AE-NeRF: Auto-Encoding Neural Radiance Fields for 3D-Aware Object Manipulation

Mira Kim*, Jaehoon Ko*, Kyusun Cho, Junmyeong Choi, Daewon Choi,
and Seungryong Kim†

Abstract—**We propose a novel framework for 3D-aware object manipulation, called Auto-Encoding Neural Radiance Fields (AE-NeRF). Our model, which is formulated in an auto-encoder architecture, extracts disentangled 3D attributes such as 3D shape, appearance, and camera pose from an image, and a high-quality image is rendered from the attributes through disentangled generative Neural Radiance Fields (NeRF). To improve the disentanglement ability, we present two losses, global-local attribute consistency loss defined between input and output, and swapped-attribute classification loss. Since training such auto-encoding networks from scratch without ground-truth shape and appearance information is non-trivial, we present a stage-wise training scheme, which dramatically helps to boost the performance. We conduct experiments to demonstrate the effectiveness of the proposed model over the latest methods and provide extensive ablation studies.**

Index Terms—**Implicit 3D representation, neural radiance field, 3D-aware image manipulation, 3D reconstruction**

I. INTRODUCTION

Manipulating an object image in a 3D-aware fashion, e.g., deforming 3D shapes, adjusting textures, or moving camera viewpoints, is one of the most fundamental and essential tasks in Computer Vision (CV) and Computer Graphics (CG) fields with numerous applications such as 3D content creation [1, 2] or augmented/virtual reality [3, 4, 5]. Due to its challenges, 2D image manipulation techniques [6, 7, 8] often fail to solve this problem due to lacking 3D awareness.

Exploring how to extract disentangled 3D attributes such as 3D shape, appearance, and camera viewpoint, and how to render a high-quality image from these attributes is the key to the 3D-aware object manipulation. For the first task, conventional techniques for monocular 3D reconstruction [9, 10, 11] rely on explicit 3D models such as mesh or voxels. They formally learn a model to extract disentangled 3D attributes, including 3D shape, texture, and camera viewpoint, from an image, and render an image from the attributes in an analysis-by-synthesis framework, where a consistency between the input and rendered image is encouraged [12, 13, 14, 15]. However, acquiring a high-quality explicit 3D model is notoriously challenging due to its limited representation capacity.

Recently, many literature have shown that an implicit volumetric representation allows for capturing and rendering high-resolution 3D structures [16, 17, 18, 19, 20, 21]. Among these

works, Neural Radiance Field (NeRF) [22] has recently proven success for novel view synthesis of a scene, which motivates us to use such representations for 3D-aware object manipulation task. However, since NeRF is an implicit model optimized per scene, it is non-trivial that directly controlling and editing an implicit continuous volumetric representation of a 3D object. To overcome this, some works presented image-conditional NeRF [23, 24] that extracts an image feature through an image encoder and conditions the NeRF, which enables partially conducting 3D-aware manipulation, e.g., moving camera viewpoint [23, 25, 26]. Nevertheless, they are not capable of editing 3D shape or appearance due to the lack of ability to extract full disentangled 3D attributes from an image.

On the other hand, some works such as GRAF [27] and GIRAFFE [28] presented a disentangled conditional NeRF, which aims to learn a generative model for radiance fields that renders a high-resolution image from the disentangled 3D attributes, such as 3D shape, appearance, and camera viewpoint. Since they are generative models starting from rafndomly-sampled latent vectors, they cannot be directly used for manipulation of given image. Some recent methods such as CodeNeRF [29], EditNeRF [30], and CLIP-NeRF [31] presented techniques to invert such disentangled conditional NeRF through a test-time optimization to explicitly edit a given image. However, due to extremely high dimensionality of 3D shape, appearance, and camera viewpoint space, such optimization is also challenging, which generates sub-optimal results, as exemplified in 2D GAN inversion literature [32, 33].

In this paper, we present a novel framework for 3D-aware object manipulation, dubbed Auto-Encoding Neural Radiance Fields (AE-NeRF), to tackle the aforementioned issues. Inspired by auto-encoder [34], we formulate, for the first time, an auto-encoder architecture for this task, consisting of encoder and decoder, where the former is designed to extract disentangled 3D attributes from an image and the latter is designed to render a high-quality image through disentangled conditional NeRF. In comparison to existing monocular 3D reconstruction methods [9, 10, 11] that rely on explicit 3D representations, our architecture allows for reconstructing an image more accurately by leveraging implicit volumetric representations, thereby realizing an analysis-by-synthesis framework better. To train the networks by exclusively using 2D image supervision without any 3D supervision, we apply image rendering loss and perceptual reconstruction losses for better reconstruction and an adversarial loss for more realistic generation. To boost the disentanglement performance, we present two losses, namely global-local attribute consistency

*Equal contribution

†Corresponding author

M. Kim, J. Ko, K. Cho, J. Choi, D.Choi and S. Kim are with the Department of Computer Science and Engineering, Korea University, Seoul 02841, Korea. (E-mail: {miramira227, kjh9604, kyustorm7, chedge, daeone0920, seungryong_kim}@korea.ac.kr).

TABLE I: **Comparison of AE-NeRF with relevant existing methods,** including original NeRF [22], PixelNeRF [23], GIRAFFE [28], CodeNeRF [29], EditNeRF [30], CG-NeRF [35], and CLIP-NeRF [31].

	NeRF [22]	PixelNeRF [23]	GIRAFFE [28]	EditNeRF [29]	CodeNeRF [30]	CG-NeRF [35]	CLIP-NeRF [31]	AE-NeRF (Ours)
Input image reconstruction?	✓	✓	✗	✓	✓	✗	✓	✓
Input image-conditioned?	✗	✓	✗	✓	✓	✓	✓	✓
Disentangled 3D attributes?	✗	✗	✓	✓	✓	✓	✓	✓
No optimization at test time?	✗	✓	✓	✗	✗	✓	✗	✓
Camera estimation?	✗	✗	✗	✗	✓	✓	✗	✓

loss and swapped-attribute classification loss. We also present a stage-wise training scheme to stably learn our auto-encoder architecture. Comparison of our AE-NeRF with relevant existing works is summarized in Table I.

The presented approach is evaluated on several standard benchmarks and examined in an ablation study. The experimental results for 3D-aware object manipulation show that this model outperforms the latest methods.

II. RELATED WORK

A. Learning-based Monocular 3D Reconstruction

Previous works on learning explicit 3D representation from an image can be categorized with respect to the representation types used to express an object, such as point clouds [36, 37], meshes [38, 39], or voxels [40, 41]. As shown in Fig. 1 (a), they formally learn a model to extract disentangled 3D attributes, including 3D shape, texture, and camera viewpoint, from an image, and render an image from the attributes in an analysis-by-synthesis framework, where a consistency between the input and rendered image is encouraged [10, 9, 1, 2, 13, 42, 15]. Some others attempted to relax the constrains of using 3D supervisions, such as keypoints, camera pose [2, 1], or categorical mean shape [43]. Meanwhile, others [13, 42, 15] tried to improve approximation process of rasterization, that yields more compelling results on reconstruction tasks. However, since its innate memory inefficiency, such explicit representation results in limited rendering quality.

B. Implicit Neural 3D Representation

Recent works have investigated the representation of continuous 3D shape by mapping xyz coordinates to occupancy fields [16] or signed distance functions [44]. Recently, NeRF [22] shows the success of the implicit representation of 3D models, which encodes a continuous volume representation of shape and view-dependent appearance with MLP. Their success inspired many follow-up works [45, 46, 47, 48, 49]. Although NeRF demonstrates outstanding results on novel view synthesis task, per-scene optimization induces high computational demands and requires multiple images with calibrated camera parameters for training a specific scene. Many approaches [26, 23, 25] focus on acquiring a scene prior to extract features from input images to an image-conditioned rendering, which is often called conditional NeRF, as shown in Fig. 1 (b), but they are limited to manipulate given image, e.g., by disentangling its attributes.

C. Generative NeRF

To reformulate the NeRF in a generative fashion, the adversarial learning framework [27, 28, 50, 51] has been explored, where the mapping function is learned between an image and randomly-sampled attributes, e.g., 3D shape, appearance, and camera pose, as shown in Fig. 1 (c). GRAF [27] first took NeRF in an adversarial framework, and achieved high-quality generation performance. GIRAFFE [28] modeled a scene with multiple entities by feeding coarse feature volume to simple CNN-based neural renderer. More recently, some other works [50, 51] adopt advanced neural rendering techniques, or [52, 53] propose efficient point sampling strategy to achieve high-resolution image generation. Even though they achieved promising results, process of generating images from random latent codes cannot be directly used for image manipulation.

