

3D-aware Image Synthesis via Learning Structural and Textural Representations

Yinghao Xu¹ Sida Peng² Ceyuan Yang¹ Yujun Shen³ Bolei Zhou¹
¹The Chinese University of Hong Kong ²Zhejiang University ³Bytedance Inc.

{xy119, yc019, bzhou}@ie.cuhk.edu.hk pengsida@zju.edu.cn shenyujun0302@gmail.com

Abstract

Making generative models 3D-aware bridges the 2D image space and the 3D physical world yet remains challenging. Recent attempts equip a Generative Adversarial Network (GAN) with a Neural Radiance Field (NeRF), which maps 3D coordinates to pixel values, as a 3D prior. However, the implicit function in NeRF has a very local receptive field, making the generator hard to become aware of the global structure. Meanwhile, NeRF is built on volume rendering which can be too costly to produce high-resolution results, increasing the optimization difficulty. To alleviate these two problems, we propose a novel framework, termed as **VolumeGAN**, for high-fidelity 3D-aware image synthesis, through explicitly learning a structural representation and a textural representation. We first learn a feature volume to represent the underlying structure, which is then converted to a feature field using a NeRF-like model. The feature field is further accumulated into a 2D feature map as the textural representation, followed by a neural renderer for appearance synthesis. Such a design enables independent control of the shape and the appearance. Extensive experiments on a wide range of datasets show that our approach achieves sufficiently higher image quality and better 3D control than the previous methods. Project page is <https://genforce.github.io/volumegan>.

1. Introduction

Learning 3D-aware image synthesis draws wide attention recently [3, 30, 35]. An emerging solution is to integrate a Neural Radiance Field (NeRF) [28] into a Generative Adversarial Network (GAN) [7]. Specifically, the 2D Convolutional Neural Network (CNN) based generator is replaced with a generative implicit function, which maps the raw 3D coordinates to point-wise densities and colors conditioned on the given latent code. Such an implicit function encodes the structure and the texture of the output image in the 3D space.

However, there are two problems of directly employing

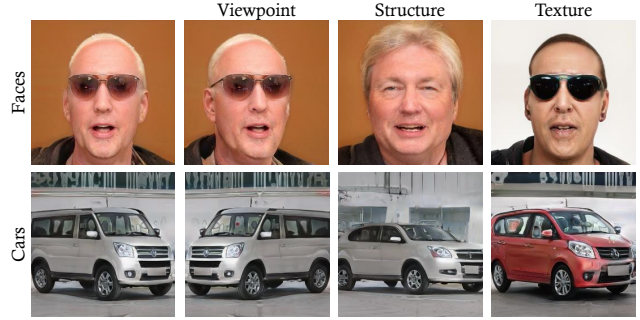


Figure 1. Images of faces and cars synthesized by VolumeGAN, which enables the control of viewpoint, structure, and texture.

NeRF [28] in the generator. On one hand, the implicit function in NeRF produces the color and density for each 3D point using a Multi-Layer Perceptron (MLP) network. With a very local receptive field, it is hard for the MLP to represent the underlying structure globally when synthesizing images. Thus only using the 3D coordinates as the inputs [3, 30, 35] is not expressive enough to guide the generator with the global structure. On the other hand, volume rendering generates the pixel values of the output image separately, which requires sampling numerous points along the camera ray regarding each pixel. The computational cost hence significantly increases when the image size becomes larger. It may cause the insufficient optimization of the model training, and further lead to unsatisfying performance for high-resolution image generation.

Prior work has found that 2D GANs benefits from valid representations learned by the generator [36, 43, 44]. Such generative representations describe a synthesis with *high-level features*. For example, Xu *et al.* [43] confirm that a face synthesis model is aware of the landmark positions of the output face, and Yang *et al.* [44] identify the multi-level variation factors emerging from generating bedroom images. These representative features encode rich texture and structure information, thereby enhancing the synthesis quality [16] and the controllability [36] of image GANs. In contrast, as mentioned above, existing 3D-aware generative models directly render the pixel values from coordinates [3, 35], without learning explicit representations.

In this work, we propose a new generative model, termed as **VolumeGAN**, which achieves 3D-aware image synthesis through *explicitly learning a structural and a textural representation*. Instead of using the 3D coordinates as the inputs, we generate a feature volume using a 3D convolutional network, which encodes the relationship between various spatial regions and hence compensates for the insufficient receptive field caused by the MLP in NeRF. With the feature volume modeling the underlying structure, we query a coordinate descriptor from the feature volume to describe the structural information for each 3D point. We then employ a NeRF-like model to create a feature field, by taking the coordinate descriptor attached with the raw coordinate as the input. The feature field is further accumulated into a 2D feature map as the textural representation, followed by a CNN with 1×1 kernel size to finally render the output image. In this way, we separately model the structure and the texture with the 3D feature volume and the 2D feature map, enabling the disentangled control of the shape and the appearance.

We evaluate our approach on various datasets and demonstrate its superior performance over existing alternatives. In terms of the image quality, VolumeGAN achieves substantially better Fréchet Inception Distance (FID) score [11]. Taking the FFHQ dataset [16] under 256×256 resolution as an instance, we improve the FID from 36.7 to 9.1. We also enable 3D-aware image synthesis on the challenging indoor scene dataset, *i.e.*, LSUN bedroom [46]. Our model also suggests stable control of the object pose and shows better consistency across different viewpoints, benefiting from the learned structural representation (*i.e.*, the feature volume). Furthermore, we conduct a detailed empirical study on the learned structural and textural representations, and analyze the trade-off between the image quality and the 3D property.

2. Related work

Neural Implicit Representations. Recent methods [5, 20, 27, 28, 32, 39] propose to represent 3D scenes with neural implicit functions, such as occupancy field [27], signed distance field [32], and radiance field [28]. To recover these representations from images, they develop differentiable renderers [19, 21, 31, 41] that render implicit functions into images, and optimize the network parameters by minimizing the difference between rendered images and observed images. These methods can reconstruct high-quality 3D shapes and perform photo-realistic view synthesis, but they have several strong assumptions on the input data, including dense camera views, precise camera parameters, and constant lighting effects. More recently, some methods [3, 13, 25, 26, 30, 35] have attempted to reduce the constraints on the input data. By appending an appearance embedding to each input image, [25] can recover

3D scenes from multi-view images with different lighting effects. [13, 26] reconstructs neural radiance fields from very sparse views by applying a discriminator to supervise the synthesized images on novel views. Different from these methods requiring multi-view images, our approach can synthesize high-resolution images by training networks only on unstructured single-view image collections.

Image Synthesis with 2D GANs. Generative Adversarial Networks (GANs) [7, 14] have made significant progress in synthesizing photo-realistic images but lack the ability to control the generation. To obtain better controllability in synthesizing process, [36, 37, 44, 49] investigate the latent space of the pre-trained GANs to determine the semantic direction. Many works [4, 33] add regularizers or modify the network structure [10, 15–17] to improve the disentanglement of variation factors without explicit supervision. Besides, recent methods [1, 9, 43, 50] adopt optimization or train encoders for controlling attributes of real images by pre-trained GANs. However, these efforts control the generation only in 2D space and ignore the 3D nature of the physical world, resulting in a lack of consistency for view synthesis.

