., & Milenkovic, O. (2017). Inhomogeneous hypergraph clustering with applications. *Advances in Neural Information Processing Systems,30*.

Li, D., Xu, Z., Li, S., & Sun, X. (2013). Link prediction in social networks based on hypergraph. In *Proceedings of the 22nd international conference on world wide web* (pp. 41–42).

Liang, Y., Yang, X., Lin, J., Li, H., Xu, X., & Chen, Y. (2024). Luciddreamer: Towards high-fidelity text-to-3d generation via interval score matching. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 6517–6526).

Liao, X., Xu, Y., & Ling, H. (2021). Hypergraph neural networks for hypergraph matching. *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 1266–1275).

Lin, C.-H., Gao, J., Tang, L., Takikawa, T., Zeng, X., Huang, X., Kreis, K., Fidler, S., Liu, M.-Y., & 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*.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., & Guo, B. (2021). Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 10012–10022).

Liu, R., Wu, R., Van Hoorick, B., Tokmakov, P., Zakharov, S., & Vondrick, C. (2023). Zero-1-to-3: Zero-shot one image to 3d object. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 9298–9309).

Long, X., Guo, Y.-C., Lin, C., Liu, Y., Dou, Z., Liu, L., Ma, Y., Zhang, S.-H., Habermann, M., Theobalt, C., et al. (2024). Wonder3d: Single image to 3d using cross-domain diffusion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 9970–9980).

Luo, C., Di, D., Ma, Y., Xue, Z., Wei, C., Yang, X., & Liu, Y. (2024). Trame: Trajectory-anchored multi-view editing for text-guided 3d gaussian splatting manipulation. arXiv preprint arXiv:2407.02034.

Ma, B., Deng, H., Zhou, J., Liu, Y. -S., Huang, T., & Wang, X. (2023). Geodream: Disentangling 2d and geometric priors for high-fidelity and consistent 3d generation. arXiv preprint arXiv:2311.17971.

Ma, Z., Jiang, Z., & Zhang, H. (2021). Hyperspectral image classification using feature fusion hypergraph convolution neural network. *IEEE Transactions on Geoscience and Remote Sensing, 60*, 1–14.

Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., & Ng, R. (2020). Nerf: Representing scenes as neural radiance fields for view synthesis. In *Eur. Conf. Comput. Vis.*

Nichol, A., Jun, H., Dhariwal, P., Mishkin, P., & Chen, M. (2022). Point-e: A system for generating 3d point clouds from complex prompts. arXiv preprint arXiv:2212.08751.

Nong, L., Peng, J., Zhang, W., Lin, J., Qiu, H., & Wang, J. (2022). Adaptive multi-hypergraph convolutional networks for 3d object classification. *IEEE Transactions on Multimedia*

Peebles, W., & Xie, S. (2023). Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 4195–4205).

Peterson, L. E. (2009). K-nearest neighbor. *Scholarpedia, 4*(2), 1883.

Poole, B., Jain, A., Barron, J. T., & Mildenhall, B. (2023). Dreamfusion: Text-to-3d using 2d diffusion. In *Int. Conf. Learn. Represent*.

Purkait, P., Chin, T.-J., Sadri, A., & Suter, D. (2016). Clustering with hypergraphs: The case for large hyperedges. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 39*(9), 1697–1711.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., & Clark, J., et al. (2021). Learning transferable visual models from natural language supervision. In *International conference on machine learning* (pp. 8748–8763). PMLR.

Shen, T., Gao, J., Yin, K., Liu, M.-Y., & Fidler, S. (2021). Deep marching tetrahedra: A hybrid representation for high-resolution 3d shape synthesis. *Advances in Neural Information Processing Systems, 34*, 6087–6101.

Shi, Y., Wang, P., Ye, J., Long, M., Li, K., & Yang, X. (2023). Mvdream: Multi-view diffusion for 3d generation. arXiv preprint arXiv:2308.16512.

Song, J., Meng, C., & Ermon, S. (2021). Denoising diffusion implicit models. In *Int. Conf. Learn. Represent*.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., & Poole, B. (2020). Score-based generative modeling through stochastic differential equations. arXiv preprint arXiv:2011.13456.

Sun, J., Zhang, B., Shao, R., Wang, L., Liu, W., Xie, Z., & Liu, Y. (2024). Dreamcraft3d: Hierarchical 3d generation with bootstrapped diffusion prior. In *Int. Conf. Learn. Represent*.

Tang, J., Chen, Z., Chen, X., Wang, T., Zeng, G., & Liu, Z. (2024). Lgm: Large multi-view gaussian model for high-resolution 3d content creation. arXiv preprint arXiv:2402.05054.

Tang, J., Ren, J., Zhou, H., Liu, Z., & Zeng, G. (2023). Dreamgaussian: Generative gaussian splatting for efficient 3d content creation. arXiv preprint arXiv:2309.16653.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., & Bengio, Y. (2018). Graph attention networks. In *Int. Conf. Learn. Represent*.

Wang, H., Du, X., Li, J., Yeh, R. A., & Shakhnarovich, G. (2023). Score jacobian chaining: Lifting pretrained 2d diffusion models for 3d generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 12619–12629).

Wang, P., Liu, L., Liu, Y., Theobalt, C., Komura, T., & Wang, W. (2021). Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *Adv. Neural Inform. Process. Syst.*

Wang, Z., Lu, C., Wang, Y., Bao, F., Li, C., Su, H., & Zhu, J. (2024). Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. *Advances in Neural Information Processing Systems*, 36.

Xie, S., Girshick, R., Dollár, P., Tu, Z., & He, K. (2017). Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1492–1500).

Yadati, N., Nitin, V., Nimishakavi, M., Yadav, P., Louis, A., & Talukdar, P. (2020). Nhp: Neural hypergraph link prediction. In *Proceedings of the 29th ACM international conference on information & knowledge management* (pp. 1705–1714).

Ye, Z., Li, W., Liu, S., Qiao, P., & Dou, Y. (2024). Absgs: Recovering Fine Details for 3d Gaussian Splatting. arXiv:2404.10484.

Yi, T., Fang, J., Wu, G., Xie, L., Zhang, X., Liu, W., Tian, Q., & Wang, X. (2023). Gaussiandreamer: Fast generation from text to 3d gaussian splatting with point cloud priors. arXiv preprint arXiv:2310.08529.

Yu, Z., Chen, A., Huang, B., Sattler, T., & Geiger, A. (2024). Mip-splatting: Alias-free 3d gaussian splatting. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 19447–19456).

Yu, Y., Zhu, S., Qin, H., & Li, H. (2024). BoostDream: Efficient refining for high-quality text-to-3D generation from multi-view diffusion.
Yu, J., Tao, D., & Wang, M. (2012). Adaptive hypergraph learning and its application in image classification. *IEEE Transactions on Image Processing, 21*(7), 3262–3272.
Zhang, B., Cheng, Y., Yang, J., Wang, C., Zhao, F., Tang, Y., Chen, D., & Guo, B. (2024). Gaussiancube: A structured and explicit radiance representation for 3d generative modeling. arXiv.
Zhang, H. (2019). 3d model generation on architectural plan and section training through machine learning. *Technologies, 7*(4), 82.
Zhang, S., Cui, S., & Ding, Z. (2020). Hypergraph spectral analysis and processing in 3d point cloud. *IEEE Transactions on Image Processing, 30*, 1193–1206.
Zhuang, X., Huang, Y., Palaniappan, K., & Zhao, Y. (1996). Gaussian mixture density modeling, decomposition, and applications. *IEEE Transactions on Image Processing, 5*(9), 1293–1302.