Likelihood-Based Generative Radiance Field with Latent Space Energy-Based Model for 3D-Aware Disentangled Image Representation

Yaxuan Zhu
Department of Statistics
University of California - UCLA
8125 Math Sciences Bldg, Los Angeles, CA, USA
yaxuanzhu@ucla.edu

Jianwen Xie, Ping Li
Cognitive Computing Lab
Baidu Research
10800 NE 8th St. Bellevue WA 98004, USA
{jianwen.kenny, pingli98}@gmail.com

Abstract

[1]We propose the NeRF-LEBM, a likelihood-based top-down 3D-aware 2D image generative model that incorporates 3D representation via Neural Radiance Fields (NeRF) and 2D imaging process via differentiable volume rendering. The model represents an image as a rendering process from 3D object to 2D image and is conditioned on some latent variables that account for object characteristics and are assumed to follow informative trainable energy-based prior models. We propose two likelihood-based learning frameworks to train the NeRF-LEBM: (i) maximum likelihood estimation with Markov chain Monte Carlo-based inference and (ii) variational inference with the reparameterization trick. We study our models in the scenarios with both known and unknown camera poses. Experiments on several benchmark datasets demonstrate that the NeRF-LEBM can infer 3D object structures from 2D images, generate 2D images with novel views and objects, learn from incomplete 2D images, and learn from 2D images with known or unknown camera poses.

1 Introduction

1.1 Motivation

Towards the goal of 3D-aware image synthesis, existing methods generate 3D representations of objects either in a voxel-based format [55] or via intermediate 3D features [1],

---

[1]The work of Yaxun Zhu was conducted as a research intern at Cognitive Compuitng Lab, Baidu Research – Bellevue, WA, USA

---

and then use the differentiable rendering operation to render the generated 3D object into 2D views. However, the voxel-based 3D representation is discrete and memory-inefficient so that the methods are limited to generating low-quality and low-resolution images of objects. Notably, the Neural Radiance Field (NeRF) [25] has become a new type of 3D representation of objects and shown impressive results for new view synthesis. It represents a continuous 3D scene or object by a mapping function parameterized by a neural net, which takes as input a 3D location and a viewing direction and outputs the values of color and density. The visualization of the 3D object can be achieved through generating different views of images by querying the mapping function at each specific 3D location and viewing direction, followed by volume rendering operation [17] to produce image pixel intensities. In general, each NeRF function can only represent one single object and need to be trained from multiple views of images of that object. By generalizing the original NeRF function to a conditional version that involves latent variables that account for the appearance and shape of the object, GRAF [36] builds a 2D image generator based on the conditional NeRF and trains the generator for 3D-aware controllable image synthesis via adversarial learning. The NeRF-VAE [21] proposes to train the NeRF-based generator via variational inference, where the bottom-up inference network allows the inference of 3D structures of objects in unseen testing images. Both GRAF and NeRF-VAE assumes the object-specific latent variables to follow simple and non-informative Gaussian distributions. As a likelihood-based model, the NeRF-VAE can only handle training images with known camera poses because of the difficulty of inference of the unknown camera pose for each observed image. The GRAF, which is a non-likelihood-based generative model, can easily learn from images with unknown camera poses because its adversarial learning scheme does not need to deal with inference. Recently, we have witnessed the rapid advance of adversarial NeRF-based generative models, however, the progress in developing likelihood-based NeRF-based generative models has been lagging behind. Conceptually, likelihood-based

generative models have many advantages, e.g., stable learning process without a mode collapse issue, capability of inferring latent variables from training and testing examples, and capability of learning from incomplete data via unsupervised learning. Thus, this paper aims at pushing forward the progress of likelihood-based generative radiance field.

To be specific, by leveraging the NeRF-based image generator and the latent space energy-based models (LEBMs) [32], this paper proposes the NeRF-LEBM, a novel likelihood-based 3D-aware generative model for 2D images. It builds energy-based models (EBMs) on the latent space of the NeRF-based generator [36]. The latent space EBMs are treated as informative prior distributions. We can follow the empirical Bayes, and train the EBM priors and the NeRF-based generator simultaneously from observed data. The trainable EBM priors over latent variables (appearance and shape of the object) allow sampling novel objects from the model and rendering images with arbitrary viewpoints, as well as improve the capacity of the latent spaces and the expressivity of the NeRF-based generator. Suppose there is a set of 2D training images presenting multiple objects with various appearance, shapes and viewpoints. We first study the scenario of [21], in which the viewpoint of each image is known. We propose to train the models by maximum likelihood estimation (MLE) with Markov chain Monte Carlo [3] (MCMC)-based inference, in which no extra assisting network is required. At each iteration, the learning algorithm runs MCMC sampling of the latent variables from the EBM priors and the posteriors. The update of the EBM priors is based on the samples from the prior and the posterior distributions, while the update of the generator is based on the samples from the posteriors and the observed data. Furthermore, for efficient training and inference, we also propose to use the amortized inference to train the NeRF-LEBM as an alternative. Lastly, we do not assume the camera pose of each image is given and treat it as latent variables that follows a uniform prior distribution. We propose to use the von Mises-Fisher (vMF) [7] distribution to approximate the posterior of the camera pose in our amortized inference framework. Our experiments show that the proposed likelihood-based generative model can not only synthesize images with new objects and arbitrary viewpoints but also learn meaningful disentangled representation of images in scenarios of both known and unknown camera poses. The model can even learn from incomplete 2D training images for control generation and 3D aware inference. Our paper makes the following contributions:

1. We propose a novel NeRF-based 2D generative model, with a trainable energy-based latent space, for 3D-aware image synthesis and disentangled representation.

2. We propose to train the model by MLE with MCMC-based inference, which does not rely on separate networks and is more principled and statistically rigours than adversarial learning and variational learning.

3. We propose to train the model via amortized inference by recruiting inference networks. Due to the use of EBM priors, our algorithm is different from the learning framework of the NeRF-VAE.

4. We are the first to solve the problem of inferring unknown camera poses in variational inference framework through a novel posterior and prior setting.

5. We conduct extensive experiments to test the efficiency, effectiveness and performance of the proposed NeRF-LFBM model and learning algorithms.

1.2 Related work

3D-aware image synthesis Prior works study controllable image generation by adopting 3D data as supervision [38, 55] or 3D information as input [1, 31]. Several works [18, 15, 26, 22] build discriminative mapping functions from 2D images to 3D shapes, followed by differentiable rendering to project the 3D generated objects back to images for computing reconstruction errors on image domain. Unlike the aforementioned reconstruction-based frameworks, several recent works, such as GRAF [36], GIRAFFE [28], pi-GAN [5], and NeRF-VAE [21], build 2D generative models with NeRF function and differentiable rendering and assume unobserved object-specific variables to follow known Gaussian prior distributions. They are trained by adversarial learning [12] or variational inference [20]. Our model is also a NeRF-based generative model, but assumes latent object-specific variables to follow informative prior distributions parameterized by energy-based models [41]. We propose to train NeRF-based generator and EBM priors simultaneously by likelihood-based learning with either MCMC or amortized inference.

Energy-based models Recently, with the striking expressive power of modern deep networks, deep data space EBMs [41] have shown impressive performance in modeling distributions of different types of high-dimensional data, e.g., images [29, 53, 11, 45, 54, 48], videos [46, 47], 3D volumetric shapes [43, 44], and unordered point clouds [42]. Besides, deep latent space EBMs [32], which stand on generator networks and serve as prior distributions of the latent variables, have proven to be effective in learning expressive latent spaces for text [33, 50], image [32], trajectory generation [34], and saliency prediction [52]. In our paper, we build EBMs on the latent spaces of a NeRF-based generator to serve as prior distributions of object appearance and shape for camera pose-conditioned image generation.

MCMC inference Our model is also related to the theme of training deep latent variable models with MCMC inference [13, 39, 56, 40, 51, 30, 2, 52, 49]. Different from the above works, our paper is the first to study the MCMC inference in a NeRF-based generator with two EBM priors.

2 Background

2.1 Neural Radiance Field

A continuous scene can be represented by a Neural Radiance Field (NeRF) [25], which is a mapping function f_θ whose input is a 3D location $\mathbf{x} \in \mathbb{R}^3$ and a 3D unit vector as viewing direction $\mathbf{d} \in \mathbb{R}^3$, and whose output is an RGB color value $\mathbf{c} \in \mathbb{R}^3$ and a volume density σ. Formally, $f_\theta : (\mathbf{x}, \mathbf{d}) \to (\mathbf{c}, \sigma)$, where f_θ is a neural network with parameters θ. Given a fixed camera pose, to render a 2D image from the NeRF representation f_θ, we can follow the classical volume rendering method [17] to calculate the color of each pixel $\mathbf{v} \in \mathbb{R}^2$ in the 2D image. The color of the pixel is determined by the color and volume density values of all points along the camera ray r that goes through that pixel $\mathbf{v}$. In practice, we can follow [25] and sample M points $\{\mathbf{x}_i^r\}_{i=1}^M$ from the near to far bounds along the camera ray r and obtain a set of corresponding colors and densities $\{(\mathbf{c}_i^r, \sigma_i^r)\}_{i=1}^M$ by f_θ, and then we compute the color $C(r)$ for the camera ray r by

$$C(r) = \sum_{i=1}^M T_i^r (1 - \exp(-\sigma_i^r \delta_i^r)) \mathbf{c}_i^r, \tag{1}$$

where $\delta_i^r = ||\mathbf{x}_{i+1}^r - \mathbf{x}_i^r||_2$ is the distance between adjacent sample points, and $T_i^r = \exp(-\sum_{j=1}^{i-1} \sigma_j^r \delta_j^r)$ is the accumulated transmittance along the ray from the 1st point to the i-th point, i.e., the probability that the ray travels from $\mathbf{x}_1^r$ to $\mathbf{x}_i^r$ without being blocked. To render the whole image $\mathbf{I}$, we need to compute the color for the ray that corresponds to each pixel $\mathbf{v}$ in the image. Let $r(\mathbf{v})$ be the camera ray corresponding to the pixel $\mathbf{v}$, and the rendered image is given by $\mathbf{I}(\mathbf{v}) = C(r(\mathbf{v})), \mathbf{v} \in \mathcal{D}$, where $\mathcal{D}$ is the image domain.

2.2 Conditional Neural Radiance Field

The original NeRF function f_θ is a 3D representation of a single scene or object. To generalize the NeRF to represent different scenes or objects, [36] proposes the conditional NeRF function

$$g_\theta : (\mathbf{x}, \mathbf{d}, \mathbf{z}^s, \mathbf{z}^a) \to (\mathbf{c}, \sigma), \tag{2}$$

which is conditioned on object-specific variables, $\mathbf{z}^a$ and $\mathbf{z}^s$, corresponding to object appearance and shape respectively. It can be further decomposed into (i) $g_{\theta_1}^1 : (\mathbf{x}, \mathbf{z}^s) \to \mathbf{h}$, (ii) $g_{\theta_2}^2 : (\mathbf{h}, \mathbf{d}, \mathbf{z}^a) \to \mathbf{c}$, and (iii) $g_{\theta_3}^3 : \mathbf{h} \to \sigma$ to show the dependency among the input variables in the design of $g(\theta)$.