D. Disentangled Image Manipulation

Disentangling an image as interpretable attributes can be used in many downstream applications, such as image-to-image translation [54, 55, 56], domain adaptation [57], or image manipulation [6, 58, 59, 60]. For instance, Swapping Auto-encoder [8] learns a disentangled embedding space and manipulates an image. However, these works mainly focused on modeling 2D attributes. For 3D-aware object manipulation, a few recent works [29, 30, 31] presented the use of NeRF. CodeNeRF [29] estimated shape, texture, and camera viewpoint of an object from a single image by optimizing a pre-trained NeRF. CoNeRF [61] trained NeRF with a small number of mask annotations via auto-decoding optimization so that the user can manipulate the semantic parts of the image at test time. EditNeRF [30] learned to disentangle color from shape when predicting the radiance of a point, enabling editing shape and color. As a concurrent work, CG-NeRF [35] disentangles attributes given an input with pose-consistent diversity loss for view-consistent multi-modal outputs. CLIP-NeRF [31] exploits the GAN inversion [62] to effectively optimize disentangled latent codes. Our method differs from these methods in that our model directly predicts the disentangled 3D attributes through a learned encoder, as shown in Fig. 1 (d),

III. METHODOLOGY

A. Motivation and Overview

Let us denote an image as I that we would like to manipulate. Here, our objective is to manipulate I in a 3D-aware
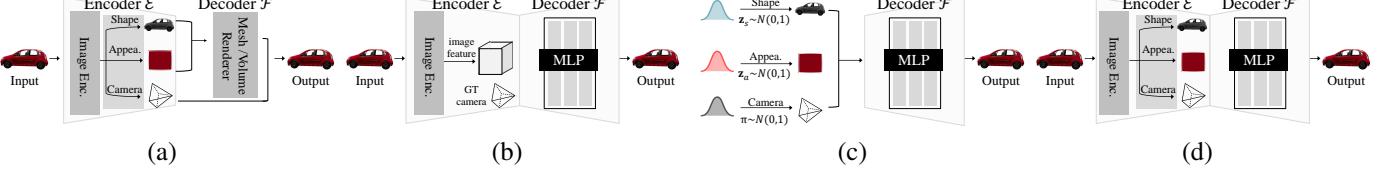
Fig. 1: **Intuition of AE-NeRF:** (a) monocular 3D reconstruction [10, 13, 11, 63] that extracts disentangled 3D attributes from input, (b) image-conditional NeRF [23, 24, 26, 25] that trains the feature encoder and NeRF as decoder, (c) disentangled generative NeRF [27, 28] that renders images from randomly sampled disentangled 3D attributes, and (d) auto-encoding NeRF (AE-NeRF) that extracts the disentangled 3D attributes from input and renders images from these attributes through disentangled generative NeRF, which can be effectively used for 3D-aware object manipulation.

fashion, e.g., novel view synthesis, shape and appearance swap, or camera pose swap with another image J. To achieve this, we focus on how to extract disentangled 3D attributes from an image and how to render a high-quality output from these attributes, which can be formulated with an auto-encoder architecture [34]. We design a model $\mathcal{G}$, as shown in Fig. 2, consisting of the encoder $\mathcal{E}$ and decoder $\mathcal{F}$, where the former is designed to extract disentangled 3D attributes, i.e., appearance code $\mathbf{z}_a$ and shape code $\mathbf{z}_s$, as well as camera pose π, from the input image I, and the latter is designed to render the output image I' from these attributes through disentangled generative NeRF. At the manipulation phase, we extract disentangled 3D attributes through the learned encoder given an image, adjust the attributes conforming to user intention, and then render a result through the learned decoder.

B. Encoder: Disentangled 3D Attribute Extraction

To understand 3D perspective of an object, disentangling camera pose and object properties, e.g., shape and appearance, from an image is of prime importance [27]. However, achieving this from a single image is notoriously challenging without explicit supervisions of them or prior knowledge. In 2D image manipulation, there were many attempts to disentangle an image as content and style [64, 65, 66, 67], but they lack 3D awareness, thus hindering applicability. To overcome this, similar to existing 3D shape recovery methods [11, 10], we attempt to extract disentangled 3D geometry and appearance information. But, unlike them [11, 10] that rely on *explicit* 3D structure, e.g., mesh, point cloud or voxel, our encoder is designed to extract the latent codes for *implicit* 3D representation, i.e., neural radiance fields, that can be used for NeRF in the decoder.

Specifically, our encoder $\mathcal{E}$ is composed of the image encoder and the separate attribute estimators; the former takes as input an image I and extracts a semantically meaningful feature through CNNs [68], and the latter outputs appearance code $\mathbf{z}_a$ and shape code $\mathbf{z}_s$, as well as camera pose π, as follows:

$$\{\mathbf{z}_a, \mathbf{z}_s, \pi\} = \mathcal{E}(I). \tag{1}$$

During training, to encourage the networks to focus more on the context information, we hide a part of input image (i.e., patch I_{patch}) randomly. This simple technique dramatically improves the performance, as exemplified in Fig. 2.

C. Decoder: Disentangled Conditional NeRF

To design an auto-encoder architecture, we design the decoder $\mathcal{F}$ to form a mapping between disentangled 3D attributes $\{\mathbf{z}_a, \mathbf{z}_s, \pi\}$ and an image I. Existing image renderers for inverse graphics such as DIB-R [13] demand explicit 3D representation such as mesh, thus having limited image generation quality. Unlike those [13, 10], we leverage a neural rendering technique based on NeRF [22, 27, 28] that encodes an image as a continuous volumetric radiance field of color and density. Thanks to its strong capability in capturing high-resolution geometry [22, 27, 28], it enables rendering photo-realistic images in the current camera view and even novel camera views, which allows for building an accurate cycle across an image to latent space and to the image within our auto-encoder architecture.

Specifically, built upon the NeRF [22], our decoder is represented as a continuous volumetric function $\mathcal{F}$ which maps a 3D location $\mathbf{x}$ and a viewing direction[1] $\mathbf{d}$ from the estimated camera pose π, together with appearance code $\mathbf{z}_a$ and shape code $\mathbf{z}_s$, to volume density σ and RGB color $\mathbf{c}$, which can be formulated as

$$\{\sigma, \mathbf{c}\} = \mathcal{F}(\gamma(\mathbf{x}), \gamma(\mathbf{d}), \mathbf{z}_a, \mathbf{z}_s), \tag{2}$$

where $\gamma(\cdot)$ is a positional encoding [22]. For architecture design of $\mathcal{F}$, we follow the GRAF [27], while any other architectures [29, 30, 31] can also be used.

D. Loss Functions

1) Image Reconstruction Loss.: Thanks to auto-encoding nature, we can utilize a 2D rendering loss between an input image I and a rendered image $\mathcal{G}(I)$. However, due to high-computational burden of volume rendering in NeRF [22], generating an image at each iteration during training would be non-trivial. To overcome this, instead of an image, we render a patch $\mathcal{G}_{\text{patch}}(I)$ similar to the GRAF [27]. In addition, to encourage the networks to more focus on context information, we artificially hide a patch I_{patch} in the image I and make the networks recover the patch, which is proven to be effective in vision tasks [69, 70]. We formulate this with l-2 loss function between I_{patch} and $\mathcal{G}_{\text{patch}}(I)$ as

$$\mathcal{L}_{\text{render}}(I) = \sum_I \|I_{\text{patch}} - \mathcal{G}_{\text{patch}}(I)\|_2. \tag{3}$$

