

---

# Diffusion with Forward Models: Solving Stochastic Inverse Problems Without Direct Supervision

---

Ayush Tewari<sup>1\*</sup> Tianwei Yin<sup>1\*</sup> George Cazenavette<sup>1</sup> Semon Rezchikov<sup>4</sup>  
Joshua B. Tenenbaum<sup>1,2,3</sup> Frédéric Durand<sup>1</sup> William T. Freeman<sup>1</sup> Vincent Sitzmann<sup>1</sup>

<sup>1</sup>MIT CSAIL <sup>2</sup>MIT BCS <sup>3</sup>MIT CBMM <sup>4</sup>Princeton IAS

## Abstract

Denoising diffusion models have emerged as a powerful class of generative models capable of capturing the distributions of complex, real-world signals. However, current approaches can only model distributions for which training samples are directly accessible, which is not the case in many real-world tasks. In inverse graphics, for instance, we seek to sample from a distribution over 3D scenes consistent with an image but do not have access to ground-truth 3D scenes, only 2D images. We present a new class of conditional denoising diffusion probabilistic models that learn to sample from distributions of signals that are never observed directly, but instead are only measured through a known differentiable forward model that generates partial observations of the unknown signal. To accomplish this, we directly integrate the forward model into the denoising process. At test time, our approach enables us to sample from the distribution over underlying signals consistent with some partial observation. We demonstrate the efficacy of our approach on three challenging computer vision tasks. For instance, in inverse graphics, we demonstrate that our model in combination with a 3D-structured conditioning method enables us to directly sample from the distribution of 3D scenes consistent with a single 2D input image.

## 1 Introduction

Consider the problem of reconstructing a 3D scene from a single picture. Since much of the 3D scene is unobserved, there are an infinite number of 3D scenes that could have produced the image, due to the 3D-to-2D projection, occlusion, and limited field-of-view that leaves a large part of the 3D scene unobserved. Given the ill-posedness of this problem, it is desirable for a reconstruction algorithm to be able to sample from the distribution over all plausible 3D scenes that are consistent with the 2D image, generating unseen parts in plausible manners. Previous data-completion methods, such as in-painting in 2D images, are trained on large sets of ground-truth output images along with their incomplete (input) counterparts. Such techniques do not easily extend to 3D scene completion, since curating a large dataset of ground-truth 3D scene representations is very challenging.

This 3D scene completion problem, known as inverse graphics, is just one instance of a broad class of problems often referred to as *Stochastic Inverse Problems*, which arise across scientific disciplines whenever we capture partial observations of the world through a sensor. In this paper, we introduce a diffusion-based framework that can tackle this problem class, enabling us to sample from a distribution of signals that are consistent with a set of partial observations that are generated from the signal by a non-invertible, generally nonlinear, forward model. For instance, in inverse graphics, we learn to sample 3D scenes given an image, yet never observe paired observations of images and 3D scenes at training time, nor observe 3D scenes directly.

---

\* Equal Contribution. Project page: [diffusion-with-forward-models.github.io](https://diffusion-with-forward-models.github.io)

While progress in deep learning for generative modeling has been impressive, this problem remains unsolved. In particular, variational autoencoders and conditional neural processes are natural approaches but have empirically fallen short of modeling the multi-modal distributions required in, for instance, inverse graphics. They have so far been limited to simple datasets. Emerging diffusion models [1], in contrast, enable sampling from highly complex conditional distributions but require samples from the output distribution that is to be modeled for training, e.g. full 3D models. Some recent work in inverse graphics has resorted to a two-stage approach, where one first reconstructs a large dataset of 3D scenes to then train an image-conditional diffusion model to sample from the conditional distribution over these scenes [2, 3]. To avoid a two-stage approach, another recent line of work trains a conditional diffusion model to sample from the distribution over novel views of a scene, only requiring image observations at training time [4, 5]. However, such methods do *not* model the distribution over 3D scenes directly and therefore cannot sample from the distribution over 3D scenes consistent with an image observation. Thus, a multi-view consistent 3D scene can only be obtained in a costly post-processing stage [6]. A notable exception is the recently proposed RenderDiffusion [7], demonstrating that it is possible to train an unconditional diffusion model over 3D scenes from observing only monocular images. While one can perform conditional sampling even with unconditional models, they are fundamentally limited to simple distributions, in this case, single objects in canonical orientations.

Our core contribution is a novel approach for integrating any differentiable forward model that describes how partial observations are obtained from signals, such as 2D image observations and 3D scenes, with conditional denoising diffusion models. By sampling an observation from our model, we jointly sample the signal that gave rise to that observation. Our approach has a number of advantages that make it highly attractive for solving complex Stochastic Inverse Problems. First, our model is trained end-to-end and does away with two-stage approaches that first require reconstruction of a large dataset of signals. Second, our model directly yields diverse samples of the signal of interest. For instance, in the inverse graphics setting, our model directly yields highly diverse samples of 3D scenes consistent with an observation that can then be rendered from novel views with guaranteed multi-view consistency. Finally, our model naturally leverages domain knowledge in the form of known forward models, such as differentiable rendering, with all guarantees that such forward models provide. We validate our approach on three challenging computer vision tasks: inverse graphics (the focus of this paper), as well as single-image motion prediction and GAN inversion.

In summary, we make the following contributions:

1. We propose a new method that integrates differentiable forward models with conditional diffusion models, replacing prior two-step approaches with a conditional generative model trained end-to-end.
2. We apply our framework to build the first conditional diffusion model that learns to sample from the distribution of 3D scenes trained only on 2D images. In contrast to prior work, we *directly* learn image-conditional 3D radiance field generation, instead of sampling from the distribution of novel views conditioned on a context view. Our treatment of inverse graphics exceeds a mere application of the proposed framework, contributing a novel, 3D-structured denoising step that leverages differentiable rendering both for conditioning and for the differentiable forward model.
3. We formally prove that under natural assumptions, as the number of observations of each signal in the training set goes to infinity, the proposed model maximizes not only the likelihood of observations, but also the likelihood of the unobserved signals.
4. We demonstrate the efficacy of our model for two more downstream tasks with structured forward models: single-image motion prediction, where the forward model is a warping operation, and GAN inversion, where the forward model is a pretrained StyleGAN [8] generator.

## 2 Method

Consider observations  $(\mathbf{O}_j^i, \phi_j^i)$  that are generated from underlying signals  $\mathbf{S}_j$  according to a known forward model `forward()`, i.e.,  $\mathbf{O}_j^i = \text{forward}(\mathbf{S}_j, \phi_j^i)$ , where  $\phi_j^i$  are parameters of the forward model corresponding to observation  $\mathbf{O}_j^i$ . Each observation can be *partial*. Specifically, given a *single* observation, there is an infinite number of signals that could have generated this observation. However, we assume that given a hypothetical set of *all possible* observations, the signal is fully

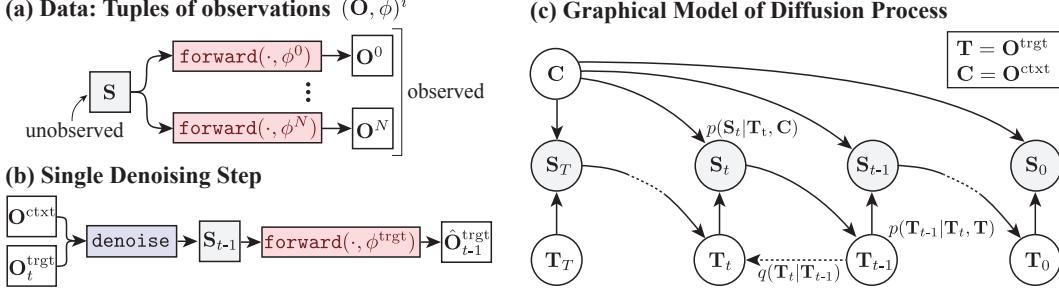


Figure 1: **Overview of our proposed method.** (a) We assume a dataset of tuples of observations  $(\mathbf{O}, \phi)^i$ , generated from *unobserved* signals  $\mathbf{S}$  via a differentiable forward model. (b) We propose to integrate the forward model directly into the denoising step of a diffusion model: given a pair of observations of the same signal, we designate context  $\mathbf{O}^{\text{ctxt}}$  and target  $\mathbf{O}^{\text{trgt}}$ . We add noise to  $\mathbf{O}^{\text{trgt}}$ , then feed  $(\mathbf{O}^{\text{ctxt}}, \phi^{\text{ctxt}}, \mathbf{O}_t^{\text{trgt}}, \phi^{\text{trgt}})$  to a neural network `denoise` to estimate the signal  $\mathbf{S}_{t-1}$ . We then apply the forward model to obtain an estimate of the clean target observation,  $\hat{\mathbf{O}}_{t-1}^{\text{trgt}}$ . (c) The graphical model of the diffusion process.

determined. In the case of inverse graphics,  $\mathbf{O}_j^i$  are image observations of 3D scenes  $\mathbf{S}_j$  and  $\phi_j^i$  are the camera parameters, where we index scenes with  $j$  and observations of the  $j$ -th scene via  $i$ . `forward()` is the rendering function. Note that if we were to capture *every possible image* of a 3D scene, the 3D scene is uniquely determined, but given a *single* image, there are an infinite number of 3D scenes that could have generated that image, both due to the fact that rendering is a projection from 3D and 2D, and due to the fact that a single image only constrains the visible part of the 3D scene. We will drop the subscript  $j$  in the following, and leave it implied that we always consider *many* observations generated from *many* signals. Fig. 1 provides an illustration of the data.

We are now interested in training a model that, at test time, allows us to sample from the distribution of signals that are consistent with a previously unseen observation  $\mathbf{O}$ . Formally, we aim to model the conditional distribution  $p(\mathbf{S}|\mathbf{O}, \phi)$ . We make the following assumptions:

- We have access to a differentiable implementation of `forward()`.
- We have access to a large dataset of observations and corresponding parameters of the forward model,  $\{(\mathbf{O}^i, \phi^i)\}_i^N$ .
- In our training set, we have access to *several* observations per signal.

Crucially, we do *not* assume that we have direct access to the underlying signal that gave rise to a particular observation, i.e., we do *not* assume access to tuples of  $(\mathbf{O}, \phi, \mathbf{S})$ . Further, we also do *not* assume that we have access to any prior distribution over the signal of interest, i.e., we never observe a dataset of signals of the form  $\{\mathbf{S}^j\}_j$ , and thus cannot train a generative model to sample from an unconditional distribution over signals.

Recent advances in deep-learning-based generative modeling have seen the emergence of denoising diffusion models as powerful generative models that can be trained to generate highly diverse samples from complex, multi-modal distributions. We are thus motivated to leverage denoising diffusion probabilistic models to model  $p(\mathbf{S}|\mathbf{O}, \phi)$ . However, existing approaches cannot be trained if we do not have access to signals  $\mathbf{S}$ . In the following, we give background on denoising diffusion models and discuss the limitation.