3D-Aware Image Synthesis. 2D GANs lack knowledge of 3D structure. Some prior works directly introduce 3D representation to perform 3D-aware image synthesis. VON [51] generates a 3D shape represented by voxels which is then projected into 2D image space by a differentiable renderer. HoloGAN [29] proposes voxelized and implicit 3D representations and then render it to 2D space with a reshape operation. While these methods can achieve good results, the synthesized images suffer from the fine details and identity shift because of the voxel resolution restriction. Instead of using the voxel representation, GRAF [35] and π -GAN [3] propose to model 3D shapes by neural implicit representation, which maps the coordinates to the RGB color. However, due to the computationally intensive rendering process, they cannot synthesize high-resolution images with good visual quality. To overcome this problem, [30] first render low-resolution feature maps with neural feature fields and then generate high-resolution images with 2D CNNs, also with the coordinates as the input. However, severe artifacts across different camera views are introduced because CNN-based decoder harms the 3D consistency. Unlike previous attempts, we leverage the feature volume to provide the feature descriptor for each coordinate and a neural renderer consisting of 1×1 convolution block to synthesize high-quality images with better multi-view consistency and 3D control.

The concurrent work StyleNeRF [8] also adopts 1×1 convolution block to synthesize high-quality images. However, we adopt the feature volume to provide the structural description for the synthesized object instead of using regularizers to improve the 3D properties.

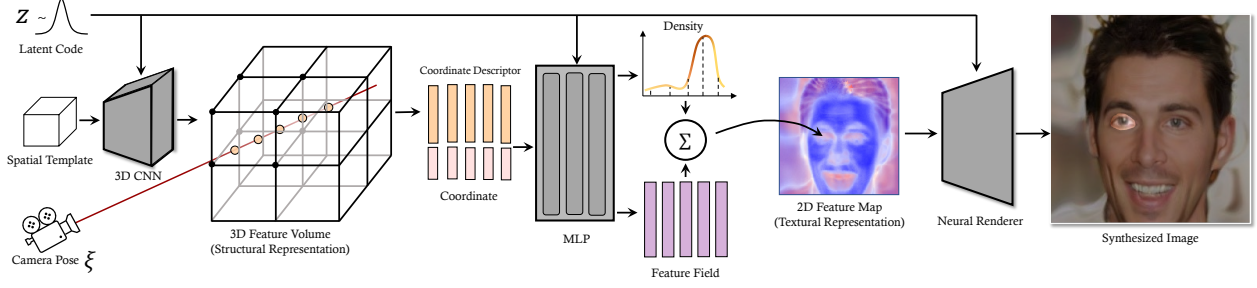


Figure 2. **Framework of the proposed VolumeGAN.** We first learn a feature volume, starting from a learnable spatial template, as the *structural representation*. Given the camera pose ξ , we sample points along a camera ray and query the coordinate descriptor of each point from the feature volume via trilinear interpolation. The resulting coordinate descriptors, concatenated with the raw 3D coordinates, are then converted to a generative feature field and further accumulated as a 2D feature map. Such a feature map is regarded as the *textural representation*, which guides the rendering of the appearance of the output synthesis with the help of a neural renderer.

3. Method

This work targets at learning 3D-aware image generative models from a collection of 2D images. Previous attempts replace the generator of a GAN model with an implicit function [28], which maps 3D coordinates to pixel values. To improve the controllability and synthesis quality, we propose to explicitly learn the structural and the textural representations that are responsible for the underlying structure and texture of the object respectively. Concretely, instead of directly bridging coordinates with densities and RGB colors, we ask the implicit function to transform 3D feature volume (*i.e.*, the structural representation) to a generative feature field, which are then accumulated into a 2D feature map (*i.e.*, the textural representation). The overall framework is illustrated in Fig. 2. Before going into details, we first briefly introduce the Neural Radiance Field (NeRF), which is a core module of the proposed model.

3.1. Preliminary

The neural radiance field [28] is formulated as a continuous function $F(\mathbf{x}, \mathbf{d}) = (\mathbf{c}, \sigma)$, which maps a 3D coordinate $\mathbf{x} \in \mathbb{R}^3$ and the viewing direction $\mathbf{d} \in \mathbb{S}^2$ to the RGB color $\mathbf{c} \in \mathbb{R}^3$ and a volume density $\sigma \in \mathbb{R}$. Then, given a sampled ray, we can predict the colors and densities of all the points that the ray goes through, which are then accumulated into the pixel value with volume rendering techniques. Typically, the function $F(\cdot, \cdot)$ is parameterized with a multi-layer perceptron (MLP), $\Phi(\cdot)$, as the backbone, and two independent heads, $\phi_c(\cdot, \cdot)$ and $\phi_d(\cdot)$, to regress the color and density:

$$\mathbf{c}(\mathbf{x}, \mathbf{d}) = \phi_c(\Phi(\mathbf{x}), \mathbf{d}), \quad (1)$$

$$\sigma(\mathbf{x}) = \phi_d(\Phi(\mathbf{x})), \quad (2)$$

where the color is related with the viewing direction \mathbf{d} due to the variation factors like lighting, while the density σ is independent of \mathbf{d} .

NeRF is primarily proposed for 3D reconstruction and novel view synthesis, which is trained with the supervision from multi-view images. To enable random sampling by learning from a collection of single-view images, recent attempts [3, 35] introduce a latent code \mathbf{z} to the function $F(\cdot, \cdot)$. In this way, the geometry and appearance of the rendered image will vary according to the input \mathbf{z} , resulting in diverse generation. Such a stochastic implicit function is asked to compete with a discriminator of GANs [7] to mimic the distribution of real 2D images. In the learning process, the revised function $F(\mathbf{x}, \mathbf{d}, \mathbf{z}) = (\mathbf{c}, \sigma)$ is supposed to encode the structure and the texture information simultaneously.

3.2. 3D-aware Generator in VolumeGAN

To improve the controllability and image quality of the NeRF-based 3D-aware generative model, we propose to explicitly learn a structural representation and a textural representation, which control the underlying structure and texture, respectively. In this part, we will introduce the design of the structural and the textural representations, as well as their integration through a generative neural feature field.

3D Feature Volume as Structural Representation. As pointed out by NeRF [28], the low-dimensional coordinates \mathbf{x} should be projected into a higher-dimensional feature to describe the complex 3D scenes. For this purpose, a typical solution is to characterize \mathbf{x} into Fourier features [40]. However, such a Fourier transformation cannot introduce additional information beyond the spatial position. It may be enough for reconstructing a fixed scene, but yet far from encoding a distributed feature for the image synthesis of different object instances. Hence, we propose to learn a grid of features providing the inputs of implicit functions, which gives a more detailed description of each spatial point. We term such a 3D feature volume, \mathbf{V} , as the *structural representation* which characterizes the underlying 3D structure.