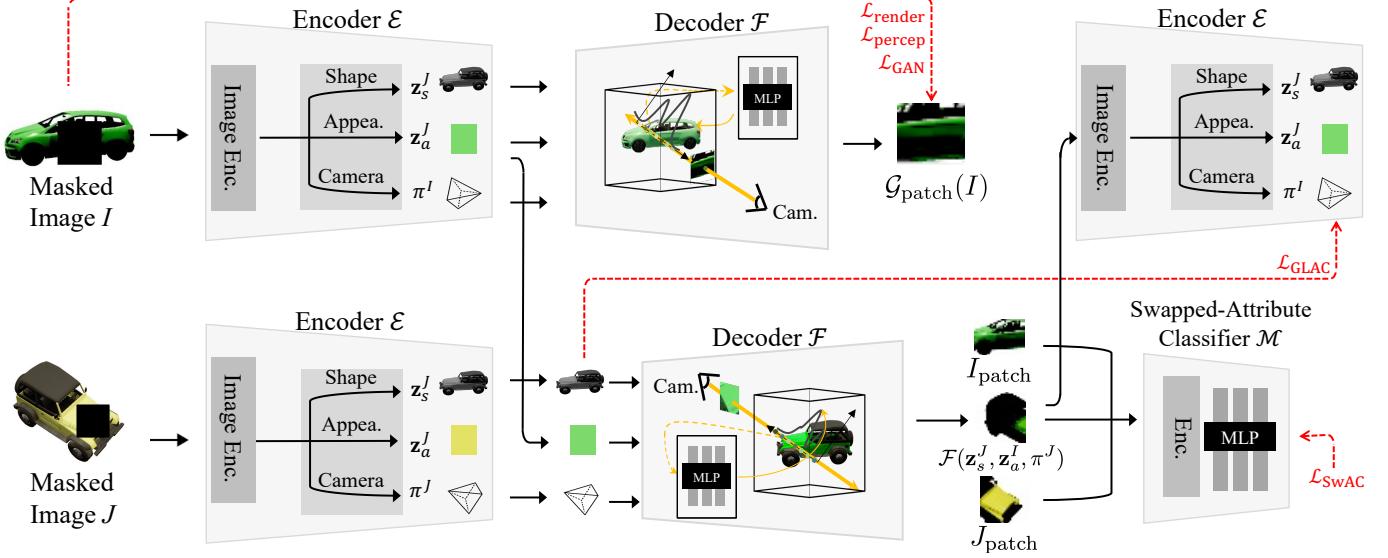[1]The viewing direction $\mathbf{d}$ is derived by estimated camera pose π.

Fig. 2: **Overall network configuration and loss functions of AE-NeRF.** Our network consists of the encoder $\mathcal{E}$ for extracting disentangled 3D attributes, including appearance code, shape code, and camera pose, and the decoder $\mathcal{F}$ for rendering the output image through disentangled generative NeRF. To train the networks, 2D rendering loss, perceptual loss, and adversarial loss are used. To improve the disentanglement, we also apply a global-local attribute consistency loss between the attributes from inputs and outputs. Moreover, by swapping the attributes across different instances, we boost the disentangling performance using swapped-attribute classifier.

2) Perceptual Reconstruction Loss.: To improve the generation quality, we further utilize a perceptual loss [71, 72] between the extracted features from an input image patch I_{patch} and a rendered patch $\mathcal{G}_{\text{patch}}(I)$ such that

$$\mathcal{L}_{\text{percep}}(I) = \sum_I \|\mathcal{R}(I_{\text{patch}}) - \mathcal{R}(\mathcal{G}_{\text{patch}}(I))\|_2, \quad (4)$$

where $\mathcal{R}$ represents VGG feature extractor [73]. Here, as [71] suggests, we compare the feature output of the first convolutional block of pretrained VGG-16 [73], extracted from input image and the generated patch.

3) Adversarial Loss.: To generate the photo-realistic output and apply losses on the novel views projected from another camera pose, we leverage an adversarial learning with a non-saturating GAN objective [74] and R_1 gradient penalty [75]:

$$\mathcal{L}_{\text{GAN}}(I, p) = \mathbb{E}_{I,p}[-\log(\mathcal{D}(\mathcal{G}_{\text{patch}}(I, p))) + \lambda\|\nabla(\mathcal{D}(\mathcal{G}_{\text{patch}}(I, p)))\|_1], \quad (5)$$

where $\mathcal{D}(\cdot)$ is a discriminator and p is a perturbation that is applied to latent attributes for better modelling an image manifold, which will be discussed in Sec. III-E. We apply the GAN loss to not only reconstructed patches, but also the generated patches from the combinations of swapped attributes.

4) Swapped-Attribute Classification Loss.: To improve the disentanglement, we first swap the estimated attributes across different instances I and J to render attribute-swapped images, and then distinguish which attribute-swapping is applied, e.g., appearance swap or shape swap. In specific, we design a swapped-attribute classifier $\mathcal{M}$ that takes I_{patch}, J_{patch}, and a rendered patch from swapped-attribute as input and outputs the probability whether the swapped attribute is either shape

or appearance. Even though there exist many possible combinations of swapped-attributes, we only consider appearance swap and shape swap from J to I, since estimating camera pose swap is notoriously challenging. We define a swapped-attribute classification loss such that

$$\mathcal{L}_{\text{SwAC}}(I, J) = \\ -\sum_{I,J} y\log(\mathcal{M}(I_{\text{patch}}, J_{\text{patch}}, \mathcal{F}(\mathbf{z}_s^J, \mathbf{z}_a^I, \pi^J))) \\ + (1 - y)\log(1 - \mathcal{M}(I_{\text{patch}}, J_{\text{patch}}, \mathcal{F}(\mathbf{z}_s^I, \mathbf{z}_a^J, \pi^J))), \quad (6)$$

where we arbitrarily set the label y as 0 when the shape attribute is swapped and set the label as 1 in the other case. During training, I_{patch} and J_{patch} are randomly sampled to maximize generalization ability. Supervised with this binary classification loss, the encoder can strictly differentiate the shape and appearance attributes.

5) Global-Local Attribute Consistency Loss.: To further improve the disentanglement ability of the network, we synthesize a patch from swapped attributes and pass it through the encoder, enforcing the output of encoder to be the same as the swapped attributes. In detail, given the rendered patch from swapped-attributes, e.g., $\mathcal{F}(\mathbf{z}_s^J, \mathbf{z}_a^I, \pi^J)$, we extract the attributes from the patch. We then constrain the output attributes to follow the original swapped attributes, which helps the encoder to boost the disentanglement performance and to relate the attributes from local patch and the corresponding attributes from global images. This can be interpreted as the local patch contains the attribute information from the two images that generate the swapped attribute pairs. Note that it is possible to extract the attribute from inputs with different resolutions by the same encoder as our encoder contains a global average pooling (GAP) module [68]. We define the
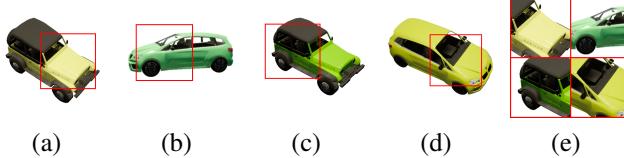
Fig. 3: **Intuitions of swapped-attribute classification loss:** (a), (b) input images, generated images by (c) appearance swap and (d) shape swap between (a) and (b), and (e) enlarged extracted patches. We present a swapped-attribute classification loss that classifies a set of patches from (a), (b), (c) and (a), (b), (d) to make the networks be aware of which attribute swap, which in turn helps to boost the disentanglement.

global-local attribute consistency loss in case of appearance swapping for image J as:

$$\mathcal{L}_{\text{GLAC}}(I, J) = \sum_{I,J} \mathcal{S}_{\text{latent}}(\{\mathbf{z}_s^J, \mathbf{z}_a^I\}, \mathcal{E}'(\mathcal{F}(\mathbf{z}_s^J, \mathbf{z}_a^I, \pi^I))),$$
(7)

where $\mathcal{E}'(I)$ outputs $\{\mathbf{z}_s^I, \mathbf{z}_a^I\}$ and $\mathcal{S}_{\text{latent}}(\mathbf{z}^I, \mathbf{z}^J) = \|\mathbf{z}^I - \mathbf{z}^J\|_2$. In this loss, we ignore the camera pose consistency since it is extremely challenging to estimate a camera pose from a local patch unlike appearance and shape codes. We swap shape attribute for image J by replacing $\mathbf{z}_s^J$ to $\mathbf{z}_s^I$ and $\mathbf{z}_a^J$ to $\mathbf{z}_s^I$.

E. Stage-Wise Training

Training the proposed auto-encoder model end-to-end from a scratch is extremely challenging due to inherent ambiguity of disentangled 3D attributes. To overcome this, we propose a novel stage-wise training scheme tailored for our auto-encoder architecture, described in the following.