## 2.1 Background: Denoising Diffusion Probabilistic Models and their Limitation

Denoising diffusion probabilistic models are a class of generative models that learn to sample from a distribution by learning to iteratively denoise samples. Consider the problem of modeling the distribution  $p_\theta(\mathbf{x})$  over samples  $\mathbf{x}$ . A forward Markovian process  $q(\mathbf{x}_{0:T})$  adds noise to the data as

$$q(\mathbf{x}_t | \mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1 - \beta_t} \mathbf{x}_{t-1}, \beta_t \mathbf{I}). \quad (1)$$

Here,  $\beta_t$ ,  $t \in 1 \dots T$  are the hyperparameters that control the variance schedule. A denoising diffusion model learns the reverse process, where samples from a distribution  $p(\mathbf{x}_T) = \mathcal{N}(\mathbf{0}, \mathbf{I})$  are transformed incrementally into the data manifold as  $p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T) \prod_{t=1}^T p_\theta(\mathbf{x}_{t-1} | \mathbf{x}_t)$ , where

$$p_\theta(\mathbf{x}_{t-1} | \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu(\mathbf{x}_t, t), \Sigma(\mathbf{x}_t, t)). \quad (2)$$

A neural network `denoise` with learnable parameters  $\theta$  learns to reverse the diffusion process. It is also possible to model conditional distributions  $p_\theta(\mathbf{x}_{0:T} | \mathbf{c})$ , where the output is computed as

$\text{denoise}_\theta(\mathbf{x}_t, t, \mathbf{c})$ . The forward process does not change in this case; in practice, we merely add the conditional signal as input to the denoising model.

Unfortunately, we cannot train existing denoising diffusion models to sample from  $p(\mathbf{S} | \mathbf{O}, \phi)$ , or, in fact, even from an unconditional distribution  $p(\mathbf{S})$ . This would require computation of the Markovian forward process in Eq. 1. However, recall that we do not have access to any signals  $\{\mathbf{S}^j\}_j$  - we thus can not add any noise to any signals to then train a denoising neural network. In other words, since no  $\mathbf{S}$  is directly observed, we *cannot* compute  $q(\mathbf{S}_t | \mathbf{S}_{t-1})$ .

## 2.2 Integrating Denoising Diffusion with Differentiable Forward Models

We now introduce a class of denoising diffusion models that we train to directly model the distribution  $p(\mathbf{S} | \mathbf{O}^{\text{ctxt}}; \phi^{\text{ctxt}})$  over signals  $\mathbf{S}$  given an observation  $(\mathbf{O}^{\text{ctxt}}, \phi^{\text{ctxt}})$ . Our key contribution is to directly integrate the differentiable forward model `forward()` into the iterative conditional denoising process. This enables us to add noise to and denoise the observations, while nevertheless sampling the underlying signal that generates that observation.

Our model is trained on pairs of “context” and “target” observations of the same signal, denoted as  $\mathbf{O}^{\text{ctxt}}$  and  $\mathbf{O}^{\text{trgt}}$ . As in conventional diffusion models, for the forward process, we have  $q(\mathbf{O}_t^{\text{trgt}} | \mathbf{O}_{t-1}^{\text{trgt}}) = \mathcal{N}(\mathbf{O}_t^{\text{trgt}}; \sqrt{1 - \beta_t} \mathbf{O}_{t-1}^{\text{trgt}}, \beta_t \mathbf{I})$ . In the reverse process, we similarly denoise  $\mathbf{O}^{\text{trgt}}$  conditional on  $\mathbf{O}^{\text{ctxt}}$ :

$$p_\theta(\mathbf{O}_{0:T}^{\text{trgt}} | \mathbf{O}^{\text{ctxt}}; \phi^{\text{ctxt}}, \phi^{\text{trgt}}) = p(\mathbf{O}_T^{\text{trgt}}) \prod_{t=0}^T p_\theta(\mathbf{O}_{t-1}^{\text{trgt}} | \mathbf{O}_t^{\text{trgt}}, \mathbf{O}^{\text{ctxt}}; \phi^{\text{ctxt}}, \phi^{\text{trgt}}), \quad (3)$$

However, unlike conventional diffusion models, we implement  $p_\theta(\mathbf{O}_{t-1}^{\text{trgt}} | \mathbf{O}_t^{\text{trgt}}, \mathbf{O}^{\text{ctxt}}; \phi^{\text{ctxt}}, \phi^{\text{trgt}})$  by first predicting an estimate of the underlying signal  $\mathbf{S}_{t-1}$  and then mapping it to an estimate of the denoised observations via the differentiable `forward`:

$$\mathbf{S}_{t-1} = \text{denoise}_\theta(\mathbf{O}^{\text{ctxt}}, \mathbf{O}_t^{\text{trgt}}; t, \phi^{\text{ctxt}}, \phi^{\text{trgt}}), \quad (4)$$

$$\hat{\mathbf{O}}_{t-1}^{\text{trgt}} = \text{forward}(\mathbf{S}_{t-1}, \phi^{\text{trgt}}) \quad (5)$$

$$\mathbf{O}_{t-1}^{\text{trgt}} \sim \mathcal{N}(\hat{\mathbf{O}}_{t-1}^{\text{trgt}}; C_{t-1} \hat{\mathbf{O}}_{t-1}^{\text{trgt}}, \hat{\beta}_{t-1} \mathbf{I}) \quad (6)$$

Here,  $\hat{\mathbf{O}}_{t-1}^{\text{trgt}}$  is an estimate of the *clean* observation, and the constants  $C_{t-1}$  and  $\hat{\beta}_{t-1}$  are chosen to match the total noise added by the forward process at time  $t-1$ . See Fig. 1 for an overview. At test time, a signal is sampled by iterating Eq. 4, 5, and 6 starting with  $p(\mathbf{O}_{t=T}^{\text{trgt}}) \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ . Importantly, our models define a generative model over the underlying signal via Eq. 4:

$$p_{\theta, \phi^{\text{trgt}}}(\mathbf{S}_{0:T} | \mathbf{O}^{\text{ctxt}}; \phi^{\text{ctxt}}) = \prod_{t=1}^T p_\theta(\mathbf{S}_{t-1} | \mathbf{O}_t^{\text{trgt}}, \mathbf{O}^{\text{ctxt}}; \phi^{\text{ctxt}}, \phi^{\text{trgt}}). \quad (7)$$

We will suppress the subscript in the notation, and refer to this distribution as  $p(\mathbf{S}_{0:T} | \mathbf{O}^{\text{ctxt}}; \phi^{\text{ctxt}})$  for brevity from now.

**Loss Function.** We train to minimize the following two loss terms:

$$\mathcal{L}_\theta^{\text{trgt}} = \mathbb{E}_{\mathbf{O}^{\text{ctxt}}, \mathbf{O}^{\text{trgt}}, \phi^{\text{ctxt}}, \phi^{\text{trgt}}, t} \left[ \left\| \mathbf{O}^{\text{trgt}} - \underbrace{\text{forward}(\text{denoise}_\theta(\mathbf{O}^{\text{ctxt}}, \mathbf{O}_t^{\text{trgt}}; t, \phi^{\text{ctxt}}, \phi^{\text{trgt}}), \phi^{\text{trgt}})}_{= \hat{\mathbf{O}}_{t-1}^{\text{trgt}}} \right\|^2 \right], \quad (8)$$

$$\mathcal{L}_\theta^{\text{novel}} = \mathbb{E}_{\mathbf{O}^{\text{ctxt}}, \mathbf{O}^{\text{novel}}, \phi^{\text{ctxt}}, \phi^{\text{trgt}}, \phi^{\text{novel}}, t} \left[ \left\| \mathbf{O}^{\text{novel}} - \underbrace{\text{forward}(\text{denoise}_\theta(\mathbf{O}^{\text{ctxt}}, \mathbf{O}_t^{\text{trgt}}; t, \phi^{\text{ctxt}}, \phi^{\text{trgt}}), \phi^{\text{novel}})}_{= \hat{\mathbf{O}}_{t-1}^{\text{novel}}} \right\|^2 \right]. \quad (9)$$

Here, we compute the estimate of the observation from the target, as well as a separate, novel forward model parameter  $\phi^{\text{novel}}$ . In the supplemental document, we show that these losses approximate a total observation loss, maximizing the likelihood of all possible observations of the signal  $\mathbf{S}$ .

**Characterizing the Conditional Distribution Over Signals.** Due to the complexity of the reverse process, it may not be clear that the learned distribution over signals will agree with the true distribution, even in the limit of infinite data. However, this model will indeed asymptotically learn the true conditional distribution over signals, as we formally prove in the supplement:

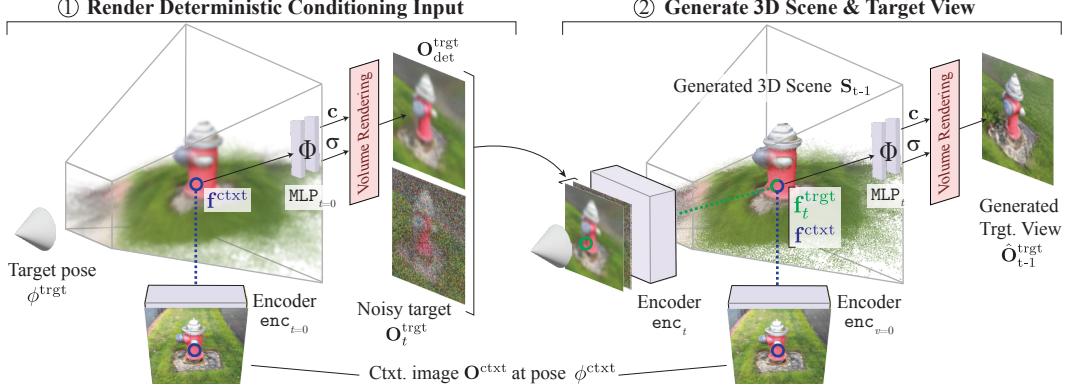


Figure 2: **Overview of 3D Generative Modeling.** We build a 3D-structured denoise operator on top of pixelNeRF [9] that learns to sample from the distribution of 3D scenes from image observations only. Given a context image  $O^{ctxt}$  with camera pose  $\phi^{ctxt}$ , we pick a target pose  $\phi^{trgt}$ . We render out a deterministic estimate of the depth, RGB, and features of the target view  $O^{trgt}$  using pixel-aligned features  $f^{ctxt}$  extracted from the context view with encoder  $enc_{t=0}$  (left, only RGB shown here). To generate a 3D scene, we concatenate the deterministic estimate with noise  $O^{trgt}$ , and extract features  $f^{trgt}$  for the *target* view with  $enc_t$ .  $f^{trgt}$  and  $f^{ctxt}$  now jointly parameterize the radiance field of the generated scene  $S_{t-1}$ , and we may render an estimate of the clean target view  $\hat{O}_{t-1}^{trgt}$ . The model is trained end-to-end via a re-rendering loss.

**Proposition 1.** Suppose that any signal  $S$  can be reconstructed from the set of all *all possible* observations of  $S$ . Under this assumption, if in the limit as the number of known observations per signal goes to infinity, there are parameters  $\theta$  such that  $\mathcal{L}_\theta^{trgt} + \mathcal{L}^{novel}$  is minimized, then the conditional probability distribution over signals discovered by our model  $p(S | O^{ctxt}; \phi^{ctxt})$  agrees with the true distribution  $p^{true}(S | O^{ctxt}; \phi^{ctxt})$ .