To obtain the feature volume, we employ a sequence of 3D convolutional layers with the Leaky ReLU (LReLU) functions [24]. Inspired by Karras *et al.* [16], we apply Adaptive Instance Normalization (AdaIN) [12] to the output of each layer to introduce diversity to the feature volume. Starting from a learnable 3D tensor, \mathbf{V}_0 , the structural representation is generated with

$$\mathbf{V} = \psi_{n_s-1} \circ \psi_{n_s-2} \circ \dots \circ \psi_0(\mathbf{V}_0), \quad (3)$$

$$\psi_i(\mathbf{V}_i) = \text{AdaIN}\left(\text{LReLU}(\text{Conv}(\text{Up}(\mathbf{V}_i, s_i))), \mathbf{z}\right), \quad (4)$$

where n_s denotes the number of layers for structure learning. $s_i \in \{1, 2\}$ is the upsampling scale for the i -th layer.

2D Feature Map as Textural Representation. As discussed before, volume rendering can be extremely slow and computationally expensive, making it costly to directly render the raw pixels of a high-resolution image. To mitigate the issue, we propose to learn a feature map at a low resolution, followed by a CNN to render a high-fidelity result. Here, the 2D feature map is responsible for describing the visual appearance of the final output. The tailing CNN consists of several Modulated Convolutional Layers (ModConv) [17], also activated by LReLU. To avoid the CNN from weakening the 3D consistency [30], we use 1×1 kernel size for all layers such that the per-pixel feature can be processed independently. In particular, given a 2D feature map, \mathbf{M} , as the textural representation, the image is generated by

$$\mathbf{I}^f = f_{n_t-1} \circ f_{n_t-2} \circ \dots \circ f_0(\mathbf{M}), \quad (5)$$

$$f_i(\mathbf{M}_i) = \text{LReLU}(\text{ModConv}(\mathbf{M}_i, t_i, \mathbf{z})), \quad (6)$$

where n_t denotes the number of layers for texture learning. $t_i \in \{1, 2\}$ is the upsampling scale for the i -th layer.

Bridging Representations with Neural Feature Field. To connect the structural and the textural representations in the framework, we introduce a neural radiance field [28] as the bridge. Different from the implicit function in the original NeRF, which maps coordinates to pixel values, we first query the coordinate descriptor, \mathbf{v} , from the feature volume, \mathbf{V} , given a 3D coordinate \mathbf{x} , and then concatenate it with \mathbf{x} to obtain $\mathbf{v}^\mathbf{x}$ as the input. Then, the implicit function transform $\mathbf{v}^\mathbf{x}$ to the density and feature vector of the field. The above process can be formulated as

$$\mathbf{v} = \text{trilinear}(\mathbf{V}, \mathbf{x}), \quad (7)$$

$$\mathbf{v}^\mathbf{x} = \text{Concat}(\mathbf{v}, \mathbf{x}), \quad (8)$$

$$\Phi(\mathbf{v}^\mathbf{x}) = \phi_{n-1} \circ \phi_{n-2} \circ \dots \circ \phi_0(\mathbf{v}^\mathbf{x}), \quad (9)$$

$$\phi_i(\mathbf{v}_i^\mathbf{x}) = \sin(\gamma_i(\mathbf{z}) \cdot (\mathbf{W}_i \mathbf{v}_i^\mathbf{x} + \mathbf{b}_i) + \beta_i(\mathbf{z})), \quad (10)$$

$$\mathbf{f}(\mathbf{x}, \mathbf{d}) = \phi_f(\Phi(\mathbf{v}^\mathbf{x}), \mathbf{d}), \quad (11)$$

$$\sigma(\mathbf{x}) = \phi_d(\Phi(\mathbf{v}^\mathbf{x})), \quad (12)$$

where n denotes the number of layers to parameterize the neural field, while \mathbf{W}_i and \mathbf{b}_i are the learnable layer-wise weight and bias. Eq. (8) concatenates coordinates \mathbf{x} onto feature \mathbf{v} to explicitly introduce the structural information. Eq. (10) follows Chan *et al.* [3], which conditions the layer-wise output of the backbone $\Phi(\cdot)$ on the frequencies, $\gamma_i(\cdot)$, and phase shifts, $\beta_i(\cdot)$, learned from the random noise \mathbf{z} . Eq. (11) replaces the color modeling in Eq. (1) with feature modeling.

A per-pixel final feature \mathbf{m} can be obtained via volume rendering along a ray \mathbf{r} (with viewing direction \mathbf{d}). A collection of \mathbf{m} regarding different ray groups into a 2D feature map as the *textural representation*, \mathbf{M} , which will be further used to render the image.

$$\mathbf{m}(\mathbf{r}) = \sum_{k=1}^N T_k (1 - \exp(-\sigma(\mathbf{x}_k) \delta_k)) \mathbf{f}(\mathbf{x}_k, \mathbf{d}), \quad (13)$$

$$T_k = \exp\left(-\sum_{j=1}^{k-1} \sigma(\mathbf{x}_j) \delta_j\right). \quad (14)$$

Eq. (13) approximates the integral of N points $\{\mathbf{x}_k\}_{k=1}^N$ on the sampled ray \mathbf{r} , where $\delta_k = \|\mathbf{x}_{k+1} - \mathbf{x}_k\|_2$ stands for the distance between adjacent sampled points.

3.3. Training Pipeline

Generative Sampling. The whole generation process is formulated as $\mathbf{I}^f = G(\mathbf{z}, \xi)$, where \mathbf{z} is a latent code sampled from a Gaussian distribution $\mathcal{N}(0, 1)$ and ξ denotes the camera pose sampled from a prior distribution p_ξ . p_ξ is tuned for different datasets as either Gaussian or Uniform.

Discriminator. Like existing approaches for 3D-aware image synthesis [3, 30, 35], we employ a discriminator $D(\cdot)$ to compete with the generator. The discriminator is a CNN consisting of several residual blocks like [17].

Training Objectives. During training, we randomly sample \mathbf{z} and ξ from the prior distributions, while the real images \mathbf{I}^r are sampled from the real data distribution $p_{\mathcal{D}}$. The generator and the discriminator are jointly trained with

$$\min \mathcal{L}_G = \mathbb{E}_{\mathbf{z} \sim p_z, \xi \sim p_\xi} [f(D(G(\mathbf{z}, \xi)))], \quad (15)$$

$$\min \mathcal{L}_D = \mathbb{E}_{\mathbf{I}^r \sim p_{\mathcal{D}}} [f(-D(\mathbf{I}^r)) + \lambda \|\nabla_{\mathbf{I}^r} D(\mathbf{I}^r)\|_2^2], \quad (16)$$

where $f(t) = -\log(1 + \exp(-t))$ is the softplus function. The last term in Eq. (16) stands for the gradient penalty regularizer and λ is the loss weight.

4. Experiment

4.1. Settings

Datasets. We evaluate the proposed VolumeGAN on five real-world unstructured datasets including CelebA [22],

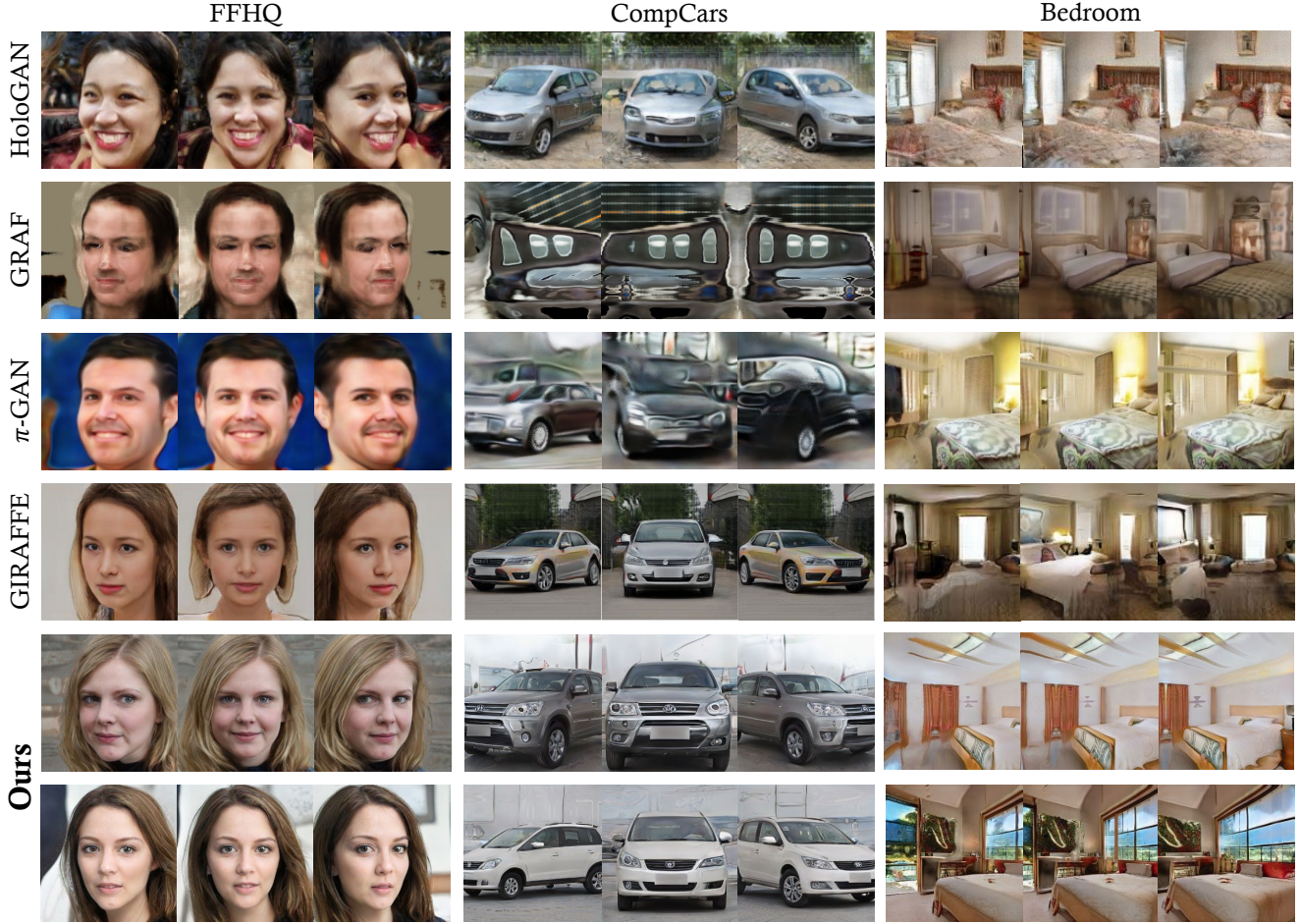


Figure 3. **Qualitative comparison between our VolumeGAN and existing alternatives** on FFHQ [16], CompCars [45], and LSUN bedroom [46] datasets. All images are in 256×256 resolution.

Cats [47], FFHQ [16], CompCars [45], LSUN bedroom [46], and a synthetic dataset Carla [6]. CelebA contains around $20K$ face images from $10K$ identities. The crop from the top of the hair to the bottom of the chin is adopted for data preprocessing on CelebA. The Cats dataset contains $6.5K$ images of cat heads at 128×128 resolution. FFHQ contains $70K$ images of real human faces in a resolution of 1024×1024 . We follow the protocol of [16] to preprocess the faces of FFHQ. CompCars includes $136K$ real cars whose pose varies greatly. The original images are in different aspect ratios. Hence we center crop the cars and resize them into 256×256 . Carla dataset contains $10K$ images which are rendered from Carla Driving simulator [6] using 16 car models with different textures. LSUN bedroom includes $300K$ samples in various camera views and aspect ratios. We also use center cropping to preprocess the bedroom images. We train VolumeGAN on resolutions of 128×128 for CelebA, Cats, and Carla and 256×256 for FFHQ, CompCars and LSUN bedroom.

Baselines. We choose four 3D-aware image synthesis approaches as the baselines, including HoloGAN [29], GRAF [35], π -GAN [3] and GIRAFFE [30]. Baseline models are officially released by the original papers or trained with the official implementation. More details can be found in **Appendix**.¹

Implementation Details. The learnable 3D template V_0 are randomly initialized in $4 \times 4 \times 4$ shape and 3D convolutions with kernel size $3 \times 3 \times 3$ are stacked to embed the template, resulting in the feature volume in $32 \times 32 \times 32$ resolution. We sample rays in a resolution of 64×64 , and four conditioned MLPs (SIREN [3, 38]) with 256 dimensions are adopted to model the feature field and the volume density. We use an Upsample block [17] and two 1×1 ModConv [2, 17] at each resolution for the neural renderer until reaching the output image resolution. We also apply progressive training strategy used in StyleGAN [16]

¹We fail to reproduce HoloGAN on LSUN bedroom with the **official implementation**, hence we do not report the quantitative results. The qualitative results of bedrooms are borrowed from the original paper [29].

Table 1. **Quantitative comparisons on different datasets.** FID [11] (lower is better) is used as the evaluation metric. Numbers in brackets indicate the improvements of our VolumeGAN over the second method.

Method	CelebA 128	Cats 128	Carla 128	FFHQ 256	CompCars 256	Bedroom 256
HoloGAN [29]	39.7	40.4	126.4	72.6	65.6	—
GRAF [35]	41.1	28.9	41.6	81.3	222.1	63.9
π -GAN [3]	15.9	17.7	30.1	53.2	194.5	33.9
GIRAFFE [30]	17.5	20.1	30.8	36.7	27.2	44.2
VolumeGAN (Ours)	8.9 (−7.0)	5.1 (−12.6)	7.9 (−22.2)	9.1 (−27.6)	12.9 (−14.3)	17.3 (−16.6)

and PG-GAN [14] to achieve better image qualities. For the network training, we use Adam [18] optimizer with $\beta_0 = 0$ and $\beta_1 = 0.999$ over 8 GPUs. The entire training requires the discriminator to see 25000K images. The batch size is 64, and the weight decay is 0. Unless specified, λ in Eq. (16) is set to 1 to balance different loss terms, and the learning rate of generator and discriminator is set to 1×10^{-4} and 2×10^{-4} , respectively. More details about network architecture and training can be found in Appendix.

4.2. Main Results

Qualitative Results. Fig. 3 compares the synthesized images of our method with baselines on FFHQ, CompCars and LSUN bedroom. The images are sampled from three views and synthesized in a resolution of 256×256 for visualization. Although all baseline methods can synthesize images under different camera poses on FFHQ, they suffer from low image quality and the identity shift across different angles. When transferred to challenging CompCars with larger viewpoint variations, some baselines such as GRAF [35] and π -GAN [3] struggle to generate realistic cars. HoloGAN can achieve good image quality but suffers from multi-view inconsistency. GIRAFFE can generate realistic cars while the color of the cars changes significantly under different views. When tested on bedroom, HoloGAN, GRAF, π -GAN and GIRAFFE cannot handle such indoor scene data with larger structure and texture variations.

VolumeGAN can synthesize high-fidelity view-consistent images. Compared with the existing approaches, it generates more fine details, such as teeth (face), headlights (car) and windows (bedroom). Even on the more challenging CompCars and LSUN bedroom datasets, VolumeGAN still achieves satisfying synthesis performance thanks to the feature volume and the neural renderer.

Quantitative Results. We quantitatively evaluate the visual quality of the synthesized images using Frechet Inception Distance (FID) [11]. We follow the evaluation protocol of StyleGAN [16] which adopts 50K real and fake samples to calculate the FID score. All baseline models are evaluated with the same setting for a fair comparison. As shown in Tab. 1, our approach leads to a significant improvement

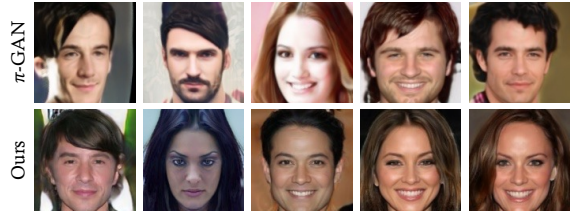


Figure 4. **Synthesized results with the front camera view** by π -GAN [3] and our VolumeGAN, where the faces proposed by VolumeGAN are more consistent to the given view, suggesting a better 3D controllability.

compared with baselines, particularly on the challenging datasets with the larger pose variation or the finer details. Note that although GIRAFFE also uses the neural renderer, our method still outperforms it with a clear margin. It demonstrates that the structural information encoded in the feature volume provides representative visual concepts, resulting in better images quality.

4.3. Ablation Studies

We conduct ablation studies on CelebA 128×128 to examine the importance of each component in VolumeGAN.

Metrics. In addition to the *FID* score that measures the image quality, we also provide two quantitative metrics to measure the multi-view consistency and the precision of 3D control as follows. 1) *Reprojection Error*. We first extract the underlying geometry of an object from the generated density using marching cubes [23]. Then, We render each object in sequence and sample five viewpoints uniformly to synthesize the images. The depth of each image is rendered from the resulting extracted mesh, which is used to calculate the reprojection error on two consecutive views by warping them each other. Specifically, we fix the yaw to be 0 and sample pitch from $[-0.3, 0.3]$. The marching cube is set to 10 due to the best visualization results of meshes. The reprojection error is calculated in the normalized image space $[-1, +1]$ like [1, 43, 50] to evaluate the multi-view consistency. 2) *Pose Error*. We synthesize 20,000 images and regard the results predicted from the head pose estimator [48] as the ground truth. The L1 distance between the given camera pose and the predicted pose is reported to evaluate the 3D control quantitatively.

Ablations on VolumeGAN Components. Our approach proposes to use Feature Volume as the structural representation and adopt the neural renderer consisting of ModConv to render textural representation into high-fidelity images. We ablate them to better understand their individual contributions. Our baseline is built upon π -GAN [3] using conditioned MLPs to achieve 3D-aware image synthesis by mapping coordinates to RGB color. The layer number of the baseline is set to be 4, the same as our setting illustrated in Sec. 4.1 for a fair comparison. As shown in Tab. 2, introducing the feature volume that provides the structural representation could further improve the FID score of the baseline approach from 18.7 to 13.6.

More importantly, lower reprojection error and pose error are also achieved, demonstrating the structural representation from the feature volume not only facilitates better visual results but also maintains the 3D properties regarding multi-view consistency and 3D explicit controlling. On top of this, the neural renderer further enhances FID to 8.9 with a slight drop in reprojection error and pose error, leading to the new state-of-the-art result on 3D-aware image synthesis. Notably, involving the neural renderer to the baseline could also boost the FID score but apparently sacrifice the 3D properties to some extent according to the 3D metrics. It also indicates that FID is not a comprehensive metric to evaluate 3D-aware image synthesis. In addition, Fig. 4 gives several synthesized samples of π -GAN baseline and our approach under the front view. More samples can be found in Appendix. Qualitatively, the poses of our synthesized samples are closer to the given camera view which is quantitatively reflected by the pose-error score.

Resolution of the Feature Volume. The feature volume resolution depicts the spatial refinement of the structural representation, and thus it plays an essential role in synthesizing images. Tab. 3 presents the metrics of the synthesis results for various resolutions of feature volume. As the resolution increases, the multi-view consistency and 3D control become better consistently while the visual quality measured by FID fluctuates little. This demonstrates that a more detailed feature volume provides better geometry consistency across various camera poses. However, increasing the feature volume resolution inevitably results in a greater computational burden. As a result, we choose a feature volume resolution of 32 in all of our experiments to maintain the balance between efficiency and image quality.

Neural Renderer Depth. The neural renderer is adopted to convert textural representations into 2D images; thus, its capacity is critical to the quality of the generated images. We adjust its capacity by varying the depth of the neural renderer to investigate its effect. Tab. 4 shows a trade-off between image quality and 3D properties. As the depth of the network increases, better image visual quality can be achieved while the quality of multi-view consistency and

Table 2. **Ablation studies on the components of VolumeGAN**, including the feature volume (FV) and the neural renderer (NR). “Rep-Er” and “Pose-Er” are the reprojection-error and pose-error.

FV	NR	FID	Rep-Er	Pose-Er
π -GAN		18.7	0.071	12.7
✓		13.6	0.031	8.3
	✓	11.3	0.103	12.1
✓	✓	8.9	0.037	8.6

Table 3. **Effect of the size of feature volume.** “Str Res” denotes the resolution of the feature volume (*i.e.*, the structural representation).

Str Res	FID	Rep-Er	Pose-Er	Speed (fps)
16	9.0	0.040	9.1	5.58
32	8.9	0.037	8.6	5.15
64	9.2	0.032	8.4	3.86

Table 4. **Effect of the depth of neural renderer.** “Tex Res” denotes the resolution of the 2D feature map (*i.e.*, the textural representation).

Depth	Tex Res	FID	Rep-Er	Pose-Er
6	64	8.0	0.051	9.7
4	64	8.8	0.046	9.3
2	64	8.9	0.037	8.6

3D control downgrades. This implies that increasing the capacity of the neural renderer would damage the 3D structure to some extent, revealing FID is not a comprehensive metric for 3D-aware image synthesis again. We thus choose the shallower network as the neural renderer for better 3D consistency and control.

4.4. Properties of Learned Representations

A key advantage of our approach over previous attempts is that by separately modeling the structure and texture with the 3D feature volume and 2D feature map, our model learns the disentangle representations for the object. These representations allow us to achieve control of the shape and appearance. The coordinate descriptor and the 3D mesh extracted from the density are visualized to interpret the learned representations.