3 Proposed framework

3.1 NeRF-based 2D generator with EBM priors

We are interested in learning a 3D-aware generative model of 2D images, with the purposes of controllable image synthesis and disentangled image representation. We build a top-down 2D image generator based on a conditional NeRF structure for the intrinsic 3D representation of the object in an image. Let $\mathbf{z}^a$ and $\mathbf{z}^s$ be the latent variables that define the shape and the appearance of an object, respectively. $\mathbf{z}^a$ and $\mathbf{z}^s$ are assumed to be independent. They together specify an object. Let $\boldsymbol{\xi}$ be the camera pose. The generator G_θ consists of an object-conditioned NeRF function g_θ as shown in Eq. (2) and a differentiable rendering function as shown in Eq. (1). θ are trainable parameters of the generator. Given an object specified by $(\mathbf{z}^a, \mathbf{z}^s)$, the generator takes the camera pose $\boldsymbol{\xi}$ as input and outputs an image by using the NeRF g_θ to render an image from the pose $\boldsymbol{\xi}$ with the render operation in Eq. (1). Given a dataset of 2D images of different objects captured from different viewing angles (i.e., different camera pose), in which the camera pose of each image is provided. We assume each image is generated by following the generative process defined by G_θ and each of the latent variables $(\mathbf{z}^a, \mathbf{z}^s)$ is assumed to follow an informative prior distribution that is defined by a trainable energy-based model (EBM). Specifically, the proposed 3D-aware image-based generative model is given by the following deep latent variable model

$$\begin{aligned} \mathbf{I} &= G_\theta(\mathbf{z}^a, \mathbf{z}^s, \boldsymbol{\xi}) + \epsilon, \\ \epsilon &\sim \mathcal{N}(0, \sigma_\epsilon^2 I), \\ \mathbf{z}^a &\sim p_{\alpha_a}(\mathbf{z}^a), \\ \mathbf{z}^s &\sim p_{\alpha_s}(\mathbf{z}^s), \end{aligned} \tag{3}$$

where ϵ is the observation residual following a Gaussian distribution $\mathcal{N}(0, \sigma_\epsilon^2 I)$ with a known standard deviation σ_ϵ, and I denotes the identity matrix. Both $p_{\alpha_a}(\mathbf{z}^a)$ and $p_{\alpha_s}(\mathbf{z}^s)$ are modeled by EBMs

$$p_{\alpha_a}(\mathbf{z}^a) = \frac{1}{Z(\alpha_a)} \exp[-U_{\alpha_a}(\mathbf{z}^a)] q_0(\mathbf{z}^a), \tag{4}$$

$$p_{\alpha_s}(\mathbf{z}^s) = \frac{1}{Z(\alpha_s)} \exp[-U_{\alpha_s}(\mathbf{z}^s)] q_0(\mathbf{z}^s), \tag{5}$$

which are in the form of exponential tilting of a Gaussian reference distribution $q_0 \sim \mathcal{N}(0, \sigma^2 I)$. (Note that q_0 could be a uniform reference distribution.) $U_{\alpha_a}(\mathbf{z}^a)$ and $U_{\alpha_s}(\mathbf{z}^s)$ are called energy functions, both of which are parameterized by multilayer perceptrons (MLPs) with trainable parameters α_a and α_s, respectively. The energy function takes the corresponding latent variables as input and outputs a scalar as energy. Besides, $Z(\alpha_a) = \int \exp[-U_{\alpha_a}(\mathbf{z}^a)] q_0(\mathbf{z}^a) d\mathbf{z}^a$ and $Z(\alpha_s) = \int \exp[-U_{\alpha_s}(\mathbf{z}^s)] q_0(\mathbf{z}^s) d\mathbf{z}^s$ are intractable normalizing constants. Although $q_0(\mathbf{z}^a)$ and $q_0(\mathbf{z}^s)$ are Gaussian distributions, $p_{\alpha_a}(\mathbf{z}^a)$ and $p_{\alpha_z}(\mathbf{x}^s)$ are non-Gaussian priors, where α_a and α_s are learned from the data together with the parameters θ of the generator G_θ.

3.2 Learning with MCMC-based inference

For convenience of notation, let $\beta = (\theta, \alpha_a, \alpha_s)$ and $\alpha = (\alpha_a, \alpha_s)$. Given a set of 2D images with known camera

poses, i.e., $\{(\mathbf{I}_i, \boldsymbol{\xi}_i), i = 1, ..., n\}$, we can train β by maximizing the observed-data log-likelihood function defined as

$$L(\beta) = \frac{1}{n} \sum_{i=1}^{n} \log p_\beta(\mathbf{I}_i | \boldsymbol{\xi}_i)$$

$$= \frac{1}{n} \sum_{i=1}^{n} \log \left[\int p_\alpha(\mathbf{z}_i^a, \mathbf{z}_i^s) p_\theta(\mathbf{I}_i | \mathbf{z}_i^a, \mathbf{z}_i^s, \boldsymbol{\xi}_i) d\mathbf{z}_i^a d\mathbf{z}_i^s \right]$$

$$= \frac{1}{n} \sum_{i=1}^{n} \log \left[\int p_{\alpha_a}(\mathbf{z}_i^a) p_{\alpha_s}(\mathbf{z}_i^s) p_\theta(\mathbf{I}_i | \mathbf{z}_i^a, \mathbf{z}_i^s, \boldsymbol{\xi}_i) d\mathbf{z}_i^a d\mathbf{z}_i^s \right],$$

where $p_\alpha(\mathbf{z}^a, \mathbf{z}^s) = p_{\alpha_a}(\mathbf{z}^a) p_{\alpha_s}(\mathbf{z}^s)$ because $\mathbf{z}^a$ and $\mathbf{z}^s$ are statistically independent, and the latent variables are integrated out in the complete-data log-likelihood. According to the Law of Large Number, maximizing the likelihood $L(\beta)$ is approximately equivalent to minimizing the Kullback-Leibler (KL) divergence between model $p_\beta(\mathbf{I}|\boldsymbol{\xi})$ and data distribution $p_{\text{data}}(\mathbf{I}|\boldsymbol{\xi})$ if the number n of training examples is very large. The gradient of $L(\beta)$ is calculated based on

$$\nabla_\beta \log p_\beta(\mathbf{I}|\boldsymbol{\xi}) = \mathbb{E}_{p_\beta(\mathbf{z}^a, \mathbf{z}^s | \mathbf{I}, \boldsymbol{\xi})} [\nabla_\beta \log p_\beta(\mathbf{I}, \mathbf{z}^a, \mathbf{z}^s | \boldsymbol{\xi})]$$

$$= \mathbb{E}_{p_\beta(\mathbf{z}^a, \mathbf{z}^s | \mathbf{I}, \boldsymbol{\xi})} [\nabla_\beta \log p_{\alpha_a}(\mathbf{z}^a) + \nabla_\beta \log p_{\alpha_s}(\mathbf{z}^s)$$
$$+ \nabla_\beta \log p_\theta(\mathbf{I}|\mathbf{z}^a, \mathbf{z}^s, \boldsymbol{\xi})], \quad (6)$$

which can be further decomposed into three parts, i.e., the gradients for the EBM prior of object appearance α_a

$$\mathbb{E}_{p_\beta(\mathbf{z}^a, \mathbf{z}^s | \mathbf{I}, \boldsymbol{\xi})} [\nabla_\beta \log p_{\alpha_a}(\mathbf{z}^a)] \quad (7)$$
$$= \mathbb{E}_{p_{\alpha_a}(\mathbf{z}^a)} [\nabla_{\alpha_a} U_{\alpha_a}(\mathbf{z}^a)] - \mathbb{E}_{p_\beta(\mathbf{z}^a | \mathbf{I}, \boldsymbol{\xi})} [\nabla_{\alpha_a} U_{\alpha_a}(\mathbf{z}^a)],$$

the gradients for the EBM prior of object shape α_s

$$\mathbb{E}_{p_\beta(\mathbf{z}^a, \mathbf{z}^s | \mathbf{I}, \boldsymbol{\xi})} [\nabla_\beta \log p_{\alpha_s}(\mathbf{z}^s)] \quad (8)$$
$$= \mathbb{E}_{p_{\alpha_s}(\mathbf{z}^s)} [\nabla_{\alpha_s} U_{\alpha_s}(\mathbf{z}^s)] - \mathbb{E}_{p_\beta(\mathbf{z}^s | \mathbf{I}, \boldsymbol{\xi})} [\nabla_{\alpha_s} U_{\alpha_s}(\mathbf{z}^s)],$$

as well as the gradients for the NeRF-based generator θ

$$\mathbb{E}_{p_\beta(\mathbf{z}^a, \mathbf{z}^s | \mathbf{I}, \boldsymbol{\xi})} [\nabla_\beta \log p_\theta(\mathbf{I}|\mathbf{z}^a, \mathbf{z}^s, \boldsymbol{\xi})] \quad (9)$$
$$= \mathbb{E}_{p_\beta(\mathbf{z}^a, \mathbf{z}^s | \mathbf{I}, \boldsymbol{\xi})} [\nabla_\theta G_\theta(\mathbf{z}^a, \mathbf{z}^s, \boldsymbol{\xi})(\mathbf{I} - G_\theta(\mathbf{z}^a, \mathbf{z}^s, \boldsymbol{\xi}))/\sigma_\epsilon^2].$$

Since the expectations in Eq. (7), Eq. (8), and Eq. (9) are analytically intractable, Langevin dynamics [27], which is a gradient-based MCMC sampling method, is employed to draw samples from the prior distributions (i.e., $p_{\alpha_a}(\mathbf{z}^a)$ and $p_{\alpha_s}(\mathbf{z}^s)$) and the posterior distribution (i.e., $p_\beta(\mathbf{z}^a, \mathbf{z}^s | \mathbf{I}, \boldsymbol{\xi})$), and then Monte Carlo averages are computed to estimate the expectation terms. As shown in Eq. (7) and Eq. (8), the update of the EBM prior model α_a (or α_s) is based on the difference between $\mathbf{z}^a$ (or $\mathbf{z}^s$) sampled from the prior distribution $p_{\alpha_s}(\mathbf{z}^a)$ (or $p_{\alpha_s}(\mathbf{z}^s)$) and $\mathbf{z}^a$ (or $\mathbf{z}^s$) inferred from the posterior distribution $p_\beta(\mathbf{z}^a | \mathbf{I}, \boldsymbol{\xi})$ (or $p_\beta(\mathbf{z}^s | \mathbf{I}, \boldsymbol{\xi})$). According to Eq.(9), the update of the generator θ relies on $\mathbf{z}^a$ and $\mathbf{z}^s$ inferred from the posterior distribution $p_\beta(\mathbf{z}^a, \mathbf{z}^s | \mathbf{I}, \boldsymbol{\xi})$. To sample from the prior distributions $p_{\alpha_a}(\mathbf{z}^a)$ and $p_{\alpha_s}(\mathbf{z}^s)$ by Langevin dynamics, we update $\mathbf{z}^a$ and $\mathbf{z}^s$ by

$$\mathbf{z}_{t+1}^a = \mathbf{z}_t^a + \delta \nabla_{\mathbf{z}^a} \log p_{\alpha_a}(\mathbf{z}_t^a) + \sqrt{2\delta} \mathbf{e}_t^a, \quad (10)$$

$$\mathbf{z}_{t+1}^s = \mathbf{z}_t^s + \delta \nabla_{\mathbf{z}^s} \log p_{\alpha_s}(\mathbf{z}_t^s) + \sqrt{2\delta} \mathbf{e}_t^s, \quad (11)$$

where t indexes the time step, δ is the Langevin step size, and $\mathbf{e}_t^a$ and $\mathbf{e}_t^s$ are independent Gaussian noises that help the MCMC chains to escape from local modes during sampling. The gradients in Eq. (10) and Eq. (11) are given by

$$\nabla_{\mathbf{z}^a} \log p_{\alpha_a}(\mathbf{z}^a) = -\nabla_{\mathbf{z}^a} U_{\alpha_a}(\mathbf{z}^a) - \mathbf{z}^a / \sigma^2, \quad (12)$$

$$\nabla_{\mathbf{z}^s} \log p_{\alpha_s}(\mathbf{z}^s) = -\nabla_{\mathbf{z}^s} U_{\alpha_s}(\mathbf{z}^s) - \mathbf{z}^s / \sigma^2, \quad (13)$$

where $\nabla_{\mathbf{z}^a} U_{\alpha_a}(\mathbf{z}^a)$ and $\nabla_{\mathbf{z}^s} U_{\alpha_s}(\mathbf{z}^s)$ are efficiently computed by back-propagation.

For each observed $(\mathbf{I}, \boldsymbol{\xi})$, we can sample from the posterior $p_\beta(\mathbf{z}^a, \mathbf{z}^s | \mathbf{I}, \boldsymbol{\xi})$ by alternately running Langevin dynamics: we fix $\mathbf{z}^s$ and sample $\mathbf{z}^a$ from $p_\beta(\mathbf{z}^a | \mathbf{z}^s, \mathbf{I}, \boldsymbol{\xi}) \propto p_\beta(\mathbf{I}, \mathbf{z}^a | \mathbf{z}^s, \boldsymbol{\xi})$, and then fix $\mathbf{z}^a$ and sample $\mathbf{z}^s$ from $p_\beta(\mathbf{z}^s | \mathbf{z}^a, \mathbf{I}, \boldsymbol{\xi}) \propto p_\beta(\mathbf{I}, \mathbf{z}^s | \mathbf{z}^a, \boldsymbol{\xi})$. The Langevin sampling step follows

$$\mathbf{z}_{t+1}^a = \mathbf{z}_t^a + \delta \nabla_{\mathbf{z}^a} \log p_\beta(\mathbf{I}, \mathbf{z}_t^a | \mathbf{z}_t^s, \boldsymbol{\xi}) + \sqrt{2\delta} \mathbf{e}_t^a, \quad (14)$$

$$\mathbf{z}_{t+1}^s = \mathbf{z}_t^s + \delta \nabla_{\mathbf{z}^s} \log p_\beta(\mathbf{I}, \mathbf{z}_t^s | \mathbf{z}_t^a, \boldsymbol{\xi}) + \sqrt{2\delta} \mathbf{e}_t^s. \quad (15)$$

The key steps in Eq. (14) and Eq. (15) are to compute the gradients of

$$\log p_\beta(\mathbf{I}, \mathbf{z}^a | \mathbf{z}^s, \boldsymbol{\xi}) = \log[p_{\alpha_a}(\mathbf{z}^a) p_\theta(\mathbf{I}|\mathbf{z}^a, \mathbf{z}^s, \boldsymbol{\xi})] = C_a$$
$$- ||\mathbf{I} - G_\theta(\mathbf{z}^a, \mathbf{z}^s, \boldsymbol{\xi})||^2 / 2\sigma_\epsilon^2 - U_{\alpha_a}(\mathbf{z}^a) - ||\mathbf{z}^a||^2 / 2\sigma^2,$$

$$\log p_\beta(\mathbf{I}, \mathbf{z}^s | \mathbf{z}^a, \boldsymbol{\xi}) = \log[p_{\alpha_s}(\mathbf{z}^s) p_\theta(\mathbf{I}|\mathbf{z}^s, \mathbf{z}^a, \boldsymbol{\xi})] = C_s$$
$$- ||\mathbf{I} - G_\theta(\mathbf{z}^a, \mathbf{z}^s, \boldsymbol{\xi})||^2 / 2\sigma_\epsilon^2 - U_{\alpha_s}(\mathbf{z}^s) - ||\mathbf{z}^s||^2 / 2\sigma^2,$$

where C_a and C_s are constants independent of $\mathbf{z}^a$, $\mathbf{z}^s$ and θ. After sufficient alternating Langevin steps, the updated $\mathbf{z}^a$ and $\mathbf{z}^s$ follow the joint posterior $p_\beta(\mathbf{z}^a, \mathbf{z}^s | \mathbf{I}, \boldsymbol{\xi})$, and $\mathbf{z}^a$ and $\mathbf{z}^s$ follow $p_\beta(\mathbf{z}^a | \mathbf{I}, \boldsymbol{\xi})$ and $p_\beta(\mathbf{z}^s | \mathbf{I}, \boldsymbol{\xi})$, respectively.

Let $\mathbf{z}_i^{a-}$ and $\mathbf{z}_i^{s-}$ be the samples drawn from the EBM priors by Langevin dynamics in Eqs. (10) and (11). Let $\mathbf{z}_i^{a+}$ and $\mathbf{z}_i^{s+}$ be the inferred latent variables of the observation $(\mathbf{I}_i, \boldsymbol{\xi}_i)$ by Langevin dynamics in Eqs. (14) and (15). The gradients of the log-likelihood L over α_a, α_s, and θ are estimated by

$$\nabla_{\alpha_a} L = \frac{1}{n} \sum_{i=1}^{n} [\nabla_{\alpha_a} U_{\alpha_a}(\mathbf{z}_i^{a-})] - \frac{1}{n} \sum_{i=1}^{n} [\nabla_{\alpha_a} U_{\alpha_a}(\mathbf{z}_i^{a+})],$$

$$\nabla_{\alpha_s} L = \frac{1}{n} \sum_{i=1}^{n} [\nabla_{\alpha_s} U_{\alpha_s}(\mathbf{z}_i^{s-})] - \frac{1}{n} \sum_{i=1}^{n} [\nabla_{\alpha_s} U_{\alpha_s}(\mathbf{z}_i^{s+})],$$

$$\nabla_\theta L = \frac{1}{n} \sum_{i=1}^{n} \left[\nabla_\theta G_\theta(\mathbf{z}_i^{a+}, \mathbf{z}_i^{s+}, \boldsymbol{\xi}_i) \frac{\mathbf{I}_i - G_\theta(\mathbf{z}_i^{a+}, \mathbf{z}_i^{s+}, \boldsymbol{\xi}_i)}{\sigma_\epsilon^2} \right].$$

The learning algorithm of the NeRF-LEBM with MCMC inference can be summarized in Algorithm 1.

3.3 Learning with amortized inference

Even though both prior and posterior sampling require Langevin dynamics. Prior sampling is more affordable than posterior sampling because the network structure of

Algorithm 1 Learning NeRF-LEBM with MCMC inference

Input: (1) Images and viewpoints $\{(\mathbf{I}_i, \boldsymbol{\xi}_i)\}_{i=1}^n$; (2) Numbers of Langevin steps for priors and posterior $\{K^-, K^+\}$; (3) Langevin step sizes for priors and posterior$\{\delta^-, \delta^+\}$; (4) Learning rates for priors and generator $\{\eta_\alpha, \eta_\theta\}$.
Output: (1) θ for generator; (2) (α_a, α_s) for EBM priors; (3) Latent variables $\{(\mathbf{z}_i^a, \mathbf{z}_i^s)\}_{i=1}^n$.

1: Randomly initialize θ, α_a, α_s, and $\{(\mathbf{z}_i^a, \mathbf{z}_i^s)\}_{i=1}^n$.
2: **repeat**
3:　　For each $(\mathbf{I}_i, \boldsymbol{\xi}_i)$, sample the prior of object appearance $\mathbf{z}_i^{a-} \sim p_{\alpha_a}(\mathbf{z}^a)$ and the prior of object shape $\mathbf{z}_i^{s-} \sim p_{\alpha_s}(\mathbf{z}^s)$ using K^- steps of Langevin dynamics with a step size δ^-, which follows Eq. (10) and Eq. (11), respectively.
4:　　For each $(\mathbf{I}_i, \boldsymbol{\xi}_i)$, run K^+ Langevin steps with a step size δ^+, to alternatively sample $\mathbf{z}_i^a$ from $p_\beta(\mathbf{z}_i^a|\mathbf{z}_i^s, \mathbf{I}_i, \boldsymbol{\xi}_i)$, while fixing $\mathbf{z}_i^s$; and sample $\mathbf{z}_i^s$ from $p_\beta(\mathbf{z}_i^s|\mathbf{z}_i^a, \mathbf{I}_i, \boldsymbol{\xi}_i)$, while fixing $\mathbf{z}_i^a$.
5:　　$\alpha_a \leftarrow \alpha_a + \eta_\alpha \nabla_{\alpha_a} L$.
6:　　$\alpha_s \leftarrow \alpha_s + \eta_\alpha \nabla_{\alpha_s} L$.
7:　　$\theta \leftarrow \theta + \eta_\theta \nabla_\theta L$.
8: **until** converge

Algorithm 2 Variational Learning for NeRF-LEBM

Input: (1) Images and viewpoints $\{(\mathbf{I}_i, \boldsymbol{\xi}_i)\}_{i=1}^n$; (2) Number of Langevin steps K^- for priors; (3) Langevin step size for priors δ^-; (4) Learning rates $\{\eta_\alpha, \eta_\omega\}$.
Output: (1) θ for generator; (2) (α_a, α_s) for EBM priors; (3) ϕ for inference net.

1: Randomly initialize θ, ϕ, α_a, and α_s.
2: **repeat**
3:　　For each $(\mathbf{I}_i, \boldsymbol{\xi}_i)$, sample the priors $\mathbf{z}_i^{a-} \sim p_{\alpha_a}(\mathbf{z}^a)$ and $\mathbf{z}_i^{s-} \sim p_{\alpha_s}(\mathbf{z}^s)$ using K^- Langevin steps with a step size δ^-, which follow Eq. (10) and Eq. (11) respectively.
4:　　For each $(\mathbf{I}_i, \boldsymbol{\xi}_i)$, sample $\mathbf{z}^a \sim q_{\phi_a}(\mathbf{z}^a|\mathbf{I}_i, \boldsymbol{\xi}_i)$ and $\mathbf{z}_i^s \sim q_{\phi_s}(\mathbf{z}^s|\mathbf{I}_i, \boldsymbol{\xi}_i)$ using the inference network.
5:　　$\alpha_a \leftarrow \alpha_a + \eta_\alpha \nabla_{\alpha_a}$ELBO ($\nabla_{\alpha_a}$ELBO is in Eq. (18)).
6:　　$\alpha_s \leftarrow \alpha_s + \eta_\alpha \nabla_{\alpha_s}$ELBO ($\nabla_{\alpha_s}$ELBO is in Eq. (19)).
7:　　$\omega \leftarrow \omega + \eta_\omega \nabla_\omega$ELBO ($\nabla_\omega$ELBO is in Eq. (20), where $\omega = (\phi_a, \theta)$).
8: **until** converge

U_{α_a} or U_{α_s} is much smaller than that of the NeRF-based generator G_θ and the posterior sampling need to perform back-propagation on G_θ, which is time-consuming. In this section, we propose to train the NeRF-LEBM by adopting amortized inference, in which the posterior distributions, $p_\beta(\mathbf{z}^a|\mathbf{I}, \boldsymbol{\xi})$ and $p_\beta(\mathbf{z}^s|\mathbf{I}, \boldsymbol{\xi})$, are approximated by separate bottom-up inference networks with reparameterization trick, $q_{\phi_a}(\mathbf{z}^a|\mathbf{I}, \boldsymbol{\xi}) = \mathcal{N}(\mathbf{z}^a|u_{\phi_a}(\mathbf{I}|\boldsymbol{\xi}), \sigma_{\phi_a}(\mathbf{I}|\boldsymbol{\xi}))$ and $q_{\phi_s}(\mathbf{z}^s|\mathbf{I}, \boldsymbol{\xi}) = \mathcal{N}(\mathbf{z}^s|u_{\phi_s}(\mathbf{I}|\boldsymbol{\xi}), \sigma_{\phi_s}(\mathbf{I}|\boldsymbol{\xi}))$, respectively. We denote $\phi = (\phi_a, \phi_s)$ for notation simplicity. The log-likelihood $\log p_\beta(\mathbf{I}|\boldsymbol{\xi})$ is lower bounded by the evidence lower bound (ELBO), which is given by

$$
\begin{aligned}
&\text{ELBO}(\mathbf{I}|\boldsymbol{\xi}; \beta, \phi) \\
&= \log p_\beta(\mathbf{I}|\boldsymbol{\xi}) - \mathbb{D}_{\text{KL}}(q_{\phi_a}(\mathbf{z}^a|\mathbf{I}, \boldsymbol{\xi})||p_\beta(\mathbf{z}^a|\mathbf{I}, \boldsymbol{\xi})) \\
&\quad - \mathbb{D}_{\text{KL}}(q_{\phi_s}(\mathbf{z}^s|\mathbf{I}, \boldsymbol{\xi})||p_\beta(\mathbf{z}^s|\mathbf{I}, \boldsymbol{\xi})) \quad (16)\\
&= - \mathbb{D}_{\text{KL}}(q_{\phi_s}(\mathbf{z}^s|\mathbf{I}, \boldsymbol{\xi})||p_{\alpha^s}(\mathbf{z}^s)) \\
&\quad - \mathbb{D}_{\text{KL}}(q_{\phi_a}(\mathbf{z}^a|\mathbf{I}, \boldsymbol{\xi})||p_{\alpha^a}(\mathbf{z}^a)) \\
&\quad + \mathbb{E}_{q_{\phi_a}(\mathbf{z}^a|\mathbf{I},\boldsymbol{\xi})q_{\phi_s}(\mathbf{z}^s|\mathbf{I},\boldsymbol{\xi})}[\log p_\theta(\mathbf{I}|\mathbf{z}^a, \mathbf{z}^s, \boldsymbol{\xi})], \quad (17)
\end{aligned}
$$

where $\mathbb{D}_{\text{KL}}$ denotes the Kullback-Leibler divergence. We assume $p_{\alpha^s}(\mathbf{z}^s) = p_{\alpha^s}(\mathbf{z}^s|\boldsymbol{\xi})$ and $p_{\alpha^a}(\mathbf{z}^a) = p_{\alpha^a}(\mathbf{z}^a|\boldsymbol{\xi})$. For the EBM prior models, the learning gradients to update α_a and α_s are given by

$$
\begin{aligned}
&\nabla_{\alpha_a}\text{ELBO}(\mathbf{I}|\boldsymbol{\xi}; \beta, \phi) \quad (18)\\
&= \mathbb{E}_{p_{\alpha_a}(\mathbf{z}^a)}[\nabla_{\alpha_a} U_{\alpha_a}(\mathbf{z}^a)] - \mathbb{E}_{q_{\phi_a}(\mathbf{z}^a|\mathbf{I},\boldsymbol{\xi})}[\nabla_{\alpha_a} U_{\alpha_a}(\mathbf{z}^a)], \\
&\nabla_{\alpha_s}\text{ELBO}(\mathbf{I}|\boldsymbol{\xi}; \beta, \phi) \quad (19)\\
&= \mathbb{E}_{p_{\alpha_s}(\mathbf{z}^s)}[\nabla_{\alpha_s} U_{\alpha_s}(\mathbf{z}^s)] - \mathbb{E}_{q_{\phi_s}(\mathbf{z}^s|\mathbf{I},\boldsymbol{\xi})}[\nabla_{\alpha_s} U_{\alpha_s}(\mathbf{z}^s)].
\end{aligned}
$$

Let $\omega = (\phi, \theta)$ be the parameters of the inference networks and generator. The learning gradients of these models are

$$
\begin{aligned}
&\nabla_\omega\text{ELBO}(\mathbf{I}|\boldsymbol{\xi}; \beta, \phi) \quad (20)\\
&= \nabla_\omega \mathbb{E}_{q_{\phi_a}(\mathbf{z}^a|\mathbf{I},\boldsymbol{\xi})q_{\phi_s}(\mathbf{z}^s|\mathbf{I},\boldsymbol{\xi})}[\log p_\theta(\mathbf{I}|\mathbf{z}^a, \mathbf{z}^s, \boldsymbol{\xi})] \\
&\quad - \nabla_\omega \mathbb{D}_{\text{KL}}(q_{\phi_a}(\mathbf{z}^a|\mathbf{I}, \boldsymbol{\xi})||p_0(\mathbf{z}^a)) - \nabla_\omega \mathbb{E}_{q_{\phi_a}(\mathbf{z}^a|\mathbf{I},\boldsymbol{\xi})}[U_{\alpha_a}(\mathbf{z}^a)] \\
&\quad - \nabla_\omega \mathbb{D}_{\text{KL}}(q_{\phi_s}(\mathbf{z}^s|\mathbf{I}, \boldsymbol{\xi})||p_0(\mathbf{z}^s)) - \nabla_\omega \mathbb{E}_{q_{\phi_s}(\mathbf{z}^s|\mathbf{I},\boldsymbol{\xi})}[U_{\alpha_s}(\mathbf{z}^s)]
\end{aligned}
$$

The first term on the right hand size of Eq.(20) is the reconstruction by the bottom-up inference encoders and the top-down generator. The second and the fourth terms are KL divergences between the inference model and the Gaussian distribution. These three terms form the learning objective of the original VAE. The variational learning of the NeRF-LEBM is given in Algorithm 2.

3.4　Learning without ground truth camera pose

Many real world datasets do not contain camera pose information, therefore fitting the models from those datasets by using Algorithms 1 or 2 is not appropriate. In this section, we study learning the NeRF-LFBM model from images without knowing the ground truth camera poses, and generalizing Algorithm 2 to this scenario. We treat the unknown camera pose as latent variables and seek to infer it together with the shape and appearance variables in the amortized learning framework. In our experiments, we assume the camera is located on a sphere and the object is put in the center of the sphere. Therefore, the camera pose $\boldsymbol{\xi}$ can be interpreted as the altitude angle ξ_1 and azimuth angle ξ_2. However, different from the shape and the appearance, the camera pose is directional and can be better explained

through a spherical representation [24]. For implementation, instead of directly representing each individual angle, we represent its Sine and Cosine values that directly construct the corresponding rotation matrix that is useful for subsequent computation. Thus, each rotation angle ξ_i is a two-dimensional unit norm vector located on a unit sphere.

Following the hyperspherical VAE in [7], we use the von Mises-Fisher (vMF) distribution to model the posterior distribution of ξ. vMF can be seen as the Gaussian distribution on a hyphershere. To model a hypershpere of dimension m, it is parameterized by a mean direction $\boldsymbol{\mu} \in \mathbb{R}^m$ and a concentration parameter $\kappa \in \mathbb{R}_{\geq 0}$. The probability density of the vMF is defined as $p_{\text{vMF}}(\xi|\boldsymbol{\mu}, \kappa) = \mathcal{C}_m(\kappa)\exp(\kappa\boldsymbol{\mu}^T\boldsymbol{\xi})$, where $\mathcal{C}_m(\kappa) = \frac{\kappa^{m/2-1}}{(2\pi)^{m/2}\mathcal{I}_{m/2-1}(\kappa)}$, with $\mathcal{I}_l$ denoting modified Bessel function of the first kind at order l. For each angel ξ in $\boldsymbol{\xi}$, we design an inference model as $q_{\phi_\xi}(\xi|\mathbf{I}) = p_{\text{vMF}}(\xi|\boldsymbol{\mu}_{\phi_\xi}(\mathbf{I}), \kappa_{\phi_\xi}(\mathbf{I}))$, where $\boldsymbol{\mu}_{\phi_\xi}(\mathbf{I})$ and $\kappa_{\phi_\xi}(\mathbf{I})$ are bottom-up networks with parameters ϕ_ξ that maps $\mathbf{I}$ to $\boldsymbol{\mu}$ and κ. We assume the prior of ξ to be a uniform distribution on the unit sphere (denoted as $U(S^{m-1})$), which is the special case of vMF with $\kappa = 0$. The key to use the amortized inference is to compute the KL divergence between the posterior and the prior, which can follow

$$\mathbb{D}_{\text{KL}}(p_{\text{vMF}}(\boldsymbol{\mu}, \kappa)||U(S^{m-1}))$$
$$=\kappa\frac{\mathcal{I}_{m/2}(\kappa)}{\mathcal{I}_{m/2-1}(\kappa)} + \log\mathcal{C}_m(\kappa) - \log\left(\frac{2(\pi^{m/2})}{\Gamma m/2}\right)^{-1}. \quad (21)$$

Besides, to compute the ELBO, we need to draw samples from the inference model $q_{\phi_\xi}(\xi|\mathbf{I})$, which amounts to sampling from the vMF distribution. We follow the sampling procedure in [7] for this purpose in our implementation.

4 Experiments

4.1 Datasets

To evaluate the proposed NeRF-LEBM framework and the learning algorithms, we conduct experiments on three datasets. The Carla dataset is rendered by [36] using the Carla Driving Simulator [8]. It contains 10k cars of different shapes, colors and textures. Each car has one 2D image rendered from one random camera pose. Another dataset is the ShapeNet [6] Car dataset, which contains 2.1k different cars for training and 700 cars for testing. We use the images rendered by [37] and follow its split to separate the training and testing sets. Each car in the training set has 250 views and we only use 50 views of them for training. Each car in the testing set has 251 views. each image is associated with its camera pose information.