The proof follows by showing that our losses implicitly minimize a diffusion model loss over *total observations*, which are collections of all possible observations of our signal. As such, when the observations suffice to completely reconstruct the signal, the correctness of the estimated distribution over total observations forces the estimated distribution over signals to be correct, as well.

### 3 Prior Work on Latent Variable Models for Inverse Problems

Variational Autoencoders [10, 11], normalizing flows [12], conditional [13] and attentive neural processes [14] are latent-variable models that can be combined with forward models to learn to sample from the distribution of unobserved signals from observations [15, 16]. However, they empirically fall short of accurately modeling complex signal distributions - in inverse graphics, for instance, such models have so far been limited to synthetic 3D scenes. Generative Adversarial Networks can be trained with differentiable forward models in-the-loop, and have yielded impressive results in unconditional generative modeling of unobserved signals [17–19]. Similarly, in concurrent work, diffusion models have been leveraged for unconditional generative modeling through differentiable forward models [2, 7, 20]. However, unconditional models are limited to tight distributions, and no conditional generative modeling of similar quality has been demonstrated. Diffusion models trained directly on signals have been effectively applied to diverse inverse problems such as super-resolution [21–24], inpainting [21, 23–25], and medical imaging [26]. These works utilize the learned prior of the data distribution to recover the latent signal through a “plug and play” approach [27–29], integrating the diffusion model with a forward measurement process according to Bayes’ rule. These approaches are versatile and can easily adapt to new inverse problems without retraining. However, unlike our models, they rely on direct supervision over the signals in the form of large datasets.

### 4 Applications

We now apply our framework to three stochastic inverse problems. We focus on applications in computer vision, where we tackle the problems of inverse graphics, single-image motion prediction, and GAN inversion. For each application, we give a detailed description of the forward model, the dataset and baselines, as well as a brief description of prior work.

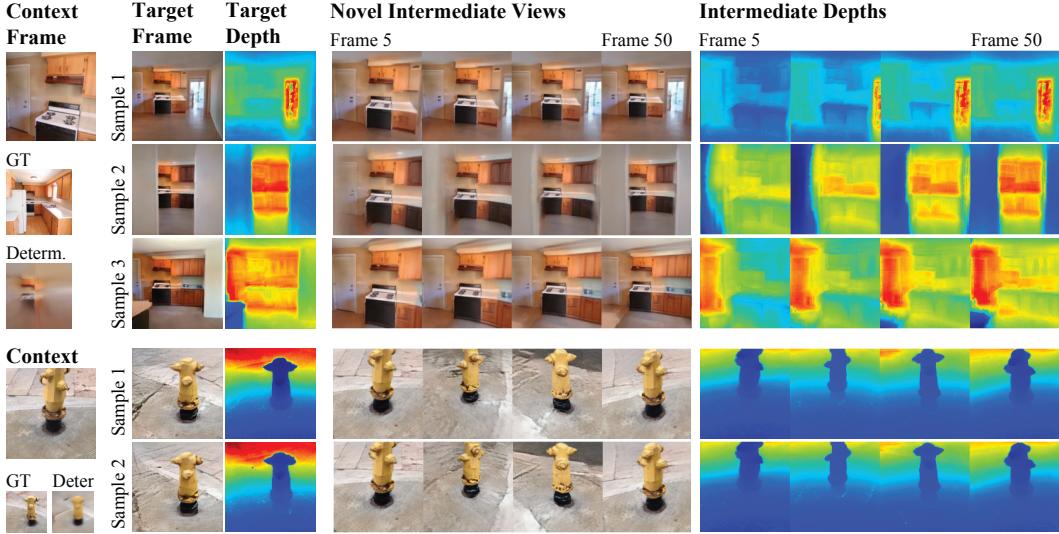


Figure 3: **Sample Diversity.** We illustrate different 3D scenes sampled from the same context image for RealEstate10k and Co3D datasets. Unlike deterministic methods like pixelNeRF [9], our method generates diverse and distinct 3D scenes that all align with the context image. Co3D results are generated using autoregressive sampling, where a 360 degree trajectory can be generated by iteratively sampling target images. Note the photorealism and diversity of the generated structures for the indoor scene, such as doors and cabinets. Also note the high-fidelity geometry of the occluded parts of the hydrant and the diverse background appearance.

#### 4.1 Inverse Graphics

We seek to learn a model that, given a single image of a 3D scene enables us to sample from the distribution over 3D scenes that are consistent with the observation. We expect that 3D regions visible in the image are reconstructed faithfully, while unobserved parts are generated plausibly. Every time we sample, we expect a *different* plausible 3D generation. Signals  $\mathbf{S}$  are 3D scenes, and observations are 2D images  $\mathbf{O}$  and their camera parameters  $\phi$ . At training time, we assume that we have access to at least two image observations and their camera parameters per scene, such that we can assemble tuples of  $(\mathbf{O}^{\text{ctx}}, \phi^{\text{ctx}}, \mathbf{O}^{\text{trgt}}, \phi^{\text{trgt}})$ , with 2D images  $\mathbf{O}^{\text{ctx}}$ ,  $\mathbf{O}^{\text{trgt}}$ , and camera parameters  $\phi^{\text{ctx}}, \phi^{\text{trgt}}$ .

**Scope.** We note that our treatment of inverse graphics exceeds a mere application of the presented framework. In particular, we not only integrate the differentiable rendering `forward` function, but further propose a novel 3D-structured `denoise` function. Here, we enable state-of-the-art conditional generation of complex, real-world 3D scenes.

**Related Work.** Few-shot reconstruction of 3D scene representations via differentiable rendering was pioneered by deterministic methods [9, 30, 31, 31–40] that blur regions of the 3D scene unobserved in the context observations. Probabilistic methods have been proposed that can sample from the distribution of novel views trained only on images [4, 5, 41–44]. While results are impressive, these methods do not allow sampling from the distribution of *3D scenes*, but only from the distribution of *novel views*. Generations are not multi-view consistent. Obtaining a 3D scene requires costly post-processing via score distillation [6]. Several approaches [2, 3] use a two-stage design: they first reconstruct a dataset of 3D scenes, and then train a 3D diffusion model. However, pre-computing large 3D datasets is expensive. Further, to obtain high-quality results, dense observations are required per scene. RenderDiffusion [7] and HoloDiffusion [20] integrate differentiable forward rendering with an unconditional diffusion model, enabling unconditional sampling of simple, single-object scenes. Similar to us, RenderDiffusion performs denoising in the image space, while HoloDiffusion uses a 3D denoising architecture. Other methods use priors learned by text-conditioned image diffusion models to optimize 3D scenes [45–47]. Here, the generative model does not have explicit knowledge about the 3D information of scenes. These methods often suffer from geometric artifacts.

**Structure of  $\mathbf{S}$  and `forward` model `render`.** We can afford only an abridged discussion here – please see the supplement for a more detailed description. We use NeRF [48] as the parameterization of 3D scenes, such that  $\mathbf{S}$  is a function that maps a 3D coordinate  $\mathbf{p}$  to a color  $\mathbf{c}$  and density  $\sigma$  as  $\mathbf{S}(\mathbf{p}) = (\sigma, \mathbf{c})$ . We require a *generalizable* NeRF that is predicted in a feed-forward pass by an encoder that takes a set of  $M$  context images and corresponding camera poses  $\{(\mathbf{O}_i, \phi_i)\}_i^M$  as input.

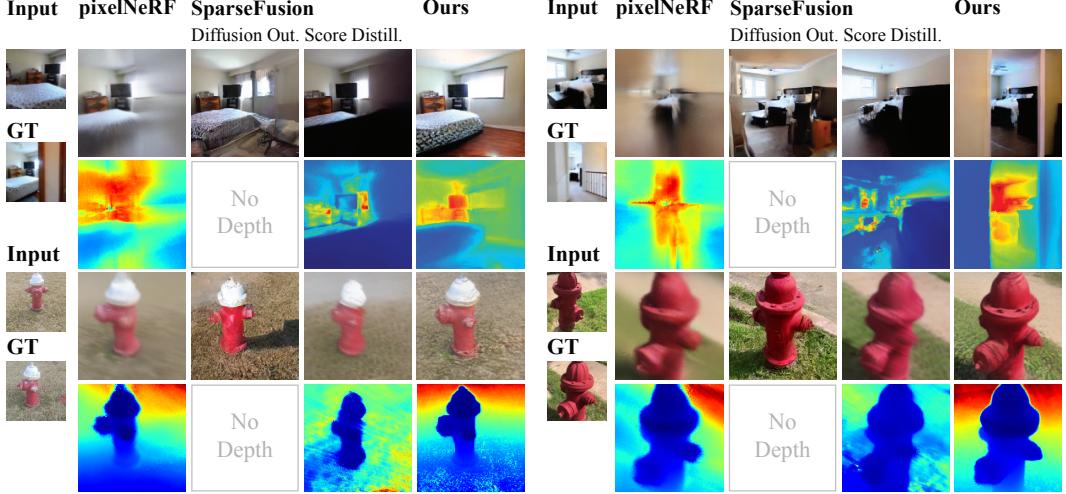


Figure 4: **Qualitative Comparison for Inverse Graphics application.** We benchmark with SparseFusion [5] and the deterministic pixelNeRF [9]. SparseFusion samples 2D novel views conditioned on a deterministic rendering (Diffusion Out.), and generates multi-view consistent 3D scenes only after Score Distillation. Our method consistently generates higher-quality scenes, while directly sampling 3D scenes.

We base our model on pixelNeRF [9]. pixelNeRF first extracts image features  $\{\mathbf{F}_i\}_i$  from each context observation via an encoder  $\text{enc}$  as  $\mathbf{F}_i = \text{enc}(\mathbf{O}_i)$ . Given a 3D point  $\mathbf{p}$ , it obtains its pixel coordinates in each context view via  $\mathbf{p}_i^{\text{pix}} = \pi(\mathbf{p}, \phi_i)$  via the projection operator  $\pi$ , and recovers a corresponding feature as  $\mathbf{f}_i = \mathbf{F}_i(\mathbf{p}_i^{\text{pix}})$  by sampling the feature map at pixel coordinate  $\mathbf{p}_i^{\text{pix}}$ . It then parameterizes  $\mathbf{S}$  via an MLP as:

$$\mathbf{S}(\mathbf{p}) = (\sigma(\mathbf{p}), \mathbf{c}(\mathbf{p})) = \text{MLP}(\{\mathbf{f}_i \oplus \mathbf{p}_i\}_i^M), \quad (10)$$

where  $\oplus$  is concatenation and  $\mathbf{p}_i$  is the 3D point  $\mathbf{p}$  transformed into the camera coordinates of observation  $i$ . The number of context images  $M$  is flexible, and we may condition  $\mathbf{S}$  on a single or several observations. It will be convenient to refer to a pixelNeRF that is reconstructed from context and target observations  $(\mathbf{O}^{\text{ctx}}, \phi^{\text{ctx}})$  and  $(\mathbf{O}^{\text{trgt}}, \phi^{\text{trgt}})$  as

$$\mathbf{S}(\cdot | \text{enc}(\mathbf{O}^{\text{ctx}}), \text{enc}(\mathbf{O}^{\text{trgt}})), \quad (11)$$

where we make the pixelNeRF encoder  $\text{enc}$  explicit and drop the poses  $\phi^{\text{trgt}}$  and  $\phi^{\text{ctx}}$ . We leverage differentiable volume rendering [48] as forward model, such that

$$\mathbf{O} = \text{render}(\mathbf{S}, \phi), \quad (12)$$

where  $\mathbf{S}$  is rendered from a camera with parameters  $\phi$ .