1) Stage 1: Pre-training Decoder $\mathcal{F}$.: We first train the decoder $\mathcal{F}$. Before training an image-conditioned model, we first sample random latent vectors for appearance and shape codes, $\mathbf{z}_a$ and $\mathbf{z}_s$, from a Gaussian distribution such that $\mathbf{z}_a, \mathbf{z}_s \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and sample camera pose parameters π from uniform distribution on the upper hemisphere of neural radiance field, and then learn a volume renderer that renders a patch image I_{patch} from $\mathbf{z}_a$, $\mathbf{z}_s$ and π. Note that the patch center $\mathbf{u} = (u, v) \in \mathbb{R}^2$ and the scale $s \in \mathbb{R}+$ of the virtual $K \times K$ patch are randomly sampled from a uniform distribution. The total loss for stage 1 training is defined as

$$\mathcal{L}_{\text{stage}-1} = \mathcal{L}_{\text{GAN}}(I, p). \tag{8}$$

Since this loss function is learned with random latent codes $\mathbf{z}_a$, $\mathbf{z}_s$, and camera π, the perturbation p is inherently considered. After training, we render an image with learned $\mathcal{F}$ and use $\{\mathbf{z}_a, \mathbf{z}_s, \pi, I_{\text{pseudo}}\}$ as *pseudo* pairs for next stages.

2) Stage 2: Pre-training Encoder $\mathcal{E}$.: After stage 1, we now have the pre-trained decoder $\mathcal{F}$, and *pseudo* pairs $\{\mathbf{z}_a, \mathbf{z}_s, \pi, I_{\text{pseudo}}\}$. In stage 2, we train the encoder $\mathcal{E}$ that extracts shape, appearance codes and camera pose from an image given $\{\mathbf{z}_a, \mathbf{z}_s, \pi, I\}$. In addition, we use the pre-trained

decoder with its parameters fixed to better train the encoder. The total loss for stage 2 is defined as follows:

$$\begin{aligned}\mathcal{L}_{\text{stage}-2} =& \lambda_{\text{render}}\mathcal{L}_{\text{render}}(I_{\text{pseudo}}) + \lambda_{\text{percep}}\mathcal{L}_{\text{percep}}(I_{\text{pseudo}})\\ &+ \lambda_{\text{pseudo}}\mathcal{L}_{\text{pseudo}}(\mathcal{E}(I_{\text{pseudo}}), \mathbf{z}_a, \mathbf{z}_s, \pi),\end{aligned}$$
(9)

where $\mathcal{L}_{\text{pseudo}}$ is defined such that

$$\begin{aligned}\mathcal{L}_{\text{pseudo}}(\mathcal{E}(I_{\text{pseudo}}), \mathbf{z}_a, \mathbf{z}_s, \pi) =& \|\mathbf{z}_a^{I_{\text{P}}} - \mathbf{z}_a\|_2 + \|\mathbf{z}_s^{I_{\text{P}}} - \mathbf{z}_s\|_2\\ &+ \mathcal{L}_{\text{cam}}(\pi^{I_{\text{P}}}, \pi),\end{aligned}$$
(10)

where $\{\mathbf{z}_a^{I_{\text{P}}}, \mathbf{z}_s^{I_{\text{P}}}, \pi^{I_{\text{P}}}\} = \mathcal{E}(I_{\text{pseudo}})$ and $\mathcal{L}_{\text{cam}}(\pi^{I_{\text{P}}}, \pi) = \|\mathbf{s}^{I_{\text{P}}} - \mathbf{s}\|_2 + \|\mathbf{t}^{I_{\text{P}}} - \mathbf{t}\|_2 + \|(\mathbf{R}^{I_{\text{P}}})^{-1}\mathbf{R} - \mathbf{I}_{3\times3}\|_2$, with $\mathbf{s}^{I_{\text{P}}}, \mathbf{t}^{I_{\text{P}}}$, and $\mathbf{R}^{I_{\text{P}}}$ are the predicted camera parameters, respectively, with scale $\mathbf{s} \in \mathbb{R}^1$, translation $\mathbf{t} \in \mathbb{R}^3$, and rotation matrix $\mathbf{R} \in \mathbb{R}^{3\times3}$. We use an identity matrix $\mathbf{I}_{3\times3}$ for the rotation matrix loss. λ_{render}, λ_{percep}, and λ_{pseudo} are the weight parameters for corresponding losses.

3) Stage 3: Fine-tuning Encoder $\mathcal{E}$.: After stage 2, we now have the pre-trained encoder $\mathcal{E}$ and the decoder $\mathcal{F}$, which can be an auto-encoder themselves. Now we have the pre-trained encoder $\mathcal{E}$ and the decoder $\mathcal{F}$ after stage 2, which can be an auto-encoder themselves. However, the disentanglement performance still depends on the architectural designs of the decoder. In addition, the encoder $\mathcal{E}$ only takes a pseudo label, which is a generated image itself, and thus, it may be sub-optimal to handle a real image. To overcome these, in stage 3, we use all the proposed loss functions including global-local attribute consistency loss as $\mathcal{L}_{\text{GLAC}}$ and swapped-attribute classification loss $\mathcal{L}_{\text{SwAC}}$.

The total loss for stage 3 training is defined as

$$\begin{aligned}\mathcal{L}_{\text{stage}-3} =& \lambda_{\text{GAN}}\mathcal{L}_{\text{GAN}}(I) + \lambda_{\text{render}}\mathcal{L}_{\text{render}}(I)\\ &+ \lambda_{\text{percep}}\mathcal{L}_{\text{percep}}(I) + \lambda_{\text{GLAC}}\mathcal{L}_{\text{GLAC}}(I, J)\\ &+ \lambda_{\text{SwAC}}\mathcal{L}_{\text{SwAC}}(I, J).\end{aligned}$$
(11)

where λ_{GLAC} and λ_{SwAC} are the weight parameters for corresponding losses.

IV. EXPERIMENTS

A. Implementation Details

Our decoder architecture follows the default setting for GRAF [27]. To extract feature from a given input, we use a ResNet-18 model [68] pre-trained on ImageNet. We dropped the last fully connected layer and instead attached 2 fully connected layers followed by shape, appearance, and camera pose estimators. For the shared 2 layers we use batch normalization and ReLU activation and its final activation as LeakyReLU. We use a PatchGAN discriminator, following Pix2Pix [76]. We sample 1,024 rays within the bounding box in an image. We use CNNs with additional fully connected layers for a swapped-attribute binary classifier. The encoder predict 6 dimensional vector for camera pose regression, following the representation of 6D rotation [77]. Camera translation vector is extracted from the rotation matrix, identical to the implementation of GRAF [27].

Fig. 4: **Examples of proposed dataset for measuring disentanglement quality.** Given pairs of images (col. 1-2, 4-5, 7-8), we set the image with swapped attribute as a ground-truth image for attribute-swap; shape-swapped (col. 3), appearance-swapped (col. 6), and camera-swapped (col. 9).

Fig. 5: **Comparison of auto-encoding results** of AE-NeRF with SA [8], EditNeRF [30], and CodeNeRF [29] on CARLA [78] benchmark.