Independent Control of Structure and Texture. At test time, we could easily swap and combine the latent codes regarding the structural and textural individually. In this way, we can investigate whether such two representations are well disentangled. For example, we could combine the structural representation (*i.e.*, feature volume code) of a certain instance with the textural (*i.e.*, generative feature field and neural renderer code) of another. The corresponding results are shown in Fig. 5. The faces results show that the feature volume code controls the shape of the face and hairstyle, whereas the feature field and neural

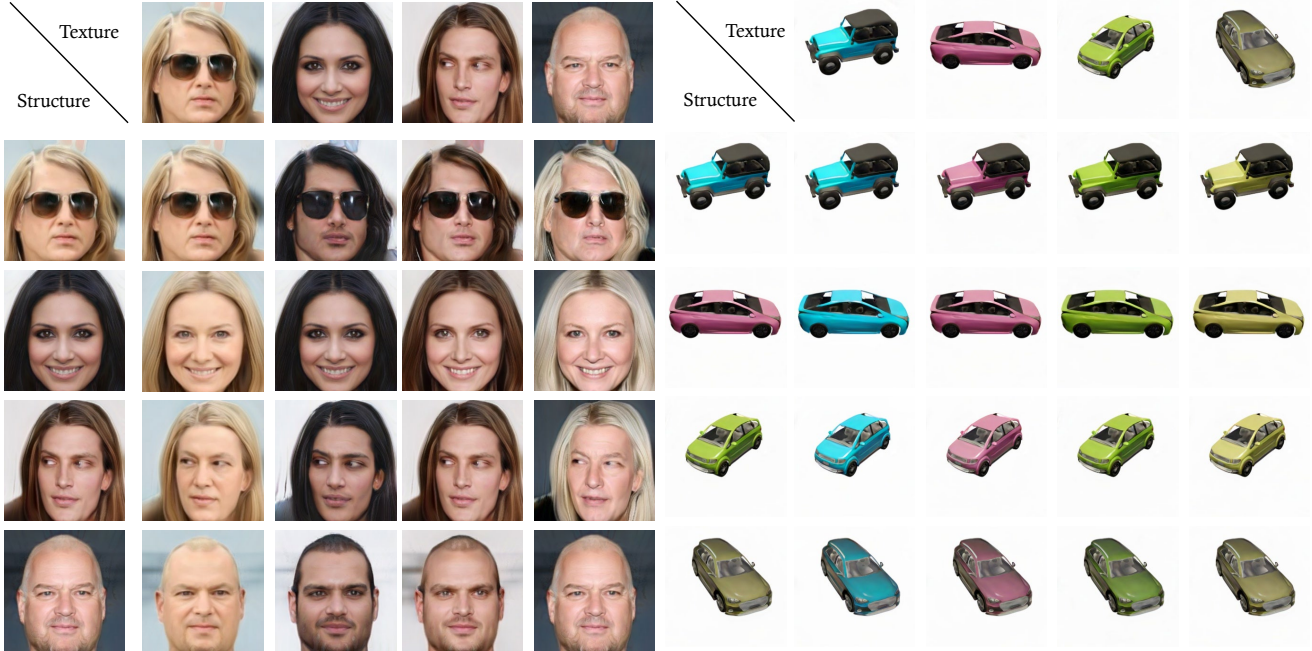


Figure 5. Synthesized results by exchanging the structural and the textural latent codes.

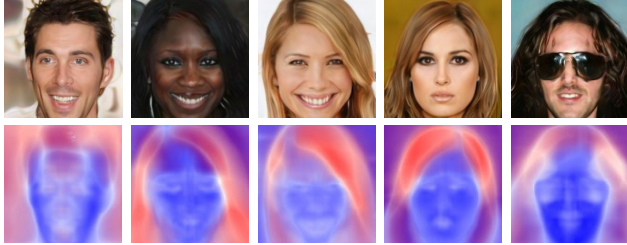


Figure 6. Visualization of coordinate descriptor. PCA is used to reduce the feature dimension.

renderer code determine the skin and hair color. Concretely, glasses are controlled by the volume code, in line with our perception. We can swap the structure and texture of cars successfully. It demonstrates that our method can disentangle shape and appearance in synthesizing images. Different from GRAF [35] and GIRRAFE [30], we do not explicitly introduce shape code and appearance code to control image synthesis. Thanks to the structural and textural representations in our framework, the disentanglement between shape and appearance emerges naturally.

Coordinate Descriptor Visualization. To further explore how the feature volume describes the underlying structure, we visualize the corresponding coordinate descriptors queried in the feature volume. Specifically, we accumulate the coordinate descriptors on each ray, resulting in a high-dimensional feature map. PCA [42] is utilized to reduce the dimension to 3 for visualization. Fig. 6 shows that the feature volume serves as a coarse structure template. The face outline, hair, and background can be recognized easily. Impressively, the eyes have a strong symmetry even

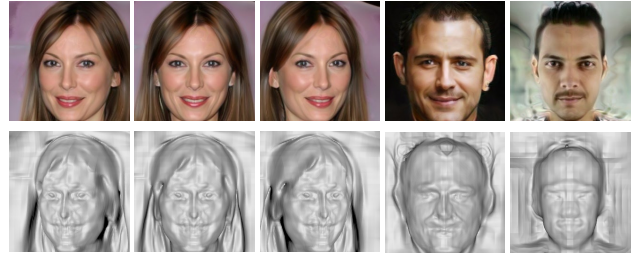


Figure 7. 3D Mesh extracted from the density.

with the glasses. Compared to raw coordinates, the feature descriptor provides a structured constraint to guide the image synthesis so that our method inherently synthesizes image with better visual quality and 3D properties.

Underlying Geometry. The volume density of the implicit representation can construct an underlying geometry of the object due to its view-independent properties. We extract the underlying geometry with marching cube [23] on the density, resulting in a surface mesh. Fig. 7 shows the meshes with various views and identities. The geometry is consistent across different views, supporting the good 3D properties of our method.

5. Conclusion and Discussion

In this paper, we propose a new 3D-aware generative model, *VolumeGAN*, for synthesizing high-fidelity images. By learning structural and textural representations, our model achieves sufficiently higher image quality and better 3D control on various challenging datasets.

Limitations. Despite the structural representation learned by VolumeGAN, the synthesized 3D mesh surface is still not smooth and lacks fine details. Meanwhile, even though we can improve the synthesis resolution via introducing a deeper CNN (*i.e.*, the neural renderer), it may weaken the multi-view consistency and 3D control. Future research will focus on generating fine-grained 3D shape as well as making the tailing CNN in VolumeGAN with improved 3D properties through introducing regularizers.

Ethical Consideration. Due to the high-quality 3D-aware synthesis performance, our approach is potentially applicable for deep fake generation. We strongly oppose the abuse of our method in violating privacy and security. On the contrary, we hope it can be used to improve the existing fake detection systems.