**Implementation of denoise.** Fig. 2 gives an overview of the denoising procedure. Following our framework, we obtain the denoised target observation  $\hat{\mathbf{O}}_{t-1}^{\text{trgt}}$  as:

$$\hat{\mathbf{O}}_{t-1}^{\text{trgt}} = \text{render}(\mathbf{S}_{t-1}, \phi^{\text{trgt}}), \quad \text{where} \quad (13)$$

$$\mathbf{S}_{t-1} = \mathbf{S}(\cdot | \text{enc}_{t=0}(\mathbf{O}^{\text{ctx}}), \text{enc}_t(\mathbf{O}_t^{\text{trgt}})), \quad (14)$$

where the image encoder  $\text{enc}_t$  is now conditioned on the timestep  $t$ . In other words, we will generate a target view  $\hat{\mathbf{O}}_{t-1}^{\text{trgt}}$  by rendering the pixelNeRF conditioned on the context and noisy target observations. However, feeding the noisy  $\mathbf{O}_t^{\text{trgt}}$  directly to pixelNeRF is insufficient. This is because the pixel-aligned features  $\text{enc}_t(\mathbf{O})$  are obtained from each view separately - thus, the features generated by  $\text{enc}_t(\mathbf{O}_t^{\text{trgt}})$  will be uninformative. To successfully generate a 3D scene, we have to augment the  $\mathbf{O}_t^{\text{trgt}}$  with information from the context view. We propose to generate conditioning information for  $\mathbf{O}_t^{\text{trgt}}$  by rendering a *deterministic estimate*  $\mathbf{O}_{\text{det}}^{\text{trgt}} = \text{render}(\mathbf{S}(\cdot | \text{enc}_{t=0}(\mathbf{O}^{\text{ctx}})), \phi^{\text{trgt}})$ . I.e., we condition pixelNeRF only on the context view, and render an estimate of the target view via volume rendering. However, in the extreme case of a completely uncertain target view, this results in a completely blurry image. We thus propose to additionally render high-dimensional features. Recall that any 3D point  $\mathbf{p}$ , we have  $(\sigma(\mathbf{p}), \mathbf{c}(\mathbf{p})) = \text{MLP}_t(\mathbf{p})$ . We modify  $\text{MLP}_t$  to also output a high-dimensional feature and

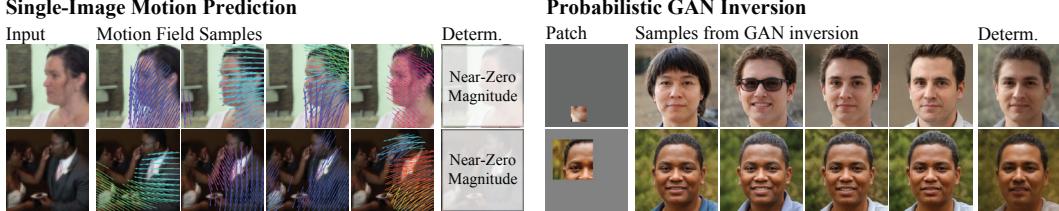


Figure 5: Qualitative Results for Single-Image Motion Prediction (left) and GAN Inversion (right).

render a deterministic feature map to augment  $\mathbf{O}_t^{\text{trgt}}$  (only RGB shown in figure). We generate the final 3D scene as  $\mathbf{S}_{t-1} = \mathbf{S}(\cdot \mid \mathbf{enc}_{t=0}(\mathbf{O}^{\text{ctx}}), \mathbf{enc}_t(\mathbf{O}_{\text{det}}^{\text{trgt}} \oplus \mathbf{O}_t^{\text{trgt}}))$ . The final denoised target view is then obtained according to the rendering Eq. 13 above.

**Loss and Training.** Our loss consists of simple least-squares terms on re-rendered views, identical to the general loss terms presented in Eqs. 8 and 9, in addition to regularizers that penalize degenerate 3D scenes. We discuss these regularizers, as well as training details, in the supplement.

#### 4.1.1 Results

**Datasets** We evaluate on two challenging real-world datasets. We use Co3D hydrants [49] to evaluate our method on object-centric scenes. For scene-level 3D synthesis, we use the challenging RealEstate10k dataset [50], consisting of indoor and outdoor videos of scenes.

**Baselines** We compare our approach with state-of-the-art approaches in deterministic and probabilistic 3D scene completion. We use pixelNeRF as the representative method for deterministic methods that takes a single image as input and deterministically reconstructs a 3D scene. Our method is the first to probabilistically reconstruct 3D scenes in an end-to-end manner. Regardless, we compare with the concurrent SparseFusion [51] that learns an image-space generative model over novel views of a 3D scene. Score distillation of this generative model is required every time we want to obtain a multi-view consistent 3D scene, which is costly.

**Qualitative Results.** In Fig. 3, we show multiple samples of 3D scenes sampled from a monocular image. For the indoor scenes of RealEstate10k, there are large regions of uncertainty. We can sample from the distribution of valid 3D scenes, resulting in significantly different 3D scenes with plausible geometry and colors. The objects are faithfully reconstructed for the object-centric Co3D scenes, and the uncertainty in the scene is captured. We can sample larger 3D scenes and render longer trajectories by autoregressive sampling, i.e., we treat intermediate diffused images as additional context observations to sample another target observation. The Co3D results in Fig. 3 were generated autoregressively for a complete 360 degrees trajectory. In Fig. 4, we compare our results with pixelNeRF [9] and SparseFusion [5]. pixelNeRF is a deterministic method and thus leads to very blurry results in uncertain regions. SparseFusion reconstructs scenes by score-distillation over a 2D generative model. This optimization is very expensive, and does not lead to natural-looking results.

**Quantitative Results.** For the object-centric Co3D dataset, we evaluate the accuracy of novel views using PSNR and LPIPS [52] metrics. Note that PSNR/LPIPS are not meaningful metrics for large scenes since the predictions have a large amount of uncertainty, i.e., a wide range of novel view images can be consistent with any input image. Thus, we report FID [53] and KID [54] scores to evaluate the realism of reconstructions in these cases. Our approach outperforms all baselines for LPIPS, FID, and KID metrics, as our model achieves more realistic results. We achieve slightly lower PSNR compared to pixelNeRF [9]. Note that PSNR favors mean estimates, and that we only evaluate our model using a single randomly sampled scene for an input image due to computational constraints.

## 4.2 Single-Image Motion Prediction

Here, we seek to train a model that, given a single static image, allows us to sample from *all possible motions* of pixels in the image. Given, for instance, an image of a person performing a task, such as kicking a soccer ball, it is possible to predict potential future states. This is a stochastic problem, as there are multiple possible motions consistent with an image. We train on a dataset of natural videos [55]. We only observe RGB frames and never directly observe the underlying motion, i.e., the pixel correspondences in time are unavailable. We use tuples of two frames from videos within a small temporal window, and use them as our context and target observations for training.

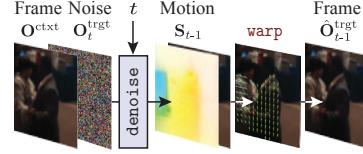
3D Scene Completion							GAN Inversion		
	Co3D				RealEstate10k		FFHQ		
	PSNR↑	LPIPS↓	FID↓	KID↓	FID↓	KID↓	Determ.	FID↓	KID↓
pixelNeRF	<b>17.93</b>	0.54	180.20	0.14	195.40	0.14			
SparseFusion	12.06	0.63	252.13	0.16	99.44	0.04	Ours	<b>7.45</b>	<b>0.002</b>
Ours	17.47	<b>0.42</b>	<b>84.63</b>	<b>0.05</b>	<b>42.84</b>	<b>0.01</b>			

Table 1: **Quantitative evaluation.** (left) We benchmark our 3D generative model with state-of-the-art baselines pixelNeRF [9] and SparseFusion [5]. (right) We benchmark with a deterministic baseline on GAN inversion, which we drastically outperform.

**Related Work.** Several papers tackle this problem, where motion in the form of optical flow [56–58], 2D trajectories [59, 60], and human motion [61, 62] are recovered from a static image; however, all these methods assume supervision over the underlying motion. Learning to reason about motion requires the neural network to learn about the properties and behavior of the different objects in the world. Thus, this serves as a useful proxy task for representation learning, and can be used as a backbone for many downstream applications [59, 63].

**Structure of S and forward model warp.** Our signal  $\mathbf{S}$  stores the appearance and motion information in a 2D grid. At any pixel  $\mathbf{u}$ , the signal is defined as  $\mathbf{S}(\mathbf{u}) = (\mathbf{S}_c(\mathbf{u}), \mathbf{S}_m(\mathbf{u}))$ , where  $\mathbf{S}_c(\mathbf{u}) \in \mathbb{R}^3$  is the color value, and  $\mathbf{S}_m(\mathbf{u}) \in \mathbb{R}^2$  is a 2D motion vector. The forward model is a warping operator, such that  $\text{warp}(\mathbf{S}, \phi)(\mathbf{u} + \phi \mathbf{S}_m(\mathbf{u})) = \mathbf{S}_c(\mathbf{u})$  and  $\phi$  is a scalar that changes the magnitude of motion. We implement this function using a differentiable point splatting operation [64].

**Implementation of denoise.** The inset figure illustrates our design. We use a 2D network that takes  $\mathbf{O}^{\text{ctxt}}$ ,  $\mathbf{O}_t^{\text{trgt}}$ , and  $t$  as input, and generates the motion map  $\mathbf{S}_m$  as the output. The signal is then reconstructed as  $\mathbf{S} = (\mathbf{O}^{\text{ctxt}}, \mathbf{S}_m)$ . Context and target frames correspond to parameters  $\phi^{\text{ctxt}} = 0$  and  $\phi^{\text{trgt}} = 1$ , and can be reconstructed from the signal using  $\text{warp}$ .



**Loss and Evaluation.** Similar to inverse graphics, we use reconstruction and regularization losses. The reconstruction losses are identical to Eqs. 8 and 9, and the regularization loss is a smoothness term that encourages a natural motion of the scene, see supplement for details. We show results in Fig. 5 (left), where we can estimate a diverse set of possible motion flows from monocular images. By smoothly interpolating  $\phi$ , we can generate short video sequences, even though our model only saw low-framerate video frames during training. We also train a deterministic baseline, which only generates a single motion field. Due to the amount of uncertainty in this problem, the deterministic estimate collapses to a near-zero motion field regardless of the input image, and thus, fails to learn any meaningful features from images.