Fig. 6: **Auto-encoding results** of AE-NeRF compared with CLIP-NeRF [31] on CARLA [78] and Photoshapes [79] benchmark.
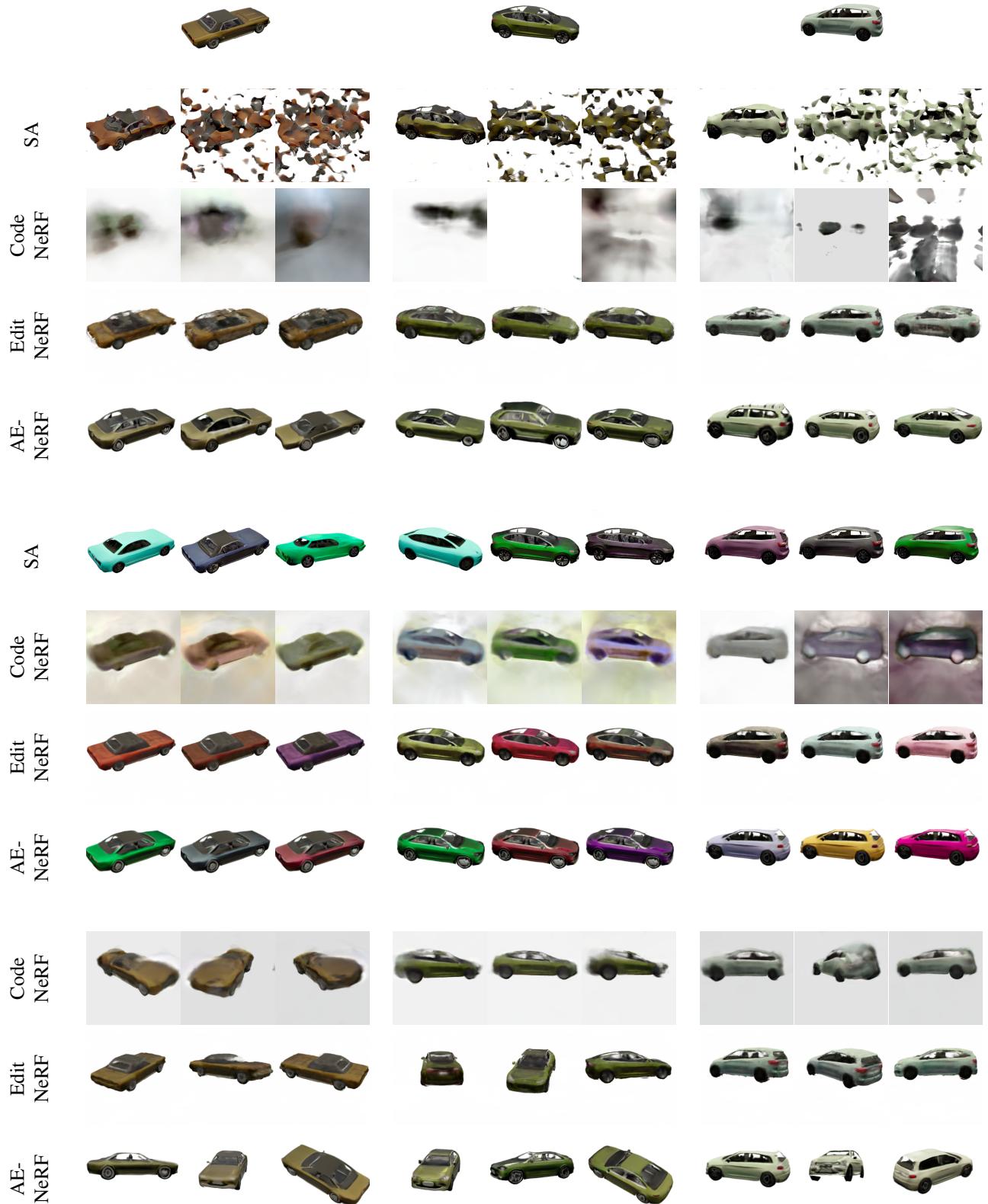
Fig. 7: **Random perturbation comparison:** results of AE-NeRF with SA [8], EditNeRF [30], and CodeNeRF [29] on CARLA [78] benchmark of applying random perturbations on shape (row 2-5), appearance codes (row 6-9) and camera (row 10-12) respectively while maintaining other attributes consistent.

Fig. 8: **Qualitative comparison on swapping disentangled shape and appearance on CARLA [80] benchmark.** Given images (row 1) as a pair of input images, we compare the attribute swapping results in the order of shape, appearance, and camera pose: SA [8] (row 2), CodeNeRF [29] (row 3), EditNeRF [30] (row 4), and AE-NeRF (row 5).
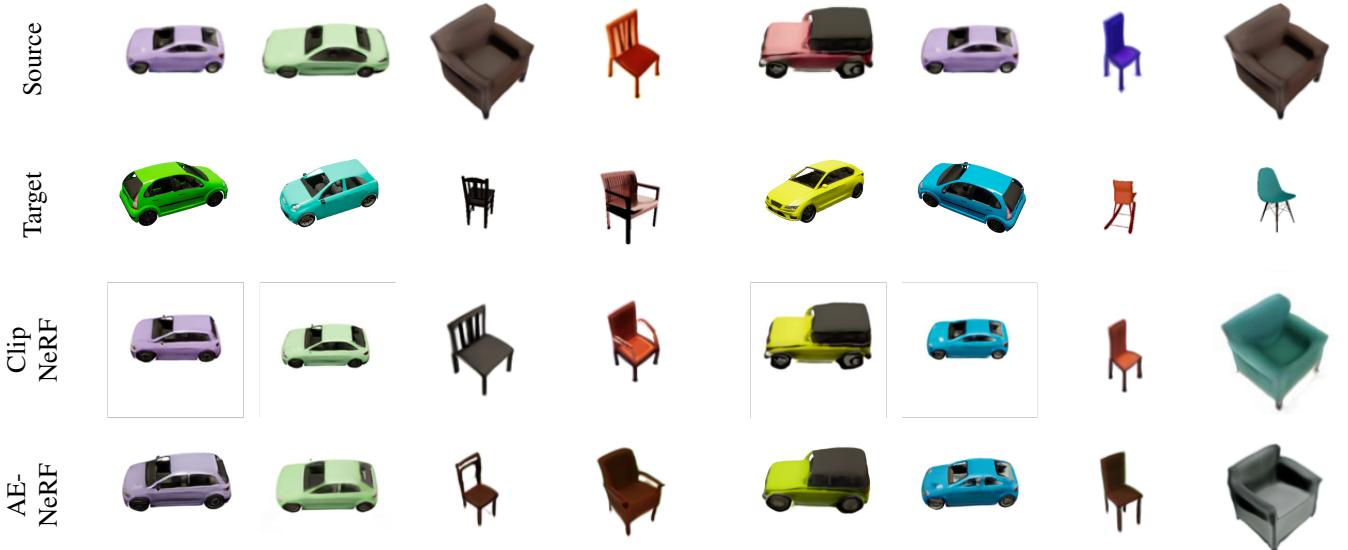

Fig. 9: **Code swapping comparison with ClipNeRF on CARLA [78] and Photoshapes [79] dataset.**

B. Experimental Settings

1) Datasets: We evaluate our method on two publicly available datasets of varying complexity: `cars` from GRAF CARLA dataset [78] and `chairs` are images from Photoshapes dataset [79], following the rendering protocol of [81]. CARLA contains 10K synthetic car images which are 18 different kinds of car models in different rendering viewpoints and colors with resolution 256×256. Photo contains 150k images with resolution 256×256.
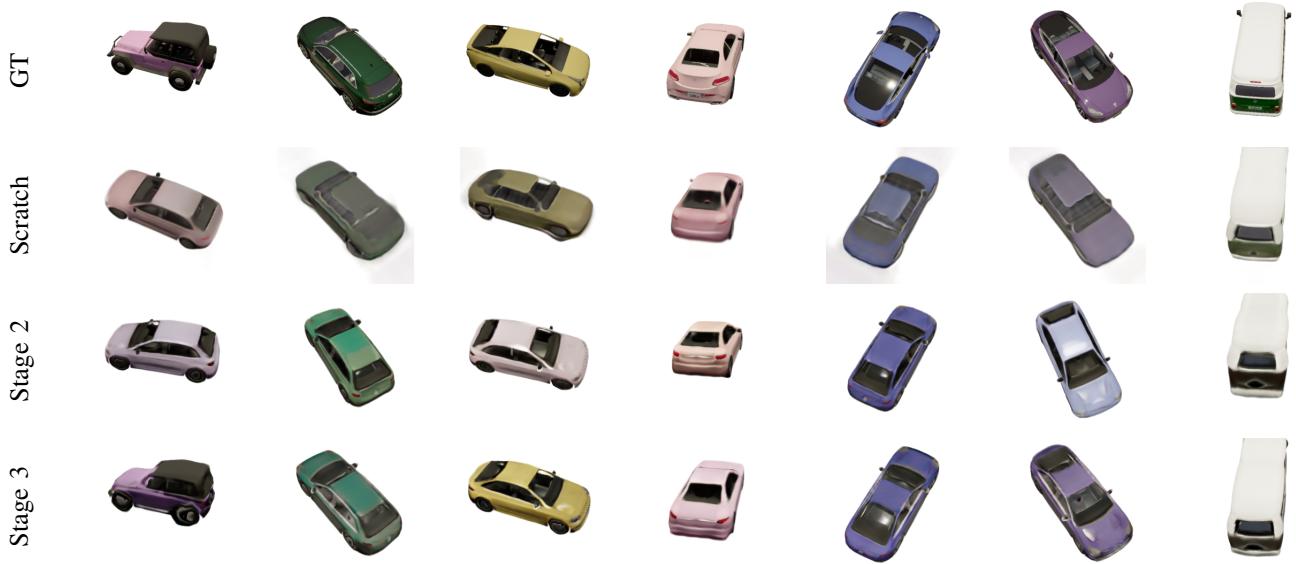
2) Test Dataset for Quantitative Evaluation: In addition, there is no public dataset to quantitatively measure the model performance of disentangling attributes from given images and 3D-aware manipulation quality, we build a new test subset of CARLA [78], where there are pairs of images that share one of such attributes, as exemplified in Fig. 4.