4.2 Random image synthesis

We first evaluate the capability of image generation of the NeRF-LEBM on the Carla dataset, where the camera pose

(a) NeRF-LEBM with MCMC inference

(b) NeRF-LEBM with amortized inference

Figure 1: Images generated by the NeRF-LEBM models trained on the Carla dataset, where the camera poses are given. (a) MCMC inference (b) amortized inference.

information is available. We try to answer whether the latent space EBMs can capture the underlying factors of objects in images and whether it is better than the Gaussian prior. We train our models on images of resolution 64×64 through both MCMC-based inference in Algorithm 1 and amortized inference in Algorithm 2. Once a model is trained, we can generate new images by first randomly sampling $(\mathbf{z}^a, \mathbf{z}^s)$ from the learned EBM priors and a camera pose $\boldsymbol{\xi}$ from a uniform distribution, and then using the NeRF-based generator to map the sampled latent variables to the image space. The synthesized images by NeRF-LEBM using MCMC inference and amortized inference are displayed in Figures 1a and 1b, respectively. We can see the learned models can generate meaningful and highly diversified cars with different shapes, appearances and camera poses. To quantitatively evaluate the generative performance, we compare our NeRF-LEBMs with some baselines in terms of Fréchet inception distance (FID) [16] in Table 1. The baselines include the NeRF-VAE [21], which is a NeRF-based generator using Gaussian prior and trained with variational learning, and the NeRF-Gaussian-MCMC, which is a NeRF-based generator using MCMC inference and Gaussian prior. To make a fair

(a)   (b)   (c)
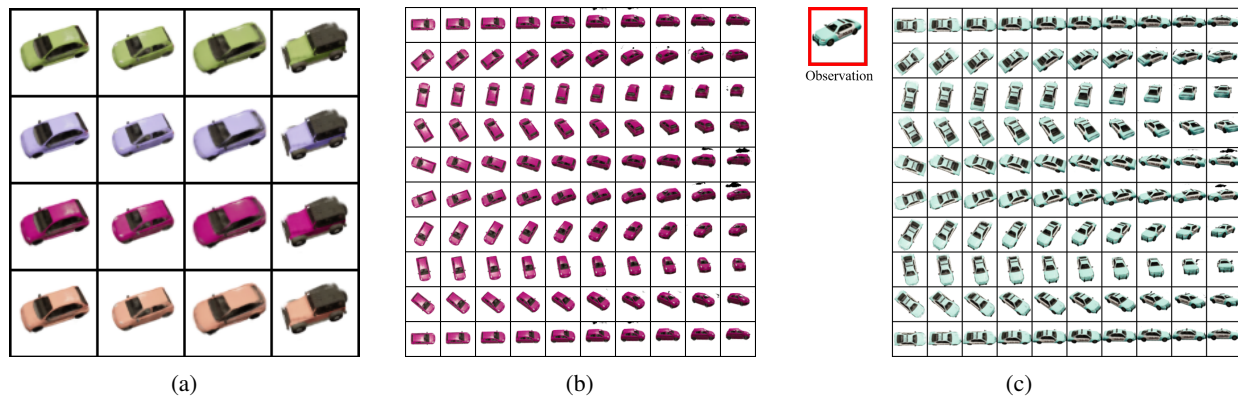
Figure 2: Disentangled representation. The generated images are obtained by the learned NeRF-LEBM using amortized inference on the Carla dataset. (a) shows the influences of the shape vector $\mathbf{z}^s$ and the appearance vector $\mathbf{z}^a$ in image synthesis. The objects in each row share the same appearance vector $\mathbf{z}^a$ and camera pose $\boldsymbol{\xi}$ but have different shape vectors $\mathbf{z}^s$, while the objects in each column share the same shape vector $\mathbf{z}^s$ and camera pose $\boldsymbol{\xi}$ but have different appearance vectors $\mathbf{z}^a$. (b) demonstrates the effect of the camera pose variable $\boldsymbol{\xi}$ by varying it while fixing the shape and appearance vectors for a randomly sampled object. (c) shows an example of novel view synthesis for an observed 2D image.

comparison, we implement the NeRF-VAE using the same NeRF-based generator and inference network as those in our NeRF-LEBM using amortized inference, except that the NeRF-VAE and the NeRF-Gaussian-MCMC only adopt Gaussian priors for latent variables. We compute FID using 10k samples. Table 1 shows that NeRF-LEBMs perform very well in the sense that the learned models can generate realistic images. Especially, the NeRF-EBM trained with amortized inference obtains the best performance. The comparison between the NeRF-VAE and our NeRF-LEBM using amortized inference demonstrates the effectiveness of the EBM priors. The efficacy of the EBM priors is also validated by the comparison between the NeRF-Gaussian-MCMC and our NeRF-LEBM using MCMC inference.

4.3 Disentangled representation

We investigate the ability of disentanglement of the NeRF-LEBM. We test the model using amortized inference trained in Section 4.2. We generate images by varying one of the three latent vectors, i.e., $\mathbf{z}^a$, $\mathbf{z}^s$ and $\boldsymbol{\xi}$, while fixing the other two, and observe how the manipulated vector influences the generated images. The generated images are shown in

Table 1: Comparing the NeRF-LEBMs with likelihood-based baselines on the 64×64 Carla dataset for random image synthesis. The image qualities are evaluated via FID.

Likelihood-based Model	FID ↓
NeRF-Gaussian-MCMC	54.13
NeRF-VAE [21]	38.15
NeRF-LEBM (MCMC inference)	37.19
NeRF-LEBM (Amortized inference)	**20.84**

Figure 2. In Figure 2a, the objects in each row share the same appearance vector and camera pose but have different shape vectors, while the objects in each column share the same shape vector and camera pose but have different appearance vectors. From Figure 2a, we can see that the shape latent vectors do not encode any appearance information, such as color. The colors in the generated images only depend on the appearance latent vectors, and are not influenced by the shape latent vectors. Figure 2b displays the synthesized images sharing the same appearance and shape vectors but having different camera poses. Figures 2a and 2b show that the learned model can successfully disentangle the appearance, shape and camera pose of an object because $\mathbf{z}^a$, $\mathbf{z}^s$ and $\boldsymbol{\xi}$ can, respectively, control the appearance, shape, and viewpoint of the generated images. We can also perform novel view synthesis of a seen 2D object by first inferring its appearance and shape vectors and then using different camera poses to generate different views of the object. Figure 2c shows one example. Please refer to Figure 11 in Appendix for results obtained by a model trained with MCMC inference.

4.4 Inferring 3D structures of unseen 2D objects

Once a NeRF-LEBM model is trained, it is capable of inferring the 3D structure of a previously unseen object from only a few observations. Following [37], we first train our model on the 128×128 resolution ShapeNet Car training set and then apply the trained model to a two-shot novel view synthesis task, in which only two views of an unseen car in the holdout testing set are given to synthesize novel views of the same car. We use the NeRF-LEBM with MCMC inference in this task. To reduce the computational cost, we follow [39] to adopt a persistent MCMC chain setting for the Langevin inference. Given a unseen object, we first
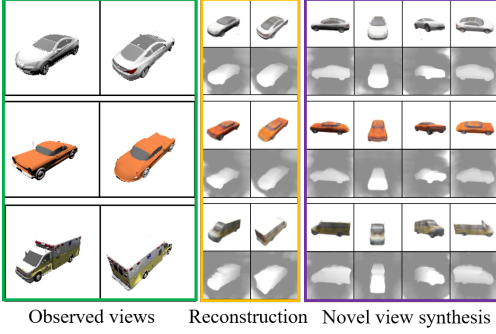
Observed views   Reconstruction   Novel view synthesis

Figure 3: Two-shot novel view synthesis results on (128×128) the ShapeNet Car testing set. The left two columns displays two views of unseen cars in testing set. The middle two columns show the reconstruction results obtained by a model trained on training set. The right four columns shows novel view synthesis for the unseen cars. For each reconstructed or synthesized image, we show an RGB image at the first row and the inverse depth map at the second row.

infer its latent appearance and shape vectors via 600 steps of MCMC guided by the posterior distribution, and then we generate novel views of the same object with randomly sampled camera poses. The qualitative results of novel view synthesis are shown in Figure 3. In Table 2, we compare our NeRF-LEBM with baselines, such as GQN [10] and the NeRF-VAE, in terms of PSNR. Our model has the best performance. We also demonstrate the generative ability of our model by showing the generated images in Figure 6. For a quantitative comparison, the FID for our NeRF-LEBM is 94.486 while the one obtained by the NeRF-VAE is 112.465.

Table 2: Comparing the NeRF-LEBM with other baselines on two-shot novel view synthesis of unseen objects in terms of PSNR. Models are tested on the ShapeNet Car dataset.

Generative Models	PSNR ↑
GQN [10]	18.79
NeRF-VAE [21]	18.37
NeRF-LEBM (MCMC inference)	**20.28**

4.5 Learning from incomplete 2D observations

To show the advantage of the MCMC-based inference, we test our model on a task where observations are incomplete or masked. To create a dataset for this task, we firstly randomly select 500 cars from the original ShapeNet Car dataset, and then for each of them, we use 50 views for training and 200 views for testing. For each training image, we randomly mask an area with Gaussian noise (see Figure 4). To enable our model to learn from incomplete data, we only maximize the data likelihood computed on the unmasked areas of the training data. This only leads to a minor

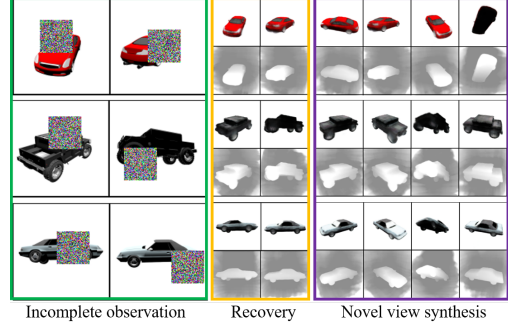Incomplete observation   Recovery   Novel view synthesis

Figure 4: Learning from incomplete 2D observations. The left two columns show some examples of the incomplete observations in the training set. The middle two columns show the corresponding recovery results obtained by the learning algorithm. The right four columns are synthesis results for unobserved views of the same objects. For each recovery or synthesized result, we show an RGB 2D image at the first row and an inverse depth map at the second row.

modification in Algorithm 1 involving the computation of $||\mathbf{I} - G_\theta(\mathbf{z}^a, \mathbf{z}^s, \boldsymbol{\xi})||^2$ in the likelihood term. For a partially observed image, we compute it by summing over only the visible pixels. The latent variables can still be inferred by explaining the visible parts of the incomplete observations, and the model can still be updated as before. In each iteration, we feed our model with two randomly selected observations of each object. Qualitative results in Figure 4 demonstrate that our algorithm can learn from incomplete images while recovering the missing pixels, and the learned model can still perform novel view synthesis. We quantitatively compare our NeRF-LEBM with the NeRF-VAE in Table 3. Our model beats the baseline using the same generator in the tasks of novel view synthesis and image generation.

Table 3: Comparison on tasks of novel view synthesis and image generation after learning the models from incomplete 128×128 2D observations on a masked ShapeNet Car dataset. PSNRs and FIDs are reported to measure the model performance on the two tasks, respectively.

Model	PSNR ↑	FID ↓
NeRF-VAE	21.26	128.27
NeRF-LEBM	**24.95**	**105.82**

4.6 Learning with unknown camera poses

We study the scenario of training the NeRF-LEBM from 2D images without known camera poses. We assume the camera to locate on a sphere and the object is in the center of this sphere. Thus, we need to infer the altitude and azimuth angles for each observed image. As discussed in Section 3.4, we introduce a camera pose inference network
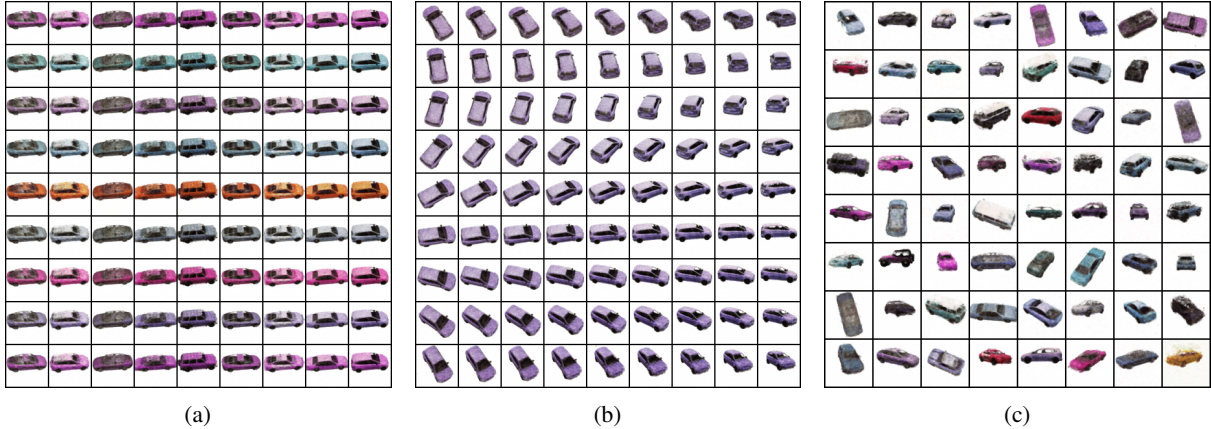
(a)   (b)   (c)

Figure 5: Synthesis results on 64×64 Carla dataset without camera poses. (a) The objects in each row share the same appearance vector $\mathbf{z}^a$ but have different shape vectors $\mathbf{z}^s$ while the objects in each column share the same $\mathbf{z}^s$ but have different $\mathbf{z}^a$. They all share the same camera pose $\boldsymbol{\xi}$. (b) The Effect of changing the camera pose while fixing the shape and appearance vectors for a sampled object. (c) Generated samples by randomly sampling $\mathbf{z}^a$, $\mathbf{z}^s$ and camera pose $\boldsymbol{\xi}$.