### 4.3 GAN Inversion

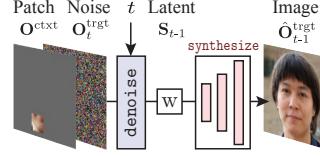
Projecting images onto the latent space of generative adversarial networks is a well-studied problem [8, 65], and enables interesting applications, as manipulating latents along known directions allows a user to effectively edit images [66–68]. Here, we solve the problem of projecting partial images: given a small visible patch in an image, our goal is to model the distribution of possible StyleGAN2 [8] latents that agree with the input patch. There are a diverse set of latents that can correspond to the input observation, and we train our method without observing supervised (image, latent) pairs. Instead, we train on pairs of  $(\mathbf{O}^{\text{ctxt}}, \mathbf{O}^{\text{trgt}})$  observations, where  $\mathbf{O}^{\text{ctxt}}$  are the small patches in images, and  $\mathbf{O}^{\text{trgt}}$  are the full images.

**Related Work.** While most GAN inversion methods focus on inverting a complete image into the generator’s latent space [69–77], some also reconstruct GAN latents from small patches via supervised training. Inversion is not trivial, and papers often rely on regularization [76] or integrate the inversion with editing tasks [78] for higher quality. We also integrate the inpainting task with the inversion, and seek to model the uncertainty of the GAN inversion task given only a partial observation (patch) of the target image.

**Structure of S and forward model synthesize.** Our signal  $\mathbf{S} \in \mathbb{R}^{512}$  is a 512 dimensional latent code representing the “w” space of StyleGAN2 [8] trained on the FFHQ [79] dataset. The forward model  $\text{synthesize}(\mathbf{S}, \phi) = \text{GAN}(\mathbf{S})[\phi]$  first reconstructs the image corresponding to  $\mathbf{S}$  using a

forward pass of the GAN. It then extracts a patch using the forward model’s parameters  $\phi$  that encode the patch coordinates.

**Implementation of denoise, Loss, and Evaluation.** Please see the inset figure for an illustration of the method. The denoising network receives  $O^{ctx}$ ,  $O_t^{tgt}$ , and timestep  $t$  as input, and generates an estimate of the StyleGAN latent  $w$ . The loss function is identical to Eq. 8 and compares the reconstructed sample with ground truth. We show results in Fig. 5 (right). We obtain diverse samples that are all consistent with the input patch. We also compare with a deterministic baseline that minimizes the same loss but only produces a single estimate. While this deterministic estimate also agrees with the input image, it does not model the diversity of outputs. We consequently achieve significantly better FID [53] and KID [54] scores than the deterministic baseline, reported in Tab. 1 (right).



## 5 Discussion

**Limitations.** While our method makes significant advances in generative modeling, it still has several limitations. Sampling 3D scenes at test time can be very slow, due to the expensive nature of the denoising process and the cost of volume rendering. We need multi-view observations of training scenes for the inverse graphics application. Our models are not trained on very large-scale datasets, and can thus not generalize to out-of-distribution data.

**Conclusion** We have introduced a new method that tightly integrates differentiable forward models and conditional diffusion models. Our model learns to sample from the distribution of signals trained only using their observations. We demonstrate the efficacy of our approach on three challenging computer vision problems. In inverse graphics, our method, in combination with a 3D-structured conditioning method, enables us to directly sample from the distribution of real-world 3D scenes consistent with a single image observation. We can then render multi-view consistent novel views while obtaining diverse samples of 3D geometry and appearance in unobserved regions of the scene. We further tackle single-image conditional motion synthesis, where we learn to sample from the distribution of 2D motion conditioned on a single image, as well as GAN inversion, where we learn to sample images that exist in the latent space of a GAN that are consistent with a given patch. With this work, we make contributions that broaden the applicability of state-of-the-art generative modeling to a large range of scientifically relevant applications, and hope to inspire future research in this direction.

**Acknowledgements.** This work was supported by the National Science Foundation under Grant No. 2211259, by the Singapore DSTA under DST00OECI20300823 (New Representations for Vision), by the NSF award 1955864 (Occlusion and Directional Resolution in Computational Imaging), by the ONR MURI grant N00014-22-1-2740, and by the Amazon Science Hub. We are grateful for helpful conversations with members of the Scene Representation Group David Charatan, Cameron Smith, and Boyuan Chen. We thank Zhizhuo Zhou for thoughtful discussions about the SparseFusion baseline. This article solely reflects the opinions and conclusions of its authors and no other entity.

**Author contributions.** Ayush and Vincent conceived the idea of diffusion with forward models, designed experiments, generated most figures, and wrote most of the paper. Ayush contributed the key insight to integrate differentiable rendering with diffusion models by denoising in image space while generating 3D scenes. Ayush and Vincent generalized this to general forward models, and conceived the single-image motion application. Vincent contributed the 3D-structured conditioning and generated the overview and methods figures. Ayush wrote all initial code and ran all initial experiments. Ayush and Tianwei implemented the inverse graphics application and generated most of the 3D results of our model, while George helped with the baseline 3D results. Ayush executed all single-image motion experiments. George conceived, implemented, and executed all GAN inversion experiments. Semon helped formalizing the method and wrote the proposition and its proof. Frédo and Bill were involved in regular meetings and gave valuable feedback on results and experiments. Josh provided intriguing cognitive science perspectives and feedback on results and experiments, and provided a significant part of the compute. Vincent’s Scene Representation Group provided a significant part of the compute, and the project profited from code infrastructure developed by and conversations with other members of the Scene Representation Group.

## References

- [1] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proc. ICML*, 2015. [2](#)
- [2] Norman Müller, , Yawar Siddiqui, Lorenzo Porzi, Samuel Rota Bulò, Peter Kontschieder, and Matthias Nießner. Diffrrf: Rendering-guided 3d radiance field diffusion. *Proc. CVPR*, 2023. [2](#), [5](#), [6](#)
- [3] Seung Wook Kim, Bradley Brown, Kangxue Yin, Karsten Kreis, Katja Schwarz, Daiqing Li, Robin Rombach, Antonio Torralba, and Sanja Fidler. Neuralfield-ldm: Scene generation with hierarchical latent diffusion models. *Proc. CVPR*, 2023. [2](#), [6](#)
- [4] Eric R Chan, Koki Nagano, Matthew A Chan, Alexander W Bergman, Jeong Joon Park, Axel Levy, Miika Aittala, Shalini De Mello, Tero Karras, and Gordon Wetzstein. Generative novel view synthesis with 3d-aware diffusion models. *arXiv preprint arXiv:2304.02602*, 2023. [2](#), [6](#)
- [5] Zhizhuo Zhou and Shubham Tulsiani. Sparsefusion: Distilling view-conditioned diffusion for 3d reconstruction. *Proc. CVPR*, 2023. [2](#), [6](#), [7](#), [8](#), [9](#)
- [6] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *Proc. ICLR*, 2023. [2](#), [6](#)
- [7] Titas Auciukevičius, Zexiang Xu, Matthew Fisher, Paul Henderson, Hakan Bilen, Niloy J Mitra, and Paul Guerrero. Renderdiffusion: Image diffusion for 3d reconstruction, inpainting and generation. *Proc. CVPR*, 2023. [2](#), [5](#), [6](#)
- [8] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proc. CVPR*, 2020. [2](#), [9](#)
- [9] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *Proc. CVPR*, 2021. [5](#), [6](#), [7](#), [8](#), [9](#)
- [10] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *Proc. ICLR*, 2014. [5](#)
- [11] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *Proc. ICML*, 2014. [5](#)
- [12] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *Proc. ICML*, 2015. [5](#)
- [13] Marta Garnelo, Dan Rosenbaum, Christopher Maddison, Tiago Ramalho, David Saxton, Murray Shanahan, Yee Whye Teh, Danilo Rezende, and SM Ali Eslami. Conditional neural processes. In *Proc. ICML*, 2018. [5](#)
- [14] Hyunjik Kim, Andriy Mnih, Jonathan Schwarz, Marta Garnelo, Ali Eslami, Dan Rosenbaum, Oriol Vinyals, and Yee Whye Teh. Attentive neural processes. *Proc. ICLR*, 2019. [5](#)
- [15] 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 *Proc. ICML*, 2021. [5](#)
- [16] Pol Moreno, Adam R Kosiorek, Heiko Strathmann, Daniel Zoran, Rosalia G Schneider, Björn Winckler, Larisa Markeeva, Théophane Weber, and Danilo J Rezende. Laser: Latent set representations for 3d generative modeling. *arXiv preprint arXiv:2301.05747*, 2023. [5](#)
- [17] 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 *Proc. CVPR*, 2021. [5](#)
- [18] Jun Gao, Tianchang Shen, Zian Wang, Wenzheng Chen, Kangxue Yin, Daiqing Li, Or Litany, Zan Gojcic, and Sanja Fidler. Get3d: A generative model of high quality 3d textured shapes learned from images. *Proc. NeurIPS*, 2022. [5](#)
- [19] Terrance DeVries, Miguel Angel Bautista, Nitish Srivastava, Graham W Taylor, and Joshua M Susskind. Unconstrained scene generation with locally conditioned radiance fields. In *Proc. ICCV*, 2021. [5](#)
- [20] Animesh Karnewar, Andrea Vedaldi, David Novotny, and Niloy Mitra. Holodiffusion: Training a 3d diffusion model using 2d images. *Proc. CVPR*, 2023. [5](#), [6](#)
- [21] Hyungjin Chung, Jeongsol Kim, Michael Thompson Mccann, Marc Louis Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. In *Proc. ICLR*, 2023. [5](#)