3) Evaluation Metric: To measure the model performance of disentangling 3D attributes from a given image, we use Fréchet Inception Distance (FID) score [82] to measure the degree of fidelity for generated images on 1k dataset of CARLA [78] whose optimized style codes are offered by EditNeRF [30].

Fig. 10: **Comparison of attribute interpolation and extrapolation between (EditNeRF [30] (1-4, 9-12 rows) and AE-NeRF (5-8, 13-16 rows)**. We manipulate shape, appearance, camera viewpoint attribute and the combination of the three between two images. Given two different images, our AE-NeRF smoothly interpolates the generated images within real data distribution as well as disentangles the shape, appearance, and camera poses, while EditNeRF [30] often fails.

Fig. 11: **Results of stage-wise training:** Given a row of input image GT, following rows are results from different learning framework, in terms of stage-wise training. Direct training from scratch tends to generates blurry images, and frequently fails in reconstruction. Comparing stage2 and stage3, the latter represent better quality on predicting suitable shape and appearance code for reconstruction.

TABLE II: **Quantitative evaluation of 3D reconstruction using metric evaluation on CARLA dataset [78].** The best result is shown in bold, and the second best is underlined.

	FID
SA [29]	<u>58.31</u>
CodeNeRF [29]	128.82
EditNeRF [30]	76.63
AE-NeRF	**42.92**

C. Reconstruction

We quantitatively evaluate the image quality and accuracy of auto-encoding results using the Frechet Inception Distances (FID) [82] by reconstructing randomly selected 1K images for CARLA Dataset [78] and 1K images for Photoshapes [79] Dataset. Quantitative and qualitative results showed in Table II and Fig. 5 prove that we have significantly higher results compared to previous 3D attribute controlling models such as CodeNeRF [29] and EditNeRF [83]. More importantly, as our encoder first learns to map the distribution of generative models and then optimizes the output in an instance-wise manner, the accurate reconstruction with high-fidelity is enabled. We also demonstrate the qualitative results of CLIP-NeRF [31] in a GAN-inversion manner in Fig. 9. Note that at test time the CLIP-NeRF additionally optimizes the model for a given input while AE-NeRF directly predicts the output.

D. Image Manipulation for a Single Image

In this section, we first evaluate 3D-aware attribute manipulation from a single image. We prove that the proposed method disentangles each attribute of the given image by injecting perturbations to each attribute separately. The results in Fig. 7

TABLE III: **Comparison of AE-NeRF to SA [8], CodeNeRF [29], EditNeRF [30] on code swapping results with CARLA [78].** We evaluate the swapping attribute results of a source object instance to match a target instance. The best result is shown in bold, and the second best is underlined.

	Percep.			FID		
	Sh.	App.	Cam.	Sh.	App.	Cam.
SA [8]	0.2390	0.2831	-	<u>58.32</u>	<u>57.26</u>	-
CodeNeRF [29]	0.3421	0.3311	0.3212	102.5	113.21	127.52
EditNeRF [30]	<u>0.1912</u>	**0.1142**	**0.1641**	97.27	77.84	<u>97.95</u>
AE-NeRF	**0.1731**	<u>0.1948</u>	<u>0.1691</u>	**42.88**	**41.97**	**42.29**

show that our model is aware of each attribute and thus separately manipulate the attribute very well while maintaining the others more consistently than compared methods [8, 30, 29]. In addition, since EditNeRF [30] and CodeNeRF [29] optimize the radiance field focused on given camera pose, 3D attributes are disentangled based on their viewpoint. This property discourages EditNeRF and CodeNeRF to generate realistic images using disentangled attributes which are insufficient for novel viewpoints. On the other hand, AE-NeRF can disentangle 3D attributes regardless of viewpoints by using stage 1 (adversarial learning) where the model is focused on synthesize realistic images from any latent attributes or viewpoints. This ensures the high fidelity of generated images and consecutively reconstructed images and even synthesized images from novel viewpoints.

E. Image Manipulation Across Multiple Images

We then compare our 3D-aware image manipulation across multiple images, e.g., swapping appearance, shape, or camera
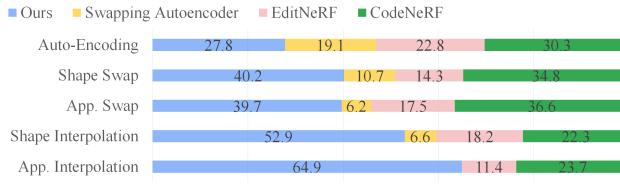
Fig. 12: **User study results**, which is conducted in respect of auto-encoding quality, disentanglement quality, and interpolation ability. Compared to conventional methods [30, 8, 29], our AE-NeRF ranks the first in every cases.

pose across different instances on CARLA [30] dataset as in EditNeRF [30].

In addition, as shown in Table III, we evaluate our model by swapping the attribute codes, where given a pair of source and target images the source image is manipulated by replacing an extracted certain attribute code from target image, either shape, appearance or camera pose, so that the source image eventually obtains the attribute of target image. We measure the code swapping performance on proposed test set as a ground truth pair. Comparing functional aspects, EditNeRF [30] manipulates the radiance field given user's editing scribbles by only optimization procedures. This is rather time-consuming for manipulating overall texture or shape of given instance. Furthermore, EditNeRF optimize their radiance field only in the editor's viewpoint, giving inconsistent manipulation result from novel viewpoint. AE-NeRF on the other hand, since the latent features are provided by trained encoder, manipulation of the objects' overall shape and texture is relatively faster than the optimization-based methods.

F. User Study

We conducted user study on 300 participants to compare the synthesized results obtained from different models. First, we obtained 10 reconstructed images from SA [8], EditNeRF [30], CodeNeRF [29], and ours. We showed them to respondents and asked them to indicate 10 images they prefer in aspect of the image reconstruction quality with high-fidelity. Second, to compare disentanglement ability of 3 models and ours, we synthesized 10 images with each attributes swapped. Then we asked subjects to choose 10 images that show preferable disentanglement quality. Finally, we generated 10 images with attribute interpolation from 3 models and ours. We asked participants to pick 10 images that show the most well interpolated images. According to the user study displayed in Fig. 12, we could identify that our model is preferred more frequently than any other models, and outperforms SA [8], CodeNeRF [29] and EditNeRF [30] in terms of controllability.

G. Ablation Study

1) Effectiveness of Stage-wise Training.: Table IV and Fig. 11 show the effectiveness of stage-wise training. As described in Sec. III-E, training the proposed auto-encoder model from scratch (Table IV (**I**), Fig. 11 (**Scratch**)) is extremely challenging and easy to get stuck in local minimum. Thus, we propose a stage-wise training, where we pre-train the decoder

TABLE IV: **Ablation study on stage-wise training.** The best result is shown in bold.

Stage		FID		
		Sh.	App.	Cam.
(**I**)	Baseline (Stage 3)	136.49	139.32	137.13
(**II**)	Stage 1, 2	47.14	49.43	46.77
(**III**)	Stage 1, 3	46.52	47.79	47.37
(**IV**)	Ours (Stage 1, 2, 3)	**42.88**	**41.97**	**42.29**

TABLE V: **Ablation study on effectiveness of loss functions.** The best result is shown in bold.

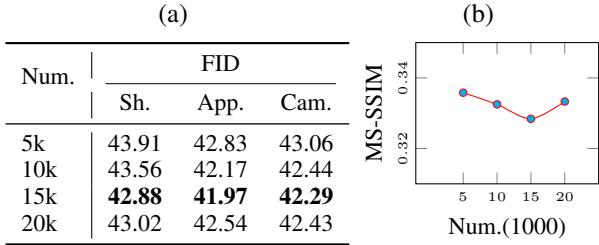| Loss combination		FID		
		Sh.	App.	Cam.
(**I**)	$\mathcal{L}_{\text{render}} + \mathcal{L}_{\text{stage}-2}$	50.18	50.87	49.43
(**II**)	(I) + $\mathcal{L}_{\text{GAN}}$	47.14	49.43	46.77
(**III**)	(II) + $\mathcal{L}_{\text{GLAC}}$	45.64	45.24	44.57
(**IV**)	(II) + $\mathcal{L}_{\text{SwAC}}$	47.46	45.49	46.44
(**V**)	(II) + $\mathcal{L}_{\text{GLAC}} + \mathcal{L}_{\text{SwAC}}$	**42.88**	**41.97**	**42.29**

in stage 1 to generate pseudo labels used for encoder training in stage 2, followed by stage 3 with losses for disentanglement. We figured out that the performance of our model gets boosted when the model is supervised with *pseudo data* as this strongly enforce the model to map the given image and the generated attributes. The comparison of Table IV (**III**), (**IV**) proves that mapping the output distribution of encoder with pseudo labels highly contributes in performance improvement.