Figure 6: Synthesis by the NeRF-LEBM trained on the ShapeNet Car data, with persistent chain MCMC inference.

to approximate the posterior of the latent camera pose of each observation, and train EBM priors, NeRF-based generator and inference networks simultaneously via amortized inference. In practice, we let the inference networks for camera pose $\boldsymbol{\xi}$, appearance $\mathbf{z}^a$ and shape $\mathbf{z}^s$ to share the lower layers and only differ in their prediction heads. We carried out an experiment on the Carla dataset with a resolution of 64×64. Unlike in Section 4.2, here we only use the rendered images and do not use the ground truth camera pose associated with each image. The results on are shown in Figure 5. From Figure 5a, we can see the learned model can disentangle shape and appearance factors. Figure 5b shows that the learned model can factor out the camera pose from the data in an unsupervised manner. Figure 5c shows some random synthesized examples generated from the model by randomly sampling $\mathbf{z}^a$, $\mathbf{z}^s$ and $\boldsymbol{\xi}$. Although the camera pose can take a valid value from $[0, 2\pi)$, the altitude and azimuth angles of the training examples in the Carla dataset might only lie in limited ranges. Thus, after training, we estimate

a camera pose distribution using 10,000 training examples and when we generate images, we sample camera poses according to this distribution. Please refer to Figure 13 in Appendix for more results on the CelebA dataset. These results verify that our model can learn meaningful 3D-aware 2D image generator without known camera poses.

5 Conclusion and limitations

This paper pushes forward the progress of development of likelihood-based generative radiance field models for disentangled representation by proposing the NeRF-LEBM framework, in which we build informative and trainable energy-based priors for latent variables of object appearance and object shape on top of a NeRF-based top-down 2D image generator, and presenting two maximum likelihood learning algorithms, one with MCMC-based inference and the other with amortized inference. We formulate the NeRF-LEBM framework in two scenarios. One is training with known camera poses and the other is with unknown camera poses. Through several tasks we show that NeRF-LEBM can generate meaningful samples, infer 3D structure from 2D observations, learn to infer from incomplete 2D observation, and even learn from images with unknown camera poses. However, our current model also have some limitations. First, despite its accuracy and memory efficiency, the MCMC-based inference is indeed time-consuming. Thus we have to resort to amortized inference for efficient computation. Secondly, as for training from images with unknown camera poses, we find that it is still changeling for our model to obtain comparable generation performance with GAN-based models that can get around the inference step. Also, the training with unknown camera poses using amortized inference is still unstable and difficult. We will continue to improve our model in the future work.

Acknowledgements

References

[1] Hassan Abu Alhaija, Siva Karthik Mustikovela, Andreas Geiger, and Carsten Rother. Geometric image synthesis. In *Proceedings of the 14th Asian Conference on Computer Vision (ACCV), Part VI*, pages 85–100, Perth, Australia, 2018.

[2] Dongsheng An, Jianwen Xie, and Ping Li. Learning deep latent variable models by short-run MCMC inference with optimal transport correction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15415–15424, 2021.

[3] Adrian Barbu and Song-Chun Zhu. *Monte Carlo Methods*, volume 35. Springer, 2020.

[4] Samarth Brahmbhatt, Jinwei Gu, Kihwan Kim, James Hays, and Jan Kautz. Geometry-aware learning of maps for camera localization. In *Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2616–2625, Salt Lake City, UT, 2018.

[5] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5799–5809, virtual, 2021.

[6] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, and Fisher Yu. ShapeNet: An information-rich 3D model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[7] Tim R. Davidson, Luca Falorsi, Nicola De Cao, Thomas Kipf, and Jakub M. Tomczak. Hyperspherical variational auto-encoders. In *Proceedings of the Thirty-Fourth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 856–865, Monterey, CA, 2018.

[8] Alexey Dosovitskiy, Germán Ros, Felipe Codevilla, Antonio M. López, and Vladlen Koltun. CARLA: an open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning (CoRL)*, pages 1–16, Mountain View, CA, 2017.

[9] Vincent Dumoulin, Ethan Perez, Nathan Schucher, Florian Strub, Harm de Vries, Aaron Courville, and Yoshua Bengio. Feature-wise transformations. *Distill*, 3(7):e11, 2018.

[10] SM Ali Eslami, Danilo Jimenez Rezende, Frederic Besse, Fabio Viola, Ari S Morcos, Marta Garnelo, Avraham Ruderman, Andrei A Rusu, Ivo Danihelka, Karol Gregor, et al. Neural scene representation and rendering. *Science*, 360(6394):1204–1210, 2018.

[11] Ruiqi Gao, Yang Song, Ben Poole, Ying Nian Wu, and Diederik P. Kingma. Learning energy-based models by diffusion recovery likelihood. In *Proceedings of the 9th International Conference on Learning Representations (ICLR)*, Virtual Event, Austria, 2021.

[12] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2672–2680, Montreal, Canada, 2014.

[13] Tian Han, Yang Lu, Song-Chun Zhu, and Ying Nian Wu. Alternating back-propagation for generator network. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI)*, pages 1976–1984, 2017.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, Las Vegas, NV, 2016.

[15] Philipp Henzler, Niloy J. Mitra, and Tobias Ritschel. Escaping plato's cave: 3D shape from adversarial rendering. In *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9983–9992, Seoul, Korea, 2019.

[16] 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 *Advances in Neural Information Processing Systems (NIPS)*, pages 6626–6637, Long Beach, CA, 2017.

[17] James T. Kajiya and Brian Von Herzen. Ray tracing volume densities. In *Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, pages 165–174, Minneapolis, MN, 1984.

[18] Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. Neural 3D mesh renderer. In *Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3907–3916, Salt Lake City, UT, 2018.

[19] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, San Diego, CA, 2015.

[20] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*, Banff, Canada, 2014.

[21] Adam R. Kosiorek, Heiko Strathmann, Daniel Zoran, Pol Moreno, Rosalia Schneider, Sona Mokrá, and Danilo Jimenez Rezende. NeRF-VAE: A geometry aware 3D scene generative model. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, pages 5742–5752, Virtual Event, 2021.

[22] Shichen Liu, Weikai Chen, Tianye Li, and Hao Li. Soft rasterizer: A differentiable renderer for image-based 3D reasoning. In *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7707–7716, Seoul, Korea, 2019.

[23] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, pages 3730–3738, Santiago, Chile, 2015.

[24] Kantilal Varichand Mardia. Statistics of directional data. *Journal of the Royal Statistical Society: Series B (Methodological)*, 37(3):349–371, 1975.

[25] 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. *Commun. ACM*, 65(1):99–106, 2022.

[26] K L Navaneet, Priyanka Mandikal, Varun Jampani, and R. Venkatesh Babu. DIFFER: moving beyond 3D reconstruction with differentiable feature rendering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPR Workshops)*, pages 18–24, Long Beach, CA, 2019.

[27] Radford M Neal. MCMC using hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 2(11):2, 2011.

[28] Michael Niemeyer and Andreas Geiger. GIRAFFE: representing scenes as compositional generative neural feature fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11453–11464, virtual, 2021.

[29] Erik Nijkamp, Mitch Hill, Song-Chun Zhu, and Ying Nian Wu. Learning non-convergent nonpersistent short-run MCMC toward energy-based model. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5233–5243, Vancouver, Canada, 2019.

[30] Erik Nijkamp, Bo Pang, Tian Han, Linqi Zhou, Song-Chun Zhu, and Ying Nian Wu. Learning multi-layer latent variable model via variational optimization of short run MCMC for approximate inference. In *European Conference on Computer Vision (ECCV)*, volume 12351, pages 361–378, 2020.

[31] Michael Oechsle, Lars M. Mescheder, Michael Niemeyer, Thilo Strauss, and Andreas Geiger. Texture fields: Learning texture representations in function space. In *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4530–4539, Seoul, Korea, 2019.

[32] Bo Pang, Tian Han, Erik Nijkamp, Song-Chun Zhu, and Ying Nian Wu. Learning latent space energy-based prior model. In *Advances in Neural Information Processing Systems (NeurIPS) 2020*, virtual, 2020.

[33] Bo Pang and Ying Nian Wu. Latent space energy-based model of symbol-vector coupling for text generation and classification. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, pages 8359–8370, Virtual Event, 2021.

[34] Bo Pang, Tianyang Zhao, Xu Xie, and Ying Nian Wu. Trajectory prediction with latent belief energy-based model. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11814–11824, virtual, 2021.

[35] Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron C. Courville. FiLM: Visual reasoning with a general conditioning layer. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI)*, pages 3942–3951, New Orleans, LA, 2018.

[36] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. GRAF: generative radiance fields for 3D-aware image synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, virtual, 2020.

[37] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3D-structure-aware neural scene representations. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1119–1130, Vancouver, Canada, 2019.

[38] Xiaolong Wang and Abhinav Gupta. Generative image modeling using style and structure adversarial networks. In *Proceedings of the 14th European Conference on Computer Vision (ECCV), Part IV*, pages 318–335, Amsterdam, The Netherlands, 2016.

[39] Jianwen Xie, Ruiqi Gao, Zilong Zheng, Song-Chun Zhu, and Ying Nian Wu. Learning dynamic generator model by alternating back-propagation through time. In *The Thirty-Third AAAI Conference on Artificial Intelligence, (AAAI)*, pages 5498–5507, 2019.

[40] Jianwen Xie, Ruiqi Gao, Zilong Zheng, Song-Chun Zhu, and Ying Nian Wu. Motion-based generator model: Unsupervised disentanglement of appearance, trackable and intrackable motions in dynamic patterns. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, (AAAI)*, pages 12442–12451, 2020.

[41] Jianwen Xie, Yang Lu, Song-Chun Zhu, and Ying Nian Wu. A theory of generative ConvNet. In *Proceedings of the 33nd International Conference on Machine Learning (ICML)*, pages 2635–2644, New York City, NY, 2016.

[42] Jianwen Xie, Yifei Xu, Zilong Zheng, Song-Chun Zhu, and Ying Nian Wu. Generative PointNet: Deep energy-based learning on unordered point sets for 3D generation, reconstruction and classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14976–14985, virtual, 2021.