- [22] Jooyoung Choi, Sungwon Kim, Yonghyun Jeong, Youngjune Gwon, and Sungroh Yoon. Ilvr: Conditioning method for denoising diffusion probabilistic models. *Proc. ICCV*, 2021. 5
- [23] Bahjat Kawar, Michael Elad, Stefano Ermon, and Jiaming Song. Denoising diffusion restoration models. In *Proc. NeurIPS*, 2022. 5
- [24] Jiaming Song, Arash Vahdat, Morteza Mardani, and Jan Kautz. Pseudoinverse-guided diffusion models for inverse problems. In *Proc. ICLR*, 2023. 5
- [25] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *Proc. ICLR*, 2021. 5
- [26] Yang Song, Liyue Shen, Lei Xing, and Stefano Ermon. Solving inverse problems in medical imaging with score-based generative models. *Proc. ICLR*, 2022. 5
- [27] Johnathan M Bardsley. Mcmc-based image reconstruction with uncertainty quantification. *SIAM Journal on Scientific Computing*, 34(3):A1316–A1332, 2012. 5
- [28] Singanallur Venkatakrishnan, Charles A. Bouman, and Brendt Wohlberg. Plug-and-play priors for model based reconstruction. *2013 IEEE Global Conference on Signal and Information Processing*, pages 945–948, 2013. 5
- [29] Yaniv Romano, Michael Elad, and Peyman Milanfar. The little engine that could: Regularization by denoising (red). *SIAM Journal on Imaging Sciences*, 10(4):1804–1844, 2017. 5
- [30] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. *Proc. NeurIPS*, 2019. 6
- [31] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *Proc. CVPR*, 2020. 6
- [32] Philipp Henzler, Jeremy Reizenstein, Patrick Labatut, Roman Shapovalov, Tobias Ritschel, Andrea Vedaldi, and David Novotny. Unsupervised learning of 3d object categories from videos in the wild. In *Proc. CVPR*, 2021. 6
- [33] Prafull Sharma, Ayush Tewari, Yilun Du, Sergey Zakharov, Rares Andrei Ambrus, Adrien Gaidon, William T Freeman, Fredo Durand, Joshua B Tenenbaum, and Vincent Sitzmann. Neural groundplans: Persistent neural scene representations from a single image. In *Proc. ICLR*. 6
- [34] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In *Proc. ICCV*, 2021. 6
- [35] Yilun Du, Cameron Smith, Ayush Tewari, and Vincent Sitzmann. Learning to render novel views from wide-baseline stereo pairs. In *Proc. CVPR*, 2023. 6
- [36] Alex Trevithick and Bo Yang. Grf: Learning a general radiance field for 3d representation and rendering. In *Proc. ICCV*, 2021. 6
- [37] Mohammed Suhail, Carlos Esteves, Leonid Sigal, and Ameesh Makadia. Generalizable patch-based neural rendering. In *Proc. ECCV*, 2022. 6
- [38] Julian Chibane, Aayush Bansal, Verica Lazova, and Gerard Pons-Moll. Stereo radiance fields (srf): Learning view synthesis for sparse views of novel scenes. In *Proc. CVPR*, 2021. 6
- [39] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul P Srinivasan, Howard Zhou, Jonathan T Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. Ibrnet: Learning multi-view image-based rendering. In *Proc. CVPR*, 2021. 6
- [40] Shamit Lal, Mihir Prabhudesai, Ishita Mediratta, Adam W Harley, and Katerina Fragkiadaki. Coconets: Continuous contrastive 3d scene representations. In *Proc. CVPR*, 2021. 6
- [41] Daniel Watson, William Chan, Ricardo Martin-Brualla, Jonathan Ho, Andrea Tagliasacchi, and Mohammad Norouzi. Novel view synthesis with diffusion models. *Proc. ICLR*, 2023. 6
- [42] 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. 6
- [43] Hung-Yu Tseng, Qinbo Li, Changil Kim, Suhib Alsisan, Jia-Bin Huang, and Johannes Kopf. Consistent view synthesis with pose-guided diffusion models. *arXiv preprint arXiv:2303.17598*, 2023. 6

- [44] Jiatao Gu, Qingzhe Gao, Shuangfei Zhai, Baoquan Chen, Lingjie Liu, and Josh Susskind. Learning controllable 3d diffusion models from single-view images. *arXiv preprint arXiv:2304.06700*, 2023. 6
- [45] Luke Melas-Kyriazi, Christian Rupprecht, Iro Laina, and Andrea Vedaldi. Realfusion: 360  $\{\backslash\deg\}$  reconstruction of any object from a single image. *Proc. CVPR*, 2023. 6
- [46] Lukas Höllerin, Ang Cao, Andrew Owens, Justin Johnson, and Matthias Nießner. Text2room: Extracting textured 3d meshes from 2d text-to-image models. *arXiv preprint arXiv:2303.11989*, 2023. 6
- [47] Rafail Fridman, Amit Abecasis, Yoni Kasten, and Tali Dekel. Scenescape: Text-driven consistent scene generation. *arXiv preprint arXiv:2302.01133*, 2023. 6
- [48] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *Proc. ECCV*, 2020. 6, 7
- [49] Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotny. Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction. In *Proc. ICCV*, 2021. 8
- [50] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 37, 2018. 8
- [51] Yi Ding, Alex Rich, Mason Wang, Noah Stier, Matthew Turk, Pradeep Sen, and Tobias Höllerer. Sparse fusion for multimodal transformers. *arXiv preprint arXiv:2111.11992*, 2021. 8
- [52] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proc. CVPR*, 2018. 8
- [53] 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. *Proc. NeurIPS*, 2017. 8, 10
- [54] Mikołaj Bińkowski, Danica J Sutherland, Michael Arbel, and Arthur Gretton. Demystifying mmd gans. *arXiv preprint arXiv:1801.01401*, 2018. 8, 10
- [55] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T Freeman. Video enhancement with task-oriented flow. *International Journal of Computer Vision*, 127:1106–1125, 2019. 8
- [56] Ruohan Gao, Bo Xiong, and Kristen Grauman. Im2flow: Motion hallucination from static images for action recognition. In *Proc. CVPR*, 2018. 9
- [57] Jacob Walker, Abhinav Gupta, and Martial Hebert. Dense optical flow prediction from a static image. In *Proc. ICCV*, 2015. 9
- [58] Silvia L Pintea, Jan C van Gemert, and Arnold WM Smeulders. Déjà vu: Motion prediction in static images. In *Proc. ECCV*, 2014. 9
- [59] Jacob Walker, Carl Doersch, Abhinav Gupta, and Martial Hebert. An uncertain future: Forecasting from static images using variational autoencoders. In *Proc. ECCV*. Springer, 2016. 9
- [60] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. Flow-grounded spatial-temporal video prediction from still images. In *Proc. ICCV*, 2018. 9
- [61] Jacob Walker, Kenneth Marino, Abhinav Gupta, and Martial Hebert. The pose knows: Video forecasting by generating pose futures. In *Proc. ICCV*, 2017. 9
- [62] Angjoo Kanazawa, Shubham Tulsiani, Alexei A Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *Proc. ECCV*, 2018. 9
- [63] Subhabrata Choudhury, Laurynas Karazija, Iro Laina, Andrea Vedaldi, and Christian Rupprecht. Guess what moves: unsupervised video and image segmentation by anticipating motion. *arXiv preprint arXiv:2205.07844*, 2022. 9
- [64] Simon Niklaus and Feng Liu. Softmax splatting for video frame interpolation. In *Proc. CVPR*, 2020. 9
- [65] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A. Efros. Generative visual manipulation on the natural image manifold. In *Proc. ECCV*, 2016. 9
- [66] Erik Härkönen, Aaron Hertzmann, Jaakko Lehtinen, and Sylvain Paris. Ganspace: Discovering interpretable gan controls. *Proc. NeurIPS*, 2020. 9

- [67] Ayush Tewari, Mohamed Elgharib, Gaurav Bharaj, Florian Bernard, Hans-Peter Seidel, Patrick Pérez, Michael Zollhofer, and Christian Theobalt. Stylerig: Rigging stylegan for 3d control over portrait images. In *Proc. CVPR*, 2020. 9
- [68] Yujun Shen, Ceyuan Yang, Xiaoou Tang, and Bolei Zhou. Interfacegan: Interpreting the disentangled face representation learned by gans. *IEEE transactions on pattern analysis and machine intelligence*, 44(4):2004–2018, 2020. 9
- [69] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan: How to embed images into the stylegan latent space? In *Proc. ICCV*, 2019. 9
- [70] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan++: How to edit the embedded images? In *Proc. CVPR*, 2020. 9
- [71] David Bau, Hendrik Strobelt, William Peebles, Jonas Wulff, Bolei Zhou, Jun-Yan Zhu, and Antonio Torralba. Semantic photo manipulation with a generative image prior. *arXiv preprint arXiv:2005.07727*, 2020. 9
- [72] Yuval Alaluf, Or Patashnik, and Daniel Cohen-Or. Restyle: A residual-based stylegan encoder via iterative refinement. In *Proc. ICCV*, 2021. 9
- [73] Shanyan Guan, Ying Tai, Bingbing Ni, Feida Zhu, Feiyue Huang, and Xiaokang Yang. Collaborative learning for faster stylegan embedding. *arXiv preprint arXiv:2007.01758*, 2020. 9
- [74] Stanislav Pidhorskyi, Donald A Adjeroh, and Gianfranco Doretto. Adversarial latent autoencoders. In *Proc. CVPR*, 2020. 9
- [75] Elad Richardson, Yuval Alaluf, Or Patashnik, Yotam Nitzan, Yaniv Azar, Stav Shapiro, and Daniel Cohen-Or. Encoding in style: a stylegan encoder for image-to-image translation. In *Proc. CVPR*, 2021. 9
- [76] Omer Tov, Yuval Alaluf, Yotam Nitzan, Or Patashnik, and Daniel Cohen-Or. Designing an encoder for stylegan image manipulation. *ACM Transactions on Graphics (TOG)*, 40(4):1–14, 2021. 9
- [77] Tengfei Wang, Yong Zhang, Yanbo Fan, Jue Wang, and Qifeng Chen. High-fidelity gan inversion for image attribute editing. In *Proc. CVPR*, 2022. 9
- [78] Ayush Tewari, Mohamed Elgharib, Florian Bernard, Hans-Peter Seidel, Patrick Pérez, Michael Zollhöfer, and Christian Theobalt. Pie: Portrait image embedding for semantic control. *ACM Transactions on Graphics (TOG)*, 39(6):1–14, 2020. 9
- [79] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proc. CVPR*, 2019. 9

---

# Diffusion with Forward Models: Solving Stochastic Inverse Problems Without Direct Supervision

---

Ayush Tewari<sup>1\*</sup> Tianwei Yin<sup>1\*</sup> George Cazenavette<sup>1</sup> Semen Rezchikov<sup>4</sup>  
Joshua B. Tenenbaum<sup>1,2,3</sup> Frédéric Durand<sup>1</sup> William T. Freeman<sup>1</sup> Vincent Sitzmann<sup>1</sup>

<sup>1</sup>MIT CSAIL <sup>2</sup>MIT BCS <sup>3</sup>MIT CBMM <sup>4</sup>Princeton IAS

## Contents

### 1 Proposition

### 2 Details of the Method

2.1	Inverse Graphics	.....
2.2	Single-Image Motion Prediction	.....
2.3	GAN Inversion	.....
2.4	Sampling	.....

### 3 Limitations

## 1 Proposition

**Proposition 1.** Suppose that any signal  $\mathbf{S}$  can be reconstructed from the set of all *all possible* observations of  $\mathbf{S}$ . Under this assumption, if in the limit as the number of known observations per signal goes to infinity, there are parameters  $\theta$  such that  $\mathcal{L}_\theta^{\text{trgt}} + \mathcal{L}^{\text{novel}}$  is minimized, then the conditional probability distribution over signals discovered by our model  $p(\mathbf{S} | \mathbf{O}^{\text{ctxt}}; \phi^{\text{ctxt}})$  agrees with the true distribution  $p^{\text{true}}(\mathbf{S} | \mathbf{O}^{\text{ctxt}}; \phi^{\text{ctxt}})$ .

The total observation loss is defined in Equation equation 4 below.

After introducing some notation, we will formalize the assumptions made in the proposition.

**Definition 1.** We call the collection of all observations that correspond to a signal a *total observation* of the signal  $\mathbf{O}^{\text{total}}$ . Formally,

$$\mathbf{O}^{\text{total}} = \mathbf{O}^{\text{total}}(\mathbf{S}) = \{(\phi, \text{forward}(\mathbf{S}, \phi))\}_{\phi \in \mathcal{P}}.$$

Here,  $\mathcal{P}$  denotes the set of *parameters* of the forward model, e.g.  $\mathcal{P} = SE(3)$  for the inverse graphics application in the paper.

**Definition 2.** We define the *scattering map* as the (measurable) map sending signal  $\mathbf{S}$  to its total image  $\mathbf{O}^{\text{total}}$ :

$$\text{Scatter} : \mathbf{S} \mapsto \mathbf{O}^{\text{total}}(\mathbf{S}).$$

For a reference for the technical notion of a measurable map, see any textbook on measure theory (e.g. [1]); all maps arising in machine-learning models are measurable because they are piecewise continuous.

---

\* Equal Contribution.

**Assumption** We formalize the assumption of Proposition 1 by requiring that there is a (measurable) map  $\text{Scatter}^{-1}$  from total observations to signals which satisfies, for all signals under consideration,

$$\text{Scatter}^{-1}(\text{Scatter}(\mathbf{S})) = \mathbf{S}. \quad (1)$$

In other words, given all possible observations of a signal, we can uniquely reconstruct the signal (for the class of signals under consideration). Alternatively, the map  $\text{Scatter}$  is injective. This assumption is a basic assumption necessary for many algorithms in 3D computer vision, and underlies the recent success of differentiable rendering for 3D scene reconstruction [2] from large sets of image observations. Note that there may be total observations  $\mathbf{O}^{\text{total}} = \{(\phi, \mathbf{O}_\phi^{\text{total}})\}_{\phi \in \mathcal{P}}$  which do *not* arise as the total observations  $\mathbf{O}^{\text{total}}(S)$  of any signal  $S$ . Equation 1 makes no assumption on the behavior of  $\text{Scatter}^{-1}(\mathbf{O}^{\text{total}})$  on such ‘inconsistent’ total observations  $\mathbf{O}^{\text{total}}$ .

**Observations generated by our model are slices of total observations.** A basic property of our model is that the target observations arise from predicted signals, since

$$\mathbf{O}^{\text{trgt}} = \text{forward}(\mathbf{S}, \phi^{\text{trgt}}).$$

Thus, our model is limited to modeling the space over observations that are a member of the total observations set, i.e.,  $(\phi^{\text{trgt}}, \mathbf{O}^{\text{trgt}}) \in \mathbf{O}^{\text{total}}(S)$  for some signal  $S$ . This is an important property that is not trivially true for many existing models, e.g., for inverse graphics, many light-field-based approaches [3–6] do not satisfy this property.

**The predicted distribution over signals can be recovered from the distribution over observations.** Since we can reconstruct a signal from its total observation, we have that  $\text{Scatter}^{-1}(\text{Scatter}(U)) = U$  for any set of signals  $U$ . Writing

$$V = \{(\phi^{\text{trgt}}, \text{forward}(\mathbf{S}, \phi^{\text{trgt}})) \mid \phi^{\text{trgt}} \in \mathcal{P}, \mathbf{S} \in U\} = \{\text{Scatter}(\mathbf{S}) \mid \mathbf{S} \in U\}.$$

for the set of *total observations* of signals  $\mathbf{S} \in U$ , we therefore have that

$$p(\mathbf{O}^{\text{total}}(\mathbf{S}) \in V \mid \mathbf{O}^{\text{ctxt}}; \phi^{\text{ctxt}}) = p(\mathbf{S} \in U \mid \mathbf{O}^{\text{ctxt}}; \phi^{\text{ctxt}}). \quad (2)$$

As such, we can recover  $p(\mathbf{S} \in U \mid \mathbf{O}^{\text{ctxt}}; \phi^{\text{ctxt}})$  by computing  $p(\mathbf{O}^{\text{total}}(\mathbf{S}) \in V \mid \mathbf{O}^{\text{ctxt}}; \phi^{\text{ctxt}})$  for all possible  $V$  (where we note that if  $V$  consists of total observations that do not arise from signals then its probability is zero).

**Our loss maximizes the likelihood over total observations.** We now claim that the loss we optimize forces our model to find parameters  $\theta$  such that

$$p(\mathbf{O}^{\text{total}}(\mathbf{S}) \in V \mid \mathbf{O}^{\text{ctxt}}; \phi^{\text{ctxt}}) = p^{\text{true}}(\mathbf{O}^{\text{total}}(\mathbf{S}) \in V \mid \mathbf{O}^{\text{ctxt}}; \phi^{\text{ctxt}}) \quad (3)$$

We first define the *total observation loss*

$$\mathcal{L}_\theta^{\text{total}} = \mathbb{E}_{\mathbf{O}^{\text{ctxt}}, \mathbf{O}^{\text{trgt}}, \phi^{\text{ctxt}}, \phi^{\text{trgt}}, t} \left[ \|\mathbf{O}^{\text{trgt}} - \text{forward}(\text{denoise}_\theta(\mathbf{O}^{\text{ctxt}}, \mathbf{O}_t^{\text{total}}; t, \phi^{\text{ctxt}}), \phi^{\text{trgt}})\|^2 \right] \quad (4)$$

This is the same as  $\mathcal{L}_\theta^{\text{target}}$  of the main text, but with  $\text{denoise}_\theta$  depending on the total observation. We now have the identity

$$\begin{aligned} & \mathbb{E}_{\mathbf{O}^{\text{trgt}}, \phi^{\text{trgt}}} \left[ \|\mathbf{O}^{\text{trgt}} - \text{forward}(\text{denoise}_\theta(\mathbf{O}^{\text{ctxt}}, \mathbf{O}_t^{\text{total}}; t, \phi^{\text{ctxt}}), \phi^{\text{trgt}})\|^2 \right] \\ &= \|\mathbf{O}_t^{\text{total}} - \hat{\mathbf{O}}_{t-1}^{\text{total}}\|^2 \\ &= C_t D_{KL}(q(\mathbf{O}_{t-1}^{\text{total}} \mid \mathbf{O}_t^{\text{total}}, \mathbf{O}_0^{\text{total}}, \mathbf{O}^{\text{ctxt}}; \phi^{\text{trgt}}) \mid p_\theta(\mathbf{O}_{t-1}^{\text{total}} \mid \mathbf{O}_t^{\text{total}}, \mathbf{O}^{\text{ctxt}}; \phi^{\text{trgt}})). \end{aligned} \quad (5)$$

where  $C_t$  is some positive constant for each  $t$ ; this follows from Equations 95-99 of [7]. Thus, *if the model has parameters  $\theta$  such that Eq. 3 holds for this parameter, then this will also hold the global minimum of the loss*, since Eq. 3 holds exactly when the  $t = 1$  term of Eq. 5 is zero.

It is natural to train such a model by minimizing the loss

$$\underbrace{\|\mathbf{O}^{\text{trgt}} - \text{forward}(\text{denoise}_\theta(\mathbf{O}^{\text{ctxt}}, \mathbf{O}_t^{\text{total}}; t, \phi^{\text{ctxt}}), \phi^{\text{trgt}})\|^2}_{=\hat{\mathbf{O}}_{t-1}^{\text{trgt}}}$$

with randomly-sampled forward-model parameters  $\phi^{\text{trgt}}$ . This is what we do in our real training procedure, except that  $\text{denoise}_\theta$  now only depends on a slice of  $\mathbf{O}_t^{\text{total}}$ , namely  $\mathbf{O}_t^{\text{trgt}}$ , as well as on  $\phi^{\text{trgt}}$ , see Eq. 8 in the main paper. Now, nothing in equation 5 requires  $\hat{\mathbf{O}}_{t-1}^{\text{total}}$  to depend on all of  $\mathbf{O}_t^{\text{total}}$ ; the equation is still valid even if  $\hat{\mathbf{O}}_{t-1}^{\text{total}}$  is a function only of a slice of  $\mathbf{O}_t^{\text{total}}$ . This is precisely the case in our training procedure. As such, the addition of the term  $\mathcal{L}_{\text{novel}}$  to the loss (see Eq. 9 in the main paper), when  $\phi^{\text{novel}}$  is stochastically sampled, *forces the quantity in Equation 5 to be minimized* even though our estimate of the denoised signal only depends on the context observation and the noised target observation. Thus, the conclusion of this proposition still applies to our training procedure.

**Conclusion of proof.** We now conclude that

$$p(\mathbf{S} \in U \mid \mathbf{O}^{\text{ctxt}}; \phi^{\text{ctxt}}) = p^{\text{true}}(\mathbf{S} \in U \mid \mathbf{O}^{\text{ctxt}}; \phi^{\text{ctxt}}) \quad (6)$$

by applying (3) and using the fact that (2) holds both for  $p$  and for  $p^{\text{true}}$ . Equation 6 is the desired conclusion.

**Remark on 3D consistency.** Technically, in the models in our paper, we have that  $p(\mathbf{S} \mid \mathbf{O}^{\text{ctxt}}; \phi^{\text{ctxt}})$  is actually dependent on auxiliary parameter  $\phi^{\text{trgt}}$ : we make predictions over signals using a diffusion model coupled with a *particular choice* of forward-model parameter. As such, to be precise, we say that our model predicts a family of distributions  $p_{\phi^{\text{trgt}}}(\mathbf{S} \mid \mathbf{O}^{\text{ctxt}}; \phi^{\text{ctxt}})$  depending on  $\phi^{\text{trgt}}$ , where these distributions may differ for different values of  $\phi^{\text{trgt}}$ . Correspondingly, the model predicts a family of distributions  $p_{\phi^{\text{trgt}}}(\mathbf{O}^{\text{total}} \mid \mathbf{O}^{\text{ctxt}}; \phi^{\text{trgt}})$ . However, the addition of the  $\mathcal{L}_{\text{novel}}$  term forces learned distribution over signals to agree with the true distribution over signals in the limit of infinite observations; as such, in that same limit, the learned distribution over signals becomes independent of  $\phi^{\text{trgt}}$  since the true distribution is manifestly independent of it.

**Inverting the scatter map is unnecessary.** In the above argument, while we assumed the inverse to the *Scatter* map, we did not need to *compute* the map  $\text{Scatter}^{-1}$  to argue that the estimated probability densities  $p(\mathbf{S} \mid \mathbf{O}^{\text{ctxt}}; \phi^{\text{ctxt}})$  agree with  $p^{\text{true}}(\mathbf{S} \mid \mathbf{O}^{\text{ctxt}}; \phi^{\text{ctxt}})$ . This is a highly desirable property, as the map  $\text{Scatter}^{-1}$  often cannot be computed efficiently. Thus, our model learns correct estimates of  $p^{\text{true}}(\mathbf{S} \mid \mathbf{O}^{\text{ctxt}}; \phi^{\text{ctxt}})$  without ever explicitly computing  $\text{Scatter}^{-1}$ .

## 2 Details of the Method

### 2.1 Inverse Graphics

**Loss Function** We incorporate the use of several regularization terms:

$$\mathcal{L}_{\text{reg}} = \mathcal{L}_{\text{LPIPS}} + \mathcal{L}_{\text{depth}} + \mathcal{L}_{\text{cond}}, \quad (7)$$

$$\mathcal{L}_{\text{LPIPS}} = \mathcal{L}_{\text{LPIPS}}(\hat{\mathbf{O}}_{t-1}^{\text{trgt}}, \mathbf{O}^{\text{trgt}}), \quad (8)$$

$$\mathcal{L}_{\text{depth}} = \mathcal{L}_{\text{EAS}} + \mathcal{L}_{\text{dist}}, \quad (9)$$

$$\mathcal{L}_{\text{cond}} = \|\mathbf{O}_{\text{det}}^{\text{trgtcolor}} - \mathbf{O}^{\text{trgt}}\|^2. \quad (10)$$

Here,  $\mathcal{L}_{\text{LPIPS}}$  is the LPIPS perceptual loss [8] that encourages rendered images to be perceptually similar to the ground truth observation. This has been shown to help improve the quality of diffusion models [9]. We further regularize the depth renderings from the target and novel viewpoints using an edge-aware smoothness loss [10] and a distortion loss [11] that discourages floating geometry artifacts. These depth regularization terms encourage *natural* 3D geometry reconstructions. Finally, we use  $\mathcal{L}_{\text{cond}}$  on the rgb component of the deterministic estimate  $\mathbf{O}_{\text{det}}^{\text{trgt}}$ , denoted as  $\mathbf{O}_{\text{det}}^{\text{trgtcolor}}$ . Recall that we use  $\mathbf{O}_{\text{det}}^{\text{trgt}}$  to condition our denoising network, and that it includes color as well as high-dimensional features. We use multiplier hyperparameters 0.2 for  $\mathcal{L}_{\text{LPIPS}}$  and 0.02 for  $\mathcal{L}_{\text{depth}}$ . Our code will be publicly released to aid in reproducibility.

**Remark on regularization** Recall our assumption in Proposition 1 that the map from all observations to the signal is invertible. In the 3D setting, where we use real-world 2D datasets for training, we do not have access to all possible signal observations. On such training data, this assumption is not strictly true, since multiple 3D scenes can explain a subset of observations. However, the addition of the regularizing terms singles out *preferred* choices of 3D scenes explaining the known observations.

Heuristically, our depth and smoothness regularizers  $\mathcal{L}_{\text{depth}}$  make the map between scenes and the observations in the training dataset invertible, i.e., they single out a *uniquely determined natural* 3D scene that explains the observations in the impoverished 2D dataset.

**Training Details** Volume rendering is an expensive computation, making training our 3D models under limited memory budgets challenging. However, unlike image-space diffusion models, where the entire image is predicted directly, we can render pixels independent of each other using `render`. In practice, we render  $24 \times 24$  patches at random positions in the image in each training iteration. We use the vision transformer architecture from DiT [12] to implement the image backbone `enc` in `denoise`. We modify the MLP architecture (MLP) of pixelNeRF to support additional time conditioning input in our models. Our models are trained on 8 A100 GPUs, with a batch size of 24. Training takes around 7 days for RealEstate10k, and around 3 days for Co3D. We use ADAM with a learning rate of  $2e-5$ . We initially train at a resolution of  $64 \times 64$ . We then finetune the model at  $128 \times 128$ . For our Co3D models, we found it helpful to first pretrain the deterministic conditioning component of the model for 10k iterations. We use 64 samples each for coarse and fine stages for volume rendering of the output 3D reconstruction, and only 32 coarse samples for rendering the conditioning input.

We process the Co3D dataset following [5], i.e., we center-crop the images and resize them to a consistent resolution. We follow Chan et al. [5] to provide the absolute pose of the input image as an additional input to the encoder. We also use this input for our baselines, except when using official codebases of SparseFusion [13] and pixelNeRF [14]. During training, we randomly select the initial context frame, and pick a target frame for denoising within predetermined distance intervals. We randomly choose one additional frame between initial context frame and the target frame for computing the novel view reconstruction loss ( $\mathcal{L}_{\text{novel}}$ ). To support autoregressive sampling, we add more context frames from the dataset during training, such that the network can reason jointly from multiple input images. At test time, we iteratively sample new images that are then added as a context frame for the next frame. Autoregressive sampling allows us to cover the entire 360 regions in Co3D scenes by diffusing multiple images around the object. We follow the same training strategy for RealEstate10k, except that we do not feed in absolute poses to our encoder or the baselines. We augment the RealEstate10k dataset by randomly reversing the order of frames in the videos.

**Baselines** We use code provided by the authors for SparseFusion and train on our datasets. We note that the SparseFusion paper only demonstrated results on segmented-out objects without any background, and used multiple input images at test time, unlike our monocular method. Since SparseFusion uses a pretrained VAE backbone that only takes inputs at  $256 \times 256$  resolution, we train it at this resolution. However, the 3D optimization is performed at the same resolution as our method. We use the official repository (<https://github.com/sxyu/pixel-nerf>) for the pixelNeRF baseline. We use 50 scenes for Co3D and 100 scenes for RealEstate for the quantitative evaluations.

## 2.2 Single-Image Motion Prediction

We use an edge-aware smoothness regularization loss on the motion field that is equivalent to the smoothness loss on the depths defined in Sec. 2.1. We use the DiT architecture as our denoising model. The clean context image and the noisy target image are concatenated along the channel dimension and used as input to the network. The output is pixel-aligned motion field that is used to warp the context into the target using SoftSplat [15]. We use ADAM with a learning rate of  $2e-5$  and batch size of 72 to optimize our networks on 2 RTXA6000 GPUs. Our models are trained on the Vimeo90K dataset [16].

## 2.3 GAN Inversion

The GAN into which we are inverting is a StyleGAN2-Ada [17] trained on the  $256 \times 256$  FFHQ dataset [18]. Our “ground truth” target dataset is generated by taking random samples from this GAN with a truncation  $\psi$  of 0.5, down-sampled to  $64 \times 64$  pixels. We, again, use the DiT architecture as our denoising model. The context images are obtained by taking a ground truth sample and masking out all but a small patch of varying size. The masked context image and noise target image are concatenated along the channel dimension and used as input to the denoising network, which predicts the denoised “w” code. This w is then fed through the forward model (generator) and downsampled to

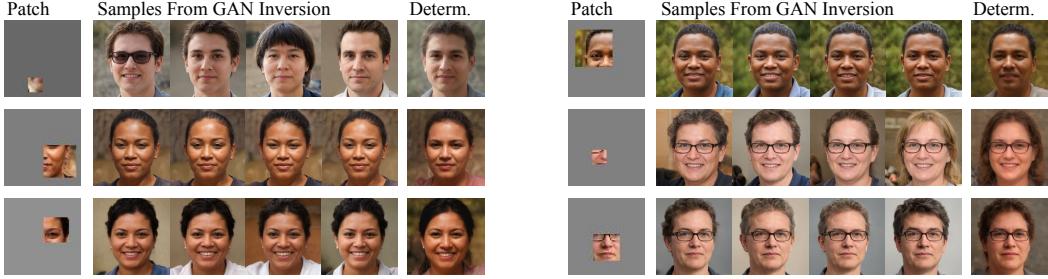


Figure 1: More samples from our GAN Inversion model. Our method produces many plausible faces given only a small patch.

$64 \times 64$  pixels to obtain our denoised target image. All of our training is done at  $64 \times 64$  resolution, but using a GAN trained on high-resolution  $256 \times 256$  images allows us to obtain high-resolution results at test time by simply not downsampling the final denoised output. We use the ADAM optimizer with a learning rate of  $2e-5$  and a batch size of 4 to train our networks on a single RTXA6000 GPU. We include more results in Figure 1.

#### 2.4 Sampling

We use 50 DDIM [19] denoising timesteps for all our results across all applications. All our models, except the inverse graphics model trained on RealEstate10k, are trained without any classifier-free guidance. For our RealEstate10k model, we use a classifier-free guidance weight of 2. Here, the model is also trained as an unconditional model, where the conditioning image is zeroed out for 10% of the iterations.

### 3 Limitations

While we present the first method that enables diffusion models to learn the conditional distribution over signals, only using observations through a forward model, our approach has several limitations. Our sampling times can be very expensive in some cases. The sampling time ranges from just a few seconds for our GAN application, to around 100 mins for 360-degree autoregressive sampling for Co3D. This is both due to the expensive nature of the iterative denoising process, as well as the cost of rendering 3D reconstructions using volume rendering. Our training has large memory requirements, and can thus not be trained on smaller GPUs. Future work on making these models easier to train would make them more applicable. Our models are not trained on very large-scale datasets, and can thus not generalize to out-of-distribution data. Finally, we only present preliminary investigations into applications outside inverse graphics; however, we hope that we offer a strong experimental base that can be beneficial for future exploration.

## References

- [1] Elias M Stein and Rami Shakarchi. *Real analysis*. Princeton lectures in analysis. Princeton University Press, Princeton, NJ, March 2005.
- [2] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *Proc. ECCV*, 2020.
- [3] Vincent Sitzmann, Semon Rezchikov, Bill Freeman, Josh Tenenbaum, and Fredo Durand. Light field networks: Neural scene representations with single-evaluation rendering. In *NeurIPS*, 2021.
- [4] Mehdi SM Sajjadi, Henning Meyer, Etienne Pot, Urs Bergmann, Klaus Greff, Noha Radwan, Suhani Vora, Mario Lučić, Daniel Duckworth, Alexey Dosovitskiy, et al. Scene representation transformer: Geometry-free novel view synthesis through set-latent scene representations. In *CVPR*, 2022.
- [5] Eric R Chan, Koki Nagano, Matthew A Chan, Alexander W Bergman, Jeong Joon Park, Axel Levy, Miika Aittala, Shalini De Mello, Tero Karras, and Gordon Wetzstein. Generative novel view synthesis with 3d-aware diffusion models. *arXiv preprint arXiv:2304.02602*, 2023.
- [6] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3d object, 2023.
- [7] Calvin Luo. Understanding diffusion models: A unified perspective. *arXiv preprint arXiv:2208.11970*, 2022.
- [8] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proc. CVPR*, 2018.
- [9] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. *arXiv preprint arXiv:2303.01469*, 2023.
- [10] Sylvain Paris, Pierre Kornprobst, Jack Tumblin, Frédo Durand, et al. Bilateral filtering: Theory and applications. *Foundations and Trends® in Computer Graphics and Vision*, 4(1):1–73, 2009.
- [11] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5470–5479, 2022.
- [12] William Peebles and Saining Xie. Scalable diffusion models with transformers. *arXiv preprint arXiv:2212.09748*, 2022.
- [13] Zhizhuo Zhou and Shubham Tulsiani. Sparsefusion: Distilling view-conditioned diffusion for 3d reconstruction. *Proc. CVPR*, 2023.
- [14] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *Proc. CVPR*, 2021.
- [15] Simon Niklaus and Feng Liu. Softmax splatting for video frame interpolation. In *Proc. CVPR*, 2020.
- [16] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T Freeman. Video enhancement with task-oriented flow. *International Journal of Computer Vision*, 127:1106–1125, 2019.
- [17] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. *Advances in neural information processing systems*, 33:12104–12114, 2020.
- [18] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proc. CVPR*, 2019.
- [19] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *ICLR*, 2021.