In addition, Table IV (**III**) depends on its disentanglement ability on the structural architecture of decoder, we propose additional losses that raise the disentanglement performance of our model as detailed in Sec. III-D. This can be confirmed by comparing Table IV (**II**), (**IV**).

2) Effectiveness of Each Loss.: Here, we evaluated effectiveness of each loss function compared to the basic loss for the auto-encoder, i.e., $\mathcal{L}_{\text{render}}$ and $\mathcal{L}_{\text{stage}-2}$, as shown in Table V. Note that, with $\mathcal{L}_{\text{cam}}$ that belongs to the $\mathcal{L}_{\text{p}}$ on *pseudo data*, our model is capable of extracting the camera pose given a single image. $\mathcal{L}_{\text{GAN}}$ helps the model to synthesize images with fine details such as object's surface and edge. The additional $\mathcal{L}_{\text{GLAC}}$ raises the disentanglement performance with understanding of consistent object attribute in generated local patch from swapped attributes. With additional convolutional binary classifier, $\mathcal{L}_{\text{SwAC}}$ improves the disentanglement performance of our model by supervising our encoder with more accurate paired attribute-swapped image.

3) Effectiveness of The Number of Pseudo Data.: We evaluated how the number of generated pseudo data affects the final model performance. Diverse set of images of high fidelity, with a high variance in the pseudo shape and appearance codes, is essential for supervising the encoder in stage 2. Based on this idea, we first sampled 5k to 20k images and compared the diversity by the mean of MS-SSIM [84] which measures the similarity across all pseudo images generated in stage 1. In this study, We regard the high MS-SSIM as low diversity across images. Note that increasing the number

TABLE VI: **Ablation study on the number of pseudo data.** (a) compares the quality of feature swapped images while (b) shows the relationship between the numbers of pseudo data and the corresponding data diversity. Data diversity does not correlate with the number of pseudo data, and 15K samples is the optimal number of pseudo data used to train the network.

(a)				(b)

Num.	FID			
	Sh.	App.	Cam.
5k	43.91	42.83	43.06
10k	43.56	42.17	42.44
15k	**42.88**	**41.97**	**42.29**
20k	43.02	42.54	42.43

of pseudo data does not always guarantee the data diversity proportionally, described in Table VI (right). Then we measure Fréchet Inception Distance (FID) score [82] of the final model performance depending on different numbers of pseudo data. As illustrated in Table VI, the model achieves better performance with more pseudo data.

V. CONCLUSION

In this paper, we have presented, for the first time, an auto-encoder architecture based on conditional NeRF for 3D-aware object manipulation, dubbed Auto-Encoding Neural Radiance Fields (AE-NeRF). We designed our model consisting of the encoder to extract disentangled 3D attributes from an image, and the decoder to render an output image through disentangled generative NeRF, which is well tailored for 3D-aware object manipulation. To improve the disentanglement, we present two losses, global-local attribute consistency loss and swapped-attribute classification loss. In addition, since training the networks from a scratch is extremely challenging, we presented a stage-wise training scheme, which dramatically helps to boost the performance. We have shown that our method surpasses other state-of-the-art in several benchmarks and manipulation tasks. Moreover, we have conducted extensive ablation studies to validate our design choices.

REFERENCES

[1] S. Goel, A. Kanazawa, and J. Malik, "Shape and viewpoint without keypoints," in *European Conference on Computer Vision.* Springer, 2020, pp. 88–104.

[2] N. Kulkarni, A. Gupta, and S. Tulsiani, "Canonical surface mapping via geometric cycle consistency," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 2202–2211.

[3] R. Hu, N. Ravi, A. C. Berg, and D. Pathak, "Worldsheet: Wrapping the world in a 3d sheet for view synthesis from a single image," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021, pp. 12 528–12 537.

[4] A. Liu, R. Tucker, V. Jampani, A. Makadia, N. Snavely, and A. Kanazawa, "Infinite nature: Perpetual view generation of natural scenes from a single image," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021.

[5] G. Riegler and V. Koltun, "Free view synthesis," in *European Conference on Computer Vision*, 2020.

[6] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4401–4410.

[7] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz, "Multimodal unsupervised image-to-image translation," in *ECCV*, 2018.

[8] T. Park, J.-Y. Zhu, O. Wang, J. Lu, E. Shechtman, A. A. Efros, and R. Zhang, "Swapping autoencoder for deep image manipulation," in *Advances in Neural Information Processing Systems*, 2020.

[9] A. Kanazawa, M. J. Black, D. W. Jacobs, and J. Malik, "End-to-end recovery of human shape and pose," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7122–7131.

[10] A. Kanazawa, S. Tulsiani, A. A. Efros, and J. Malik, "Learning category-specific mesh reconstruction from image collections," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 371–386.

[11] A. Bhattad, A. Dundar, G. Liu, A. Tao, and B. Catanzaro, "View generalization for single image textured 3d models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 6081–6090.

[12] H. Kato, Y. Ushiku, and T. Harada, "Neural 3d mesh renderer," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 3907–3916.

[13] W. Chen, H. Ling, J. Gao, E. Smith, J. Lehtinen, A. Jacobson, and S. Fidler, "Learning to predict 3d objects with an interpolation-based differentiable renderer," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32. Curran Associates, Inc., 2019.

[14] S. Liu, T. Li, W. Chen, and H. Li, "Soft rasterizer: A differentiable renderer for image-based 3d reasoning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 7708–7717.

[15] G. Loubet, N. Holzschuch, and W. Jakob, "Reparameterizing discontinuous integrands for differentiable rendering," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 6, pp. 1–14, 2019.

[16] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger, "Occupancy networks: Learning 3d reconstruction in function space," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4460–4470.

[17] M. Oechsle, L. Mescheder, M. Niemeyer, T. Strauss, and A. Geiger, "Texture fields: Learning texture representations in function space," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[18] M. Niemeyer, L. Mescheder, M. Oechsle, and A. Geiger, "Occupancy flow: 4d reconstruction by learning particle dynamics," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[19] S. Peng, M. Niemeyer, L. Mescheder, M. Pollefeys, and A. Geiger, "Convolutional occupancy networks," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16.* Springer, 2020, pp. 523–540.

[20] M. Niemeyer, L. Mescheder, M. Oechsle, and A. Geiger, "Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 3504–3515.

[21] V. Sitzmann, M. Zollhöfer, and G. Wetzstein, "Scene representation networks: Continuous 3d-structure-aware neural scene representations," in *Advances in Neural Information Processing Systems*, 2019.

[22] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," in *European conference on computer vision.* Springer, 2020, pp. 405–421.

[23] A. Yu, V. Ye, M. Tancik, and A. Kanazawa, "pixelnerf: Neural radiance fields from one or few images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 4578–4587.

[24] A. Jain, M. Tancik, and P. Abbeel, "Putting nerf on a diet: Semantically consistent few-shot view synthesis," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5885–5894.

[25] J. Chibane, A. Bansal, V. Lazova, and G. Pons-Moll, "Stereo radiance fields (srf): Learning view synthesis for sparse views of novel scenes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 7911–7920.

[26] A. Trevithick and B. Yang, "Grf: Learning a general radiance field for 3d scene representation and rendering," *arXiv preprint arXiv:2010.04595*, 2020.

[27] K. Schwarz, Y. Liao, M. Niemeyer, and A. Geiger, "Graf: Generative radiance fields for 3d-aware image synthesis," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[28] M. Niemeyer and A. Geiger, "Giraffe: Representing scenes as compositional generative neural feature fields," in *Proceedings of the IEEE/CVF*

Conference on Computer Vision and Pattern Recognition, 2021, pp. 11 453–11 464.

[29] W. Jang and L. Agapito, "Codenerf: Disentangled neural radiance fields for object categories," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 12 949–12 958.

[30] S. Liu, X. Zhang, Z. Zhang, R. Zhang, J.-Y. Zhu, and B. Russell, "Editing conditional radiance fields," in *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021.

[31] C. Wang, M. Chai, M. He, D. Chen, and J. Liao, "Clip-nerf: Text-and-image driven manipulation of neural radiance fields," *arXiv preprint arXiv:2112.05139*, 2021.