[43] Jianwen Xie, Zilong Zheng, Ruiqi Gao, Wenguan Wang, Song-Chun Zhu, and Ying Nian Wu. Learning descriptor networks for 3D shape synthesis and analysis. In *Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8629–8638, Salt Lake City, UT, 2018.

[44] Jianwen Xie, Zilong Zheng, Ruiqi Gao, Wenguan Wang, Song-Chun Zhu, and Ying Nian Wu. Generative VoxelNet: Learning energy-based models for 3D shape synthesis and analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 44(5):2468–2484, 2022.

[45] Jianwen Xie, Zilong Zheng, and Ping Li. Learning energy-based model with variational auto-encoder as amortized sampler. In *Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI)*, pages 10441–10451, 2021.

[46] Jianwen Xie, Song-Chun Zhu, and Ying Nian Wu. Synthesizing dynamic patterns by spatial-temporal generative ConvNet. In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1061–1069, Honolulu, HI, 2017.

[47] Jianwen Xie, Song-Chun Zhu, and Ying Nian Wu. Learning energy-based spatial-temporal generative ConvNets for dynamic patterns. *IEEE Trans. Pattern Anal. Mach. Intell.*, 43(2):516–531, 2021.

[48] Jianwen Xie, Yaxuan Zhu, Jun Li, and Ping Li. A tale of two flows: Cooperative learning of langevin flow and normalizing flow toward energy-based model. In *The Tenth International Conference on Learning Representations (ICLR)*, 2022.

[49] Jianwen Xie, Yaxuan Zhu, Yifei Xu, Dingcheng Li, and Ping Li. A tale of two latent flows: Learning latent space normalizing flow with short-run langevin flow for approximate inference. In *Thirty-Seventh AAAI Conference on Artificial Intelligence (AAAI)*, 2023.

[50] Peiyu Yu, Sirui Xie, Xiaojian Ma, Baoxiong Jia, Bo Pang, Ruiqi Gao, Yixin Zhu, Song-Chun Zhu, and Ying Nian Wu. Latent diffusion energy-based model for interpretable text modelling. In *International Conference on Machine Learning (ICML)*, Baltimore, Maryland, 2022.

[51] Jing Zhang, Jianwen Xie, and Nick Barnes. Learning noise-aware encoder-decoder from noisy labels by alternating back-propagation for saliency detection. In *European Conference on Computer Vision (ECCV)*, volume 12362, pages 349–366, 2020.

[52] Jing Zhang, Jianwen Xie, Nick Barnes, and Ping Li. Learning generative vision transformer with energy-based latent space for saliency prediction. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 15448–15463, virtual, 2021.

[53] Yang Zhao, Jianwen Xie, and Ping Li. Learning energy-based generative models via coarse-to-fine expanding and sampling. In *Proceedings of the 9th International Conference on Learning Representations (ICLR)*, Virtual Event, 2021.

[54] Zilong Zheng, Jianwen Xie, and Ping Li. Patchwise generative convnet: Training energy-based models from a single natural image for internal learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2961–2970, 2021.

[55] Jun-Yan Zhu, Zhoutong Zhang, Chengkai Zhang, Jiajun Wu, Antonio Torralba, Josh Tenenbaum, and Bill Freeman. Visual object networks: Image generation with disentangled 3D representations. In *Advances in Neural Information Processing Systems 31 (NeurIPS)*, pages 118–129, Montréal, Canada, 2018.

[56] Yizhe Zhu, Jianwen Xie, Bingchen Liu, and Ahmed Elgammal. Learning feature-to-feature translator by alternating back-propagation for generative zero-shot learning. In *International Conference on Computer Vision (ICCV)*, pages 9843–9853, 2019.

A Training details

In this section, we describe the network structure designs and hyper-parameter settings in our experiments.

A.1 Network structure

A.1.1 Conditional NeRF-based generator

For the experiments where camera poses of objects are given, the structure of the NeRF-based generator is shown in Figure 7. For this structure, we mainly follow the design in [36]. However, for each of the latent vectors $\mathbf{z}^s$ and $\mathbf{z}^a$, we add a mapping network to transform it before concatenating it with the positional embedding. This mapping network is composed of a normalizing layer and 4 linear layers, each of which is followed by a Leaky ReLU activation function. The concatenated features then enter the NeRF encoding module, which is an 8-layer MLP. The number of dimensions of each hidden layer or the output layer is 256. The output of the NeRF encoding module is then used to predict the color $\mathbf{c}$ and density σ information. For the experiments about learning without camera pose information on the CelebA dataset, inspired by [5], we design a network structure with the FiLM-conditioned layers [35, 9]. To be more specific, we input the transformed $\mathbf{z}^s$ to the first 4 layers of the generator and input the transformed $\mathbf{z}^a$ to the color head using the FiLM layer. The detailed architecture is shown in Figure 8.

A.1.2 Latent space EBM

We use two EBM structures. One (small version) contains 2 layers of linear transformations, each of which followed by a swish activation, while the other (large version) contains 4 layers of linear transformations with swish activation layers. The network structure is shown in Table 4. The choice of the EBM prior model for each experiment is shown in Table 5.

A.1.3 Inference model

In the experiments using amortized inference, we need an inference model to infer the latent vectors (i.e., appearance, shape, and camera pose). Inspired by [4], we use the ResNet34 [14] structure as the feature extractor and build separate inference heads on the top of it for different latent vectors. Each inference head is composed of a couple of MLP layers. We let the inference models for $\mathbf{z}^a$, $\mathbf{z}^s$ and camera pose to share the same feature extractor and only differ in their inference heads. When we update the parameters of the inference model, we use a learning rate 3×10^{-5} for the feature extractor and the pose head while using a smaller learning rate 5×10^{-6} for the inference heads of $\mathbf{z}^a$ and $\mathbf{z}^s$.

A.2 Training hyperparameters

During our training, we use the Adam[19] optimizer and we set $\beta_{\text{Adam}} = (0.9, 0.999)$. We set the standard deviation of the residual in Eq. (3) $\sigma_\epsilon = 0.05$. We disable the noise term in the Langevin inference for a better performance in the MCMC inference case. For the persistent chain MCMC inference setting, we use the Adam instead of a noise-disable Langevin dynamics to infer the latent vectors. As to the MCMC sampling for the EBM priors, we assign a weight as a hyperparameter to the noise term in the Langevin dynamics for adjusting its magnitude in our practise. Please check Table 6 for hyperparameter settings, including learning rate, batch size, etc, used in other experiments.
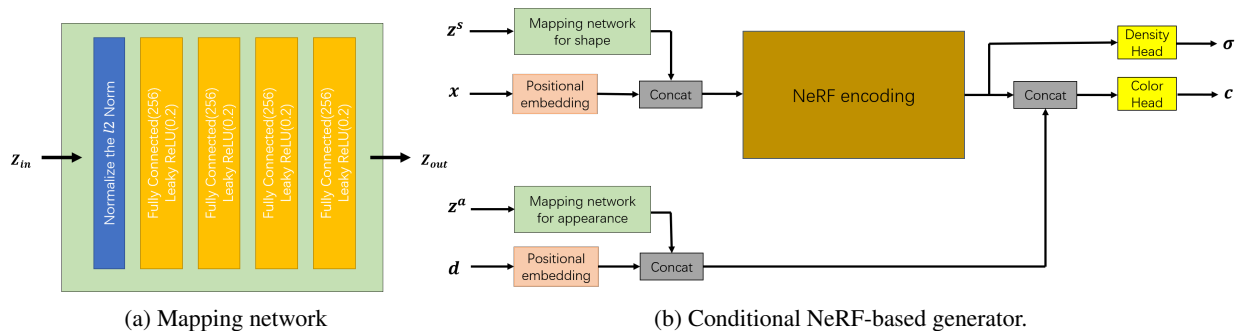

(a) Mapping network   (b) Conditional NeRF-based generator.

Figure 7: Model structure for the NeRF-LEBM generator with known camera pose.

(a) The FiLM layer.

(b) Mapping network.
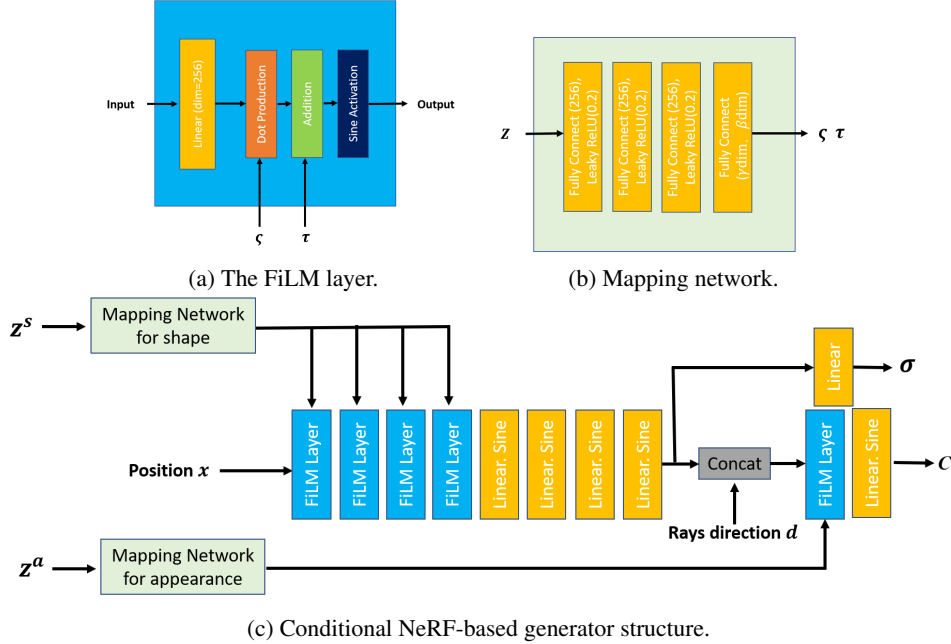

(c) Conditional NeRF-based generator structure.

Figure 8: Model structure for the NeRF-LEBM generator with unknown camera pose.

Table 4: Model structures of the latent EBMs with hidden dimension C.

(a) EBM (small)

Linear C, swish
Linear C, swish
Linear 1

(b) EBM (large)

Linear C, swish
Linear C, swish
Linear C, swish
Linear C, swish
Linear 1

Table 5: Choice of the EBM prior in each experiment.

Experiment	EBM prior for shape			EBM prior for appearance		
	type	vector dimension	hidden dimension	type	vector dimension	hidden dimension
Carla / MCMC inference / with poses	large	128	256	large	128	256
Carla / amortized inference / with poses	small	128	256	small	128	128
ShapeNet Car / MCMC inference / with poses	large	128	256	large	128	256
Carla / amortized inference / without poses	small	128	128	small	128	64

B Complexity analysis

We show a comparison of different models in terms of model size in Table 7a. By comparing our NeRF-LEBM using amortized inference with the NeRF-VAE which uses a Gaussian prior, we can find that introducing a latent EBM as a prior only slightly increase the number of parameters. If we rely on the MCMC of inference, then we can save a lot of parameters. In Table 7b, we compare the computational time of randomly sampling 100 images with a resolution of 128×128. The time is computed when the algorithm is run on a single NVIDIA RTX A6000 GPU. Comparing with the Gaussian prior, the EBM prior (with a 40-step MCMC sampling) barely affects the sampling time. That is because the latent EBM is much less computationally-intensive than the NeRF-based generator and we can sample 100 random variables in a batch altogether. We also compare the computational time of the amortized inference and MCMC inference in the few-shot inference of an unseen object in Table 7c, from which we can see that using an inference model can do a much quicker inference but the reconstruction results may be worse. On the other hand, MCMC inference may take some time but the results can be better.