References

- [1] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan: How to embed images into the stylegan latent space? In *ICCV*, 2019. 2, 6
- [2] Ivan Anokhin, Kirill Demochkin, Taras Khakhulin, Gleb Sterkin, Victor Lempitsky, and Denis Korzhnikov. Image generators with conditionally-independent pixel synthesis. In *CVPR*, 2021. 5
- [3] Eric R Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021. 1, 2, 3, 4, 5, 6, 7, 11, 12
- [4] Xi Chen, Yan Duan, Rein Houthoofd, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *NIPS*, 2016. 2
- [5] Julian Chibane, Thiemo Alldieck, and Gerard Pons-Moll. Implicit functions in feature space for 3d shape reconstruction and completion. In *CVPR*, 2020. 2
- [6] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. In *Conference on robot learning*, 2017. 5
- [7] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Adv. Neural Inform. Process. Syst.*, 2014. 1, 2, 3
- [8] Jiatao Gu, Lingjie Liu, Peng Wang, and Christian Theobalt. Stylenerf: A style-based 3d-aware generator for high-resolution image synthesis. *arXiv preprint arXiv:2110.08985*, 2021. 2
- [9] Jinjin Gu, Yujun Shen, and Bolei Zhou. Image processing using multi-code gan prior. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. 2
- [10] Zhenliang He, Meina Kan, and Shiguang Shan. Eigengan: Layer-wise eigen-learning for gans. *ICCV*, 2021. 2
- [11] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Adv. Neural Inform. Process. Syst.*, 2017. 2, 6
- [12] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Int. Conf. Comput. Vis.*, 2017. 4
- [13] Ajay Jain, Matthew Tancik, and Pieter Abbeel. Putting nerf on a diet: Semantically consistent few-shot view synthesis. In *ICCV*, 2021. 2
- [14] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *Int. Conf. Learn. Represent.*, 2018. 2, 6
- [15] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. In *NIPS*, 2021. 2
- [16] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 1, 2, 4, 5, 6
- [17] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of StyleGAN. In *CVPR*, 2020. 2, 4, 5
- [18] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Int. Conf. Learn. Represent.*, 2015. 6
- [19] Yariv Lior, Kasten Yoni, Moran Dror, Galun Meirav, Atzmon Matan, Basri Ronen, and Lipman Yaron. Multiview neural surface reconstruction by disentangling geometry and appearance. In *NeurIPS*, 2020. 2
- [20] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *NeurIPS*, 2020. 2
- [21] Shaohui Liu, Yinda Zhang, Songyou Peng, Boxin Shi, Marc Pollefeys, and Zhaopeng Cui. Dist: Rendering deep implicit signed distance function with differentiable sphere tracing. In *CVPR*, 2020. 2
- [22] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Int. Conf. Comput. Vis.*, 2015. 4
- [23] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM siggraph computer graphics*, 1987. 6, 8
- [24] Andrew L Maas, Awni Y Hannun, Andrew Y Ng, et al. Rectifier nonlinearities improve neural network acoustic models. In *ICML*, 2013. 4
- [25] Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *CVPR*, 2021. 2
- [26] Quan Meng, Anpei Chen, Haimin Luo, Minye Wu, Hao Su, Lan Xu, Xuming He, and Jingyi Yu. Gnerf: Gan-based neural radiance field without posed camera. In *ICCV*, 2021. 2
- [27] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 2
- [28] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf:

- Representing scenes as neural radiance fields for view synthesis. In *Eur. Conf. Comput. Vis.*, 2020. 1, 2, 3, 4
- [29] Thu Nguyen-Phuoc, Chuan Li, Lucas Theis, Christian Richardt, and Yong-Liang Yang. Hologan: Unsupervised learning of 3d representations from natural images. In *ICCV*, 2019. 2, 5, 6, 11
- [30] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021. 1, 2, 4, 5, 6, 8, 11
- [31] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *CVPR*, 2020. 2
- [32] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 2
- [33] William Peebles, John Peebles, Jun-Yan Zhu, Alexei Efros, and Antonio Torralba. The hessian penalty: A weak prior for unsupervised disentanglement. In *ECCV*, 2020. 2
- [34] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *AAAI*, 2018. 11
- [35] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. In *Adv. Neural Inform. Process. Syst.*, 2020. 1, 2, 3, 4, 5, 6, 8, 11
- [36] Yujun Shen, Ceyuan Yang, Xiaoou Tang, and Bolei Zhou. Interfacegan: Interpreting the disentangled face representation learned by gans. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2020. 1, 2
- [37] Yujun Shen and Bolei Zhou. Closed-form factorization of latent semantics in gans. In *CVPR*, 2021. 2
- [38] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *NIPS*, 2020. 5, 11
- [39] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. In *NeurIPS*, 2019. 2
- [40] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, 2017. 3
- [41] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. In *NeurIPS*, 2021. 2
- [42] Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 1987. 8
- [43] Yinghao Xu, Yujun Shen, Jiapeng Zhu, Ceyuan Yang, and Bolei Zhou. Generative hierarchical features from synthesizing images. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021. 1, 2, 6
- [44] Ceyuan Yang, Yujun Shen, and Bolei Zhou. Semantic hierarchy emerges in deep generative representations for scene synthesis. *Int. J. Comput. Vis.*, 2020. 1, 2
- [45] Linjie Yang, Ping Luo, Chen Change Loy, and Xiaoou Tang. A large-scale car dataset for fine-grained categorization and verification. In *CVPR*, 2015. 5
- [46] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015. 2, 5
- [47] Weiwei Zhang, Jian Sun, and Xiaoou Tang. Cat head detection-how to effectively exploit shape and texture features. In *ECCV*, 2008. 5
- [48] Yijun Zhou and James Gregson. Whenet: Real-time fine-grained estimation for wide range head pose. *BMVC*, 2020. 6
- [49] Jiapeng Zhu, Ruili Feng, Yujun Shen, Deli Zhao, Zhengjun Zha, Jingren Zhou, and Qifeng Chen. Low-rank subspaces in gans. *NIPS*, 2021. 2
- [50] Jiapeng Zhu, Yujun Shen, Deli Zhao, and Bolei Zhou. In-domain gan inversion for real image editing. In *Eur. Conf. Comput. Vis.*, 2020. 2, 6
- [51] Jun-Yan Zhu, Zhoutong Zhang, Chengkai Zhang, Jiajun Wu, Antonio Torralba, Joshua B. Tenenbaum, and William T. Freeman. Visual object networks: Image generation with disentangled 3D representations. In *NIPS*, 2018. 2

Appendix

This appendix is organized as follows. Appendix A and Appendix B introduce the network structure and the training configurations used in VolumeGAN. Appendix C describes the details of implementing baseline approaches. Appendix D shows more qualitative results.

A. Network Structure

Recall that, our VolumeGAN first learns a feature volume with *3D CNN*. The feature volume is then transformed into a feature field using a *NeRF-like model*. A 2D feature map is finally accumulated from the feature field and rendered to an image with a *2D CNN*. Taking 256 resolution as an instance, we illustrate the architectures of these three models in Tab. A1, Tab. A2, and Tab. A3, respectively.