[32] J. Zhu, Y. Shen, D. Zhao, and B. Zhou, "In-domain gan inversion for real image editing," in *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XVII*, ser. Lecture Notes in Computer Science, A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds., vol. 12362. Springer, 2020, pp. 592–608. [Online]. Available: https://doi.org/10.1007/978-3-030-58520-4_35

[33] W. Xia, Y. Zhang, Y. Yang, J.-H. Xue, B. Zhou, and M.-H. Yang, "Gan inversion: A survey," *arXiv preprint arXiv: 2101.05278*, 2021.

[34] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[35] K. Jo, G. Shim, S. Jung, S. Yang, and J. Choo, "Cg-nerf: Conditional generative neural radiance fields," *CoRR*, vol. abs/2112.03517, 2021. [Online]. Available: https://arxiv.org/abs/2112.03517

[36] C.-H. Lin, C. Kong, and S. Lucey, "Learning efficient point cloud generation for dense 3d object reconstruction," in *proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[37] P. Mandikal and V. B. Radhakrishnan, "Dense 3d point cloud reconstruction using a deep pyramid network," in *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2019, pp. 1052–1060.

[38] T. Groueix, M. Fisher, V. G. Kim, B. C. Russell, and M. Aubry, "A papier-mâché approach to learning 3d surface generation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 216–224.

[39] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang, "Pixel2mesh: Generating 3d mesh models from single rgb images," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 52–67.

[40] S. R. Richter and S. Roth, "Matryoshka networks: Predicting 3d geometry via nested shape layers," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 1936–1944.

[41] V. Sitzmann, J. Thies, F. Heide, M. Nießner, G. Wetzstein, and M. Zollhofer, "Deepvoxels: Learning persistent 3d feature embeddings," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2437–2446.

[42] W. Liu, Z. Piao, J. Min, W. Luo, L. Ma, and S. Gao, "Liquid warping gan: A unified framework for human motion imitation, appearance transfer and novel view synthesis," in *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[43] X. Li, S. Liu, K. Kim, S. De Mello, V. Jampani, M.-H. Yang, and J. Kautz, "Self-supervised single-view 3d reconstruction via semantic consistency," in *European Conference on Computer Vision*. Springer, 2020, pp. 677–693.

[44] Y. Jiang, D. Ji, Z. Han, and M. Zwicker, "Sdfdiff: Differentiable rendering of signed distance fields for 3d shape optimization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1251–1261.

[45] R. Martin-Brualla, N. Radwan, M. S. Sajjadi, J. T. Barron, A. Dosovitskiy, and D. Duckworth, "Nerf in the wild: Neural radiance fields for unconstrained photo collections," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 7210–7219.

[46] K. Park, U. Sinha, J. T. Barron, S. Bouaziz, D. B. Goldman, S. M. Seitz, and R. Martin-Brualla, "Nerfies: Deformable neural radiance fields," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5865–5874.

[47] A. Pumarola, E. Corona, G. Pons-Moll, and F. Moreno-Noguer, "D-nerf: Neural radiance fields for dynamic scenes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 10 318–10 327.

[48] S. Zhi, T. Laidlow, S. Leutenegger, and A. J. Davison, "In-place scene labelling and understanding with implicit scene representation," *arXiv preprint arXiv:2103.15875*, 2021.

[49] V. Sitzmann, J. N. Martel, A. W. Bergman, D. B. Lindell, and G. Wetzstein, "Implicit neural representations with periodic activation functions," *arXiv preprint arXiv:2006.09661*, 2020.

[50] J. Gu, L. Liu, P. Wang, and C. Theobalt, "Stylenerf: A style-based 3d-aware generator for high-resolution image synthesis," *arXiv preprint arXiv:2110.08985*, 2021.

[51] P. Zhou, L. Xie, B. Ni, and Q. Tian, "Cips-3d: A 3d-aware generator of gans based on conditionally-independent pixel synthesis," *arXiv preprint arXiv:2110.09788*, 2021.

[52] Y. Deng, J. Yang, J. Xiang, and X. Tong, "Gram: Generative radiance manifolds for 3d-aware image generation," *arXiv preprint arXiv:2112.08867*, 2021.

[53] E. R. Chan, C. Z. Lin, M. A. Chan, K. Nagano, B. Pan, S. De Mello, O. Gallo, L. Guibas, J. Tremblay, S. Khamis *et al.*, "Efficient geometry-aware 3d generative adversarial networks," *arXiv preprint arXiv:2112.07945*, 2021.

[54] D. Bhattacharjee, S. Kim, G. Vizier, and M. Salzmann, "Dunit: Detection-based unsupervised image-to-image translation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[55] D. Kotovenko, A. Sanakoyeu, S. Lang, and B. Ommer, "Content and style disentanglement for artistic style transfer," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 4422–4431.

[56] W. Wu, K. Cao, C. Li, C. Qian, and C. C. Loy, "Transgaga: Geometry-aware unsupervised image-to-image translation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8012–8021.

[57] Y.-J. Li, C.-S. Lin, Y.-B. Lin, and Y.-C. F. Wang, "Cross-dataset person re-identification via unsupervised pose disentanglement and adaptation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 7919–7929.

[58] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 8110–8119.

[59] H. Kazemi, S. M. Iranmanesh, and N. Nasrabadi, "Style and content disentanglement in generative adversarial networks," in *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2019, pp. 848–856.

[60] P. Esser, J. Haux, and B. Ommer, "Unsupervised robust disentangling of latent characteristics for image synthesis," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 2699–2709.

[61] K. Kania, K. M. Yi, M. Kowalski, T. Trzcinski, and A. Tagliasacchi, "Conerf: Controllable neural radiance fields," *CoRR*, vol. abs/2112.01983, 2021. [Online]. Available: https://arxiv.org/abs/2112.01983

[62] J. Zhu, Y. Shen, D. Zhao, and B. Zhou, "In-domain gan inversion for real image editing," in *European conference on computer vision*. Springer, 2020, pp. 592–608.

[63] J.-Y. Zhu, Z. Zhang, C. Zhang, J. Wu, A. Torralba, J. Tenenbaum, and B. Freeman, "Visual object networks: Image generation with disentangled 3d representations," in *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31. Curran Associates, Inc., 2018.

[64] H.-Y. Lee, H.-Y. Tseng, J.-B. Huang, M. K. Singh, and M.-H. Yang, "Diverse image-to-image translation via disentangled representations," in *European Conference on Computer Vision*, 2018.

[65] Q. Mao, H.-Y. Lee, H.-Y. Tseng, S. Ma, and M.-H. Yang, "Mode seeking generative adversarial networks for diverse image synthesis," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[66] H.-Y. Lee, H.-Y. Tseng, Q. Mao, J.-B. Huang, Y.-D. Lu, M. Singh, and M.-H. Yang, "Drit++: Diverse image-to-image translation via disentangled representations," *International Journal of Computer Vision*, vol. 128, no. 10, pp. 2402–2417, 2020.

[67] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz, "Multimodal unsupervised image-to-image translation," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 172–189.

[68] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[69] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," 2021.

[70] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly,

J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," 2021.

[71] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *European conference on computer vision*. Springer, 2016, pp. 694–711.

[72] J. Zhu, Y. Shen, D. Zhao, and B. Zhou, "In-domain gan inversion for real image editing," 2020.

[73] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015.

[74] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[75] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved training of wasserstein gans," *arXiv preprint arXiv:1704.00028*, 2017.

[76] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1125–1134.

[77] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li, "On the continuity of rotation representations in neural networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[78] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Conference on robot learning*. PMLR, 2017, pp. 1–16.

[79] K. Park, K. Rematas, A. Farhadi, and S. M. Seitz, "Photoshape: Photorealistic materials for large-scale shape collections," *ACM Trans. Graph.*, vol. 37, no. 6, Nov. 2018.

[80] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su *et al.*, "Shapenet: An information-rich 3d model repository," *arXiv preprint arXiv:1512.03012*, 2015.

[81] M. Oechsle, M. Niemeyer, C. Reiser, L. Mescheder, T. Strauss, and A. Geiger, "Learning implicit surface light fields," in *2020 International Conference on 3D Vision (3DV)*, 2020, pp. 452–462.

[82] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017.

[83] E. Collins, R. Bala, B. Price, and S. Susstrunk, "Editing in style: Uncovering the local semantics of gans," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 5771–5780.

[84] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier GANs," in *Proceedings of the 34th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, D. Precup and Y. W. Teh, Eds., vol. 70. PMLR, 06–11 Aug 2017, pp. 2642–2651.