Table 6: Hyperparameter settings in different experiments.

Experiements	batch size	Number of views	η_α	η_θ	η_ϕ	q_0	δ^+	K^+	δ^-	K^-	MCMC noise weight (EBM)
Carla / MCMC inference / with poses	8	1	2 e-5	1e-4	-	Uniform	0.1	60	0.5	60	0.02
Carla / amortized inference / with poses	8	1	7e-6	1e-4	1e-4	Normal	-	-	0.5	60	1.0
ShapeNet Car / persistent MCMC inference / with poses	12	2	2e-5	1e-4	-	Normal	1e-4	1	0.5	40	0.0
Carla / amortized inference / without poses	16	1	7e-6	3e-5	3e-5	Normal	-	-	0.5	60	0.0

Table 7: Complexity analysis

(a) Model size

Model	# parameters
NeRF-VAE	24.06 M
NeRF-LEBM (amortized)	24.52 M
NeRF-LEBM (MCMC)	2.21 M

(b) Synthesis time

Model	Time
EBM prior	40.97s
Gaussian prior	40.73s

(c) Inference time

Model	Time
Amortized inference	0.1s
MCMC inference	168s

C More generation results

C.1 More results for the experiment in Section 4.2 Random image synthesis

We show the generated examples from the NeRF-Gaussian-MCMC in Figure 9. Comparing the results in Figure 9 with those in Figure 1, we can see that examples obtained from the simple Gaussian prior have less diversity than those from the EBM prior. This is consistent with the numerical evaluation shown Table 1 and demonstrates the advantage of using latent EBMs as informative prior distributions for modeling latent variables.
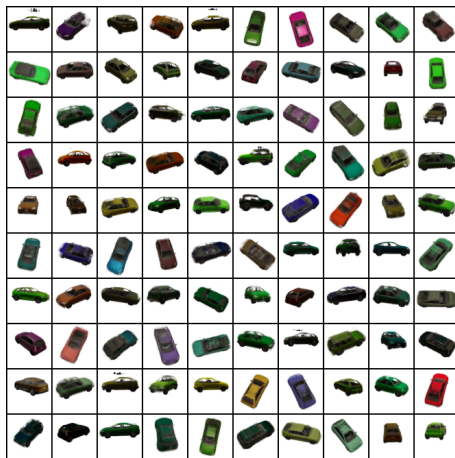

Figure 9: Images generated by the NeRF-Gaussian-MCMC baseline

C.2 More results for the experiment in Section 4.3 Disentangled representation

We show more synthesis results for the NeRF-LEBM with amortized inference on the Carla datasets in Figure 10, which are similar to Figure 2b. We can see those sampled cars are meaningful and their multi-view synthesis results are consistent. Then we show the generative results for the NeRF-LEBM with MCMC inference on the Carla dataset in Figure 11. The model can also correctly disentangle the appearance, shape and camera pose.
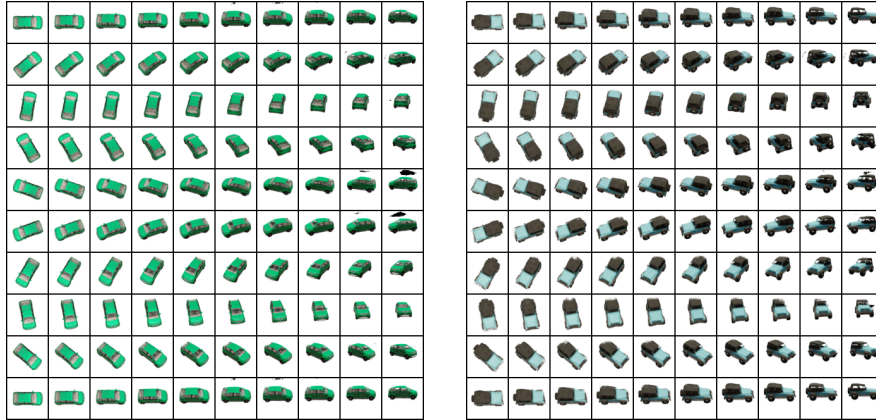
Figure 10: More generated examples by the NeRF-LEBM using amortized inference on the Carla dataset.

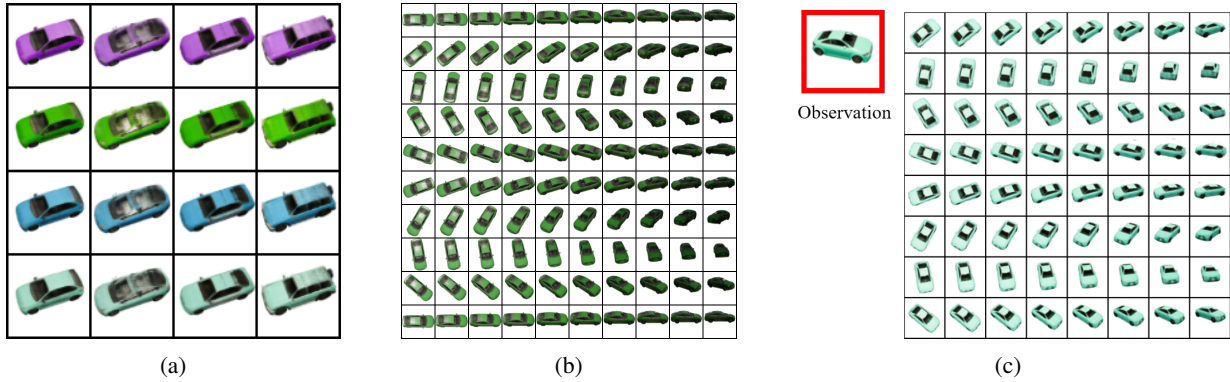(a)   (b)   (c)

Figure 11: Disentangled representation. The generated images are obtained by the learned NeRF-LEBM using MCMC inference on the Carla dataset. (a) shows the influences of the shape vector $\mathbf{z}^s$ and the appearance vector $\mathbf{z}^a$ in image synthesis. The objects in each row share the same appearance vector $\mathbf{z}^a$ and camera pose $\boldsymbol{\xi}$ but have different shape vectors $\mathbf{z}^s$, while the objects in each column share the same shape vector $\mathbf{z}^s$ and camera pose $\boldsymbol{\xi}$ but have different appearance vectors $\mathbf{z}^a$. (b) demonstrates the effect of the camera pose variable $\boldsymbol{\xi}$ by varying it while fixing the shape and appearance vectors for a randomly sampled object. (c) shows an example of novel view synthesis for an observed 2D image.

C.3   More results for the experiment in Section 4.4 Inferring 3D structures of unseen 2D objects

In the main paper, we show two-shot novel view synthesis results on the ShapeNet Car testing set in Figure 3. Here in Figure 12, we show the novel view synthesis results for cars in training image. More specifically, for each object in the training set, we use the inferred $\mathbf{z}^a$ and $\mathbf{z}^s$ from the seen views and synthesize unseen views that hasn't been used during training. Figure 12 presents three cases. For each case, the first row shows the ground truth images, and the second row displays the synthesized results by our model. Our model can correctly output unseen views when camera poses are given.

C.4   More results for the experiment in Section 4.6 Learning with unknown camera poses

In the main paper, we show the results on the synthetic dataset, Carla. We also work on a real world dataset. The CelebA dataset [23] contains 200k face images. Unlike the synthetic datasets, this dataset does not provide any camera pose information. We learn our model from this dataset without knowing the camera poses. We preprocess the face images by cropping a 128×128 patch from the center of each image and reshaping it into a 64×64 image. We show the results on CelebA dataset in Figure 13. We further show the generated examples obtained from a VAE baseline with a simple Gaussian prior in Figure 14. As we can see, our model can learn to generate meaningful samples without using the ground truth camera poses and the EBM prior provides much better generation results than the simple Gaussian prior.
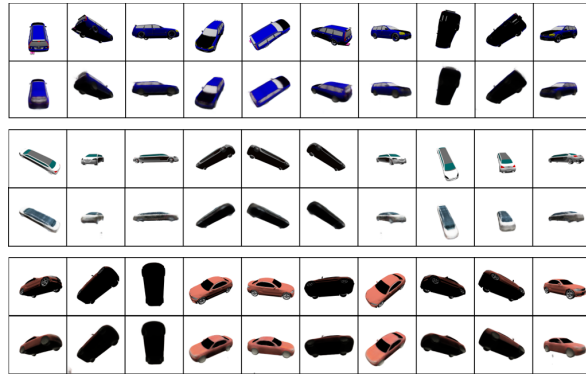
Figure 12: Novel view synthesis results for objects in the ShapeNet Car training set. Each panel shows one case, in which the first row shows the ground truth images while the second row displays the novel view synthesis results by our model.
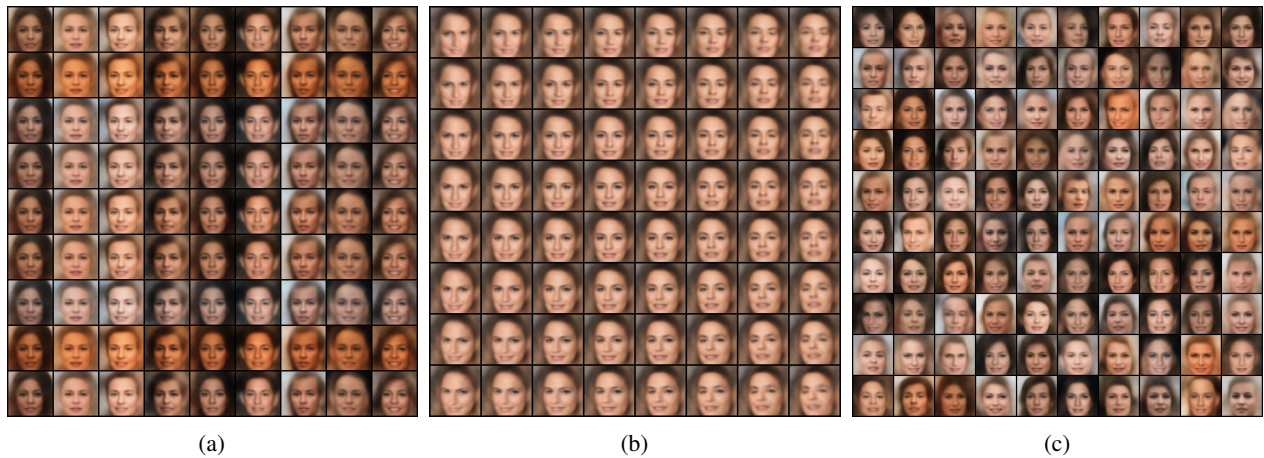

(a)   (b)   (c)

Figure 13: Synthesis results on 64×64 CelebA dataset without knowing camera poses. (a) The objects in each row share the same appearance vector $\mathbf{z}^a$ but have different shape vectors $\mathbf{z}^s$ while the objects in each column share the same $\mathbf{z}^s$ but have different $\mathbf{z}^a$. They all share the same camera pose $\boldsymbol{\xi}$. (b) The Effect of changing the camera pose while fixing the shape and appearance vectors for a sampled object. (c) Generated samples by randomly sampling $\mathbf{z}^a$, $\mathbf{z}^s$ and camera pose $\boldsymbol{\xi}$.

Figure 14: Images generated by a VAE baseline (Gaussian prior), trained on the CelebA dataset without camera poses.