B. Training Configurations

Because of the wildly divergent data distribution, the training parameters vary greatly on different datasets. Tab. A4 illustrates the detailed training configuration of different datasets. Fov, Range_{depth}, and #Steps are the field of view, depth range and the number of sampling steps along a camera ray. Range_h and Range_v denotes the horizontal and vertical angle range of the camera pose ξ . 'Sample_Dist' denotes the sampling scheme of the camera pose. We only use Gaussian or Uniform sampling in our experiments. λ is the loss weight of the gradient penalty.

Table A1. Network structure for learning a feature volume as the structural representation. The output size is with order $\{C \times H \times W \times D\}$, where D denotes the depth dimension.

Stage	Block	Output Size
input	Learnable Template	$256 \times 4 \times 4 \times 4$
block ₁	$\begin{bmatrix} 3 \times 3 \times 3 \text{ Conv, 128} \\ \text{AdaIN, 128} \\ \text{Upsample} \\ \text{LeakyReLU, 0.2} \end{bmatrix}$	$128 \times 8 \times 8 \times 8$
block ₂	$\begin{bmatrix} 3 \times 3 \times 3 \text{ Conv, 64} \\ \text{AdaIN, 64} \\ \text{Upsample} \\ \text{LeakyReLU, 0.2} \end{bmatrix}$	$64 \times 16 \times 16 \times 16$
block ₃	$\begin{bmatrix} 3 \times 3 \times 3 \text{ Conv, 32} \\ \text{AdaIN, 32} \\ \text{Upsample} \\ \text{LeakyReLU, 0.2} \end{bmatrix}$	$32 \times 32 \times 32 \times 32$

C. Implementation Details of Baselines

HoloGAN [29]. We use the official implementation of HoloGAN.² We train HoloGAN for 50 epochs. The generator of HoloGAN can only synthesize images in 64×64 or 128×128 resolution. We extend the generator with an extra Upsample and AdaIN block to synthesize 256×256 images for comparison.

GRAF [35]. We use the official implementation of GRAF.³ We directly use the pre-trained checkpoints of CelebA and Carla provided by the authors. For the other datasets, we train GRAF with the same data and camera parameters as ours at the target resolution.

π -GAN [3]. We use the official implementation of π -GAN.⁴ We also directly use the pre-trained checkpoints of CelebA, Carla and Cat for comparison and retrain π -GAN models on the other three datasets, including FFHQ, CompCars and LSUN bedroom. The retrained models are progressively trained from a resolution of 32×32 to 256×256 following the official implementation.

GIRAFFE [30]. We use the official GIRAFFE implementation.⁵ GIRAFFE provides the pre-trained weights of FFHQ and CompCars in a resolution of 256×256 . The remaining datasets are also trained with the same camera distribution for a fair comparison.

D. Additional Results

Synthesis with front camera view. To better illustrate the 3D controllability, we show additional results of generating

Table A2. Network structure of the generative feature field. The output size is with order $\{H \times W \times S \times C\}$, where S is the number of sampling points along a certain camera ray. FiLM denotes the FiLM layer [34] and Sine stands for the Sine activation [38].

Stage	Block	Output Size
input	—	$64 \times 64 \times 12 \times (32 + 3)$
mlp ₁	$\begin{bmatrix} \text{FC, 256} \\ \text{FiLM, 256} \\ \text{Sine} \end{bmatrix}$	$64 \times 64 \times 12 \times 256$
mlp ₂	$\begin{bmatrix} \text{FC, 256} \\ \text{FiLM, 256} \\ \text{Sine} \end{bmatrix}$	$64 \times 64 \times 12 \times 256$
mlp ₃	$\begin{bmatrix} \text{FC, 256} \\ \text{FiLM, 256} \\ \text{Sine} \end{bmatrix}$	$64 \times 64 \times 12 \times 256$
mlp ₄	$\begin{bmatrix} \text{FC, 256} \\ \text{FiLM, 256} \\ \text{Sine} \end{bmatrix}$	$64 \times 64 \times 12 \times 256$

Table A3. Network structure of the neural renderer, which renders a 2D feature map to a synthesized image. The output size is with order $\{C \times H \times W\}$.

Stage	Block	Output Size
input	—	$256 \times 64 \times 64$
block ₁	$\begin{bmatrix} 1 \times 1 \text{ ModConv, 128} \\ \text{LeakyReLU, 0.2} \\ 1 \times 1 \text{ ModConv, 128} \\ \text{Upsample} \\ \text{LeakyReLU, 0.2} \end{bmatrix}$	$128 \times 128 \times 128$
block ₂	$\begin{bmatrix} 1 \times 1 \text{ ModConv, 64} \\ \text{LeakyReLU, 0.2} \\ 1 \times 1 \text{ ModConv, 64} \\ \text{Upsample} \\ \text{LeakyReLU, 0.2} \end{bmatrix}$	$64 \times 256 \times 256$
RGB	$3 \times 3 \text{ Conv, 3}$	$3 \times 256 \times 256$

images with the front view. As shown in Fig. A1, the faces synthesized by VolumeGAN are more consistent with the given view, demonstrating a better 3D controllability.

Synthesis with varying camera views. Besides the front camera view, we provide a demo video, which shows more results with varying camera views. From the video, we can see the continuous 3D control achieved by our VolumeGAN. We also include in the demo video the comparisons with the state-of-the-art methods, *i.e.*, π -GAN [3] and GIRAFFE [30].

²<https://github.com/thunguyenphuoc/HoloGAN>

³<https://github.com/autonomousvision/graf>

⁴<https://github.com/marcoamonteiro/pi-GAN>

⁵<https://github.com/autonomousvision/giraffe>

Table A4. Training configurations regarding different datasets.

Datasets	Fov	Range _{depth}	#Steps	Range _h	Range _v	Sample_Dist	λ
CelebA	12	[0.88, 1.12]	12	$[\pi/2 - 0.3, \pi/2 + 0.3]$	$[\pi/2 - 0.15, \pi/2 + 0.15]$	Gaussian	0.2
Cat	12	[0.8, 1.2]	12	$[\pi/2 - 0.5, \pi/2 + 0.5]$	$[\pi/2 - 0.4, \pi/2 + 0.4]$	Gaussian	0.2
Carla	30	[0.7, 1.3]	36	$[0, 2\pi]$	$[\pi/2 - \pi/8, \pi/2 + \pi/8]$	Uniform	1
FFHQ	12	[0.8, 1.2]	14	$[\pi/2 - 0.4, \pi/2 + 0.4]$	$[\pi/2 - 0.2, \pi/2 + 0.2]$	Gaussian	1
CompCars	20	[0.8, 1.2]	30	$[0, 2\pi]$	$[\pi/2 - \pi/8, \pi/2 + \pi/8]$	Uniform	1
Bedroom	26	[0.7, 1.3]	40	$[\pi/2 - \pi/8, \pi/2 + \pi/8]$	$[\pi/2 - \pi/10, \pi/2 + \pi/10]$	Uniform	1

Synthesized samples with the front camera view by π -GAN

Synthesized samples with the front camera view by VolumeGAN

Figure A1. **More Synthesized results with the front camera view** by π -GAN [3] and our VolumeGAN, where the faces proposed by VolumeGAN are more consistent with the given view, suggesting a better 3D controllability.