SYNTHESIZING NOVEL VIEWS WITH DIFFUSION MODELS

An Undergraduate Research Scholars Thesis

by

BRANDON G. NGUYEN

Submitted to the LAUNCH: Undergraduate Research office at
Texas A&M University
in partial fulfillment of the requirements for the designation as an

UNDERGRADUATE RESEARCH SCHOLAR

Approved by
Faculty Research Advisor: Dr. Nima Kalantari

May 2023

Major: Computer Science

RESEARCH COMPLIANCE CERTIFICATION

Research activities involving the use of human subjects, vertebrate animals, and/or biohazards must be reviewed and approved by the appropriate Texas A&M University regulatory research committee (i.e., IRB, IACUC, IBC) before the activity can commence. This requirement applies to activities conducted at Texas A&M and to activities conducted at non-Texas A&M facilities or institutions. In both cases, students are responsible for working with the relevant Texas A&M research compliance program to ensure and document that all Texas A&M compliance obligations are met before the study begins.

I, Brandon G. Nguyen, certify that all research compliance requirements related to this Undergraduate Research Scholars thesis have been addressed with my Faculty Research Advisor prior to the collection of any data used in this final thesis submission.

This project did not require approval from the Texas A&M University Research Compliance & Biosafety office.

TABLE OF CONTENTS

ABSTRACT

Synthesizing Novel Views with Diffusion Models

Brandon G. Nguyen
Department of Computer Science and Engineering
Texas A&M University

Faculty Research Advisor: Dr. Nima Kalantari
Department of Computer Science and Engineering
Texas A&M University

Diffusion models have become the state-of-the-art generative model in a multitude of generative tasks such as audio and image synthesis. Until recently, there has not been much success with the specific image-to-image task of novel view synthesis; where a model is given a reference frame of a scene and is then queried about what the scene may look like from another, different, view. One of these recent developments is the 3DiM model, where a conditional diffusion model is modified with cross attention modules in order to leverage features across views in order to synthesize more consistent and higher quality novel views. In this work, we re-implement this 3DiM model with PyTorch to gauge its performance and to analyze its capabilities in synthesizing views with varying parameters and spatial resolutions.

ACKNOWLEDGMENTS

Contributors

Thanks to Dr. Nima Kalantari and his students Avinash Paliwal and Pedro Figueirêdo for their guidance and support throughout the course of this research.

All other work conducted for the thesis was completed by the student independently.

Funding Sources

Portions of this research were conducted with the advanced computing resources provided by Texas A&M High Performance Research Computing.

NOMENCLATURE

VAE	Variational Auto-Encoder
ELBO	Evidence Lower Bound Objective
DDPM	Denoising Diffusion Probabilistic Models
NeRF	Neural Radiance Fields
LFN	Light Field Networks
FID	Fréchet Inception Distance
PE	Sinusoidal Positional Embedding
FiLM	Feature-wise Linear Modulation
3DiM	3D Diffusion Model

1. INTRODUCTION

Despite the proven success of diffusion models in a wide variety of image-to-image tasks, there have been limited results with novel view synthesis until recently [1]. The objective of novel view synthesis is to create a new view of a scene using information from existing different views of the same scene.

Previous work in this area are limited in aspects resolution, quality, scene representation, or multi-view consistency. 3D GANs can offer faster high quality sampling like with EG3D [2], however, this is often at the expense of consistency. NeRF [3] are able to generate consistent views, but are highly compute-intensive and do not generalize well to multiple scenes. And other methods like MPI [4] are able to work on arbitrary scenes, but struggle with view-dependent effects like reflections.

With diffusion [1], it is possible to obtain a model that is able to generate high quality samples that is also generalizable to many different scenes. A few of the outstanding issues with diffusion models is with synthesizing 3D consistent views, model size, and sampling speed. In [1] they attend to the consistency issue, and in more recent work like [5] there has been attempts to speed up diffusion sampling without sacrificing sample quality.

In this work, we aimed at reimplementing the 3DiM method [1] using PyTorch and to evaluate the performance to see areas that can be extended in future work.

All codes used in this work has been published to GitHub as an open-source repository at `https://github.com/barnden/novel-view-diffusion`. The data-sets [6] [7] [8] have been either released to the public or can be obtained for research purposes.

2. BACKGROUND

2.1 Variational Auto-Encoders

The Variational Auto-Encoder (VAE) [9] is a generative model based in Bayesian networks. As is usual in classification or regressive tasks learning a conditional model $p_\theta(\mathbf{z}|\mathbf{x})$ is more useful than the unconditional $p_\theta(\mathbf{x})$. That is, VAEs aim to answer the question of what is $\mathbf{z}$ if we have made the observation of $\mathbf{x}$? In the context of a classifier, a more concrete question that may arise is that: if we are given an image $\mathbf{x}$, then what is a likely label $\mathbf{z}$ which corresponds to the image?

In a regressive context, as is pertinent to this work, it may be that we have a directed probabilistic process and we want to learn the probability of an event $\mathbf{z}$ conditioned against multiple observations $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$. Or, suppose that the conditional distribution $p(\mathbf{x}|\mathbf{z})$ is known and we wanted to reverse it to get $p(\mathbf{z}|\mathbf{x})$.

Unfortunately, computing the reverse probability $p(\mathbf{z}|\mathbf{x})$ is an intractable problem in most situations. To see why, we use the definition of conditional probability to see that

$$p(\mathbf{z}|\mathbf{x}) = \frac{p(\mathbf{x}|\mathbf{z})}{p(\mathbf{x})} \tag{1}$$

where $p(\mathbf{x}|\mathbf{z})$ is known due to the construction of the problem and the probability

$$p(\mathbf{x}) = \int p(\mathbf{x}, \mathbf{z}) \, \mathrm{d}\mathbf{z}. \tag{2}$$

In the context of deep learning networks, it is almost guaranteed that computing this integral for $p(\mathbf{x})$ is an intractable problem. The VAE utilizes a reparameterization trick to estimate a variational lower bound to an otherwise intractable posterior distribution of a directed probabilistic model.

To explain the problem, suppose that there is some latent variable $\mathbf{z}_i$ is generated from a prior distribution $p_\theta(\mathbf{z}_i)$, an $\mathbf{x}_i \sim p_\theta(\mathbf{x}|\mathbf{z})$, and the true parameters θ and values of $\mathbf{z}_i$ are unknown. Then, it is often the case that the marginal likelihood $p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{z}) p_\theta(\mathbf{x}|\mathbf{z}) \, \mathrm{d}\mathbf{z}$ and the true poste-

rior $p_\theta(\mathbf{z}|\mathbf{x}) = p_\theta(\mathbf{z}|\mathbf{x})p_\theta(\mathbf{z}/p_\theta(\mathbf{x}))$ are both intractable. The stochastic gradient variational Bayes (SGVB) estimator [9] gives an efficient way to infer an approximation $q_\phi(\mathbf{z}|\mathbf{x})$ of the posterior to the latent $\mathbf{z}$ with observation $\mathbf{x}$.

Without going into much detail, a random variable $\mathbf{z}^{(i)}$ sampled from the approximate posterior $q_\phi(\mathbf{z}|\mathbf{x})$ can be reparameterized using a transformation $g_\phi\left(\boldsymbol{\epsilon}^{(i)}, \mathbf{x}^{(i)}\right)$ with $\boldsymbol{\epsilon}^{(i)}$ being sampled from some probability distribution $p(\boldsymbol{\epsilon}^{(i)})$ such that $\mathbf{z}^i = g_\phi\left(\boldsymbol{\epsilon}^{(i)}, \mathbf{x}^{(i)}\right)$. Then, $q_\phi(\mathbf{z}|\mathbf{x})\prod_i \mathrm{d}z_i = p(\boldsymbol{\epsilon})\prod_i \mathrm{d}\epsilon_i$. Therefore, $\int q_\phi(\mathbf{z}|\mathbf{x})f(\mathbf{z})\,\mathrm{d}\mathbf{z} = \int p(\boldsymbol{\epsilon})f(g_\phi(\boldsymbol{\epsilon}, \mathbf{x}))\,\mathrm{d}\boldsymbol{\epsilon}$. It is from this fact that the SGVB is derived from [9].

In a VAE, the prior is the multivariate Gaussian $p_\theta(\mathbf{z}) = \mathcal{N}(\mathbf{z}; 0, \boldsymbol{I})$ and $p_\theta(\mathbf{x}|\mathbf{z})$ is also a multivariate Gaussian with distribution parameters computed from $\mathbf{z}$ with a MLP and, as a result, the true posterior $p_\theta(\mathbf{z}|\mathbf{x})$ is intractable. Then utilizing the trick, the variational approximation to the posterior is also a multivariate Gaussian with respect to an observation $\mathbf{x}^{(i)}$

$$\log q_\theta\left(\mathbf{z}|\mathbf{x}^{(i)}\right) = \log \mathcal{N}\left(\mathbf{z}; \boldsymbol{\mu}^{(i)}, \boldsymbol{\sigma}^{2(i)}\boldsymbol{I}\right) \tag{3}$$

where the mean and variance of the approximate posterior are outputs of the encoder. And the loss

$$\mathcal{L} \simeq \mathbb{E}_{q_\phi}\left[\log p_\theta\left(\mathbf{x}^{(i)}|\mathbf{z}^{(i,l)}\right)\right] \tag{4}$$

where $\mathbf{z}^{(i,l)} = \boldsymbol{\mu}^{(i)} + \boldsymbol{\sigma}^{(i)}\boldsymbol{\epsilon}$ and $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$.

2.2 Diffusion Models

Diffusion models have proven to be successful in a multitude of generative tasks, and particularly in image-to-image applications that include image synthesis [10], super-resolution [11], coloration, in-painting [12], and even video synthesis [13].

The diffusion process as outlined in [10] has two driving forces: the forward and the reverse process. In each step of the forward process, small amounts of noise are gradually added onto a clean image. As we tend towards infinity the result increasingly becomes indistinguishable from pure noise. Then, in the reverse process we learn the noise that was added at each step in the

forward process. This reverse process will take a completely noised image or pure noise and incrementally denoise it. At the end of the reverse process, a new clean image is formed that is coherent to the images the model was trained on.

Compared to other methods like VAEs or flow models, diffusion models offer significantly more mode coverage (diversity in generated images), and sample quality (quality in generated images). However, this is with a trade-off with sampling speed; diffusion often requires a relatively large model, and involves removing noise in many steps. There has been work with *stable diffusion* [14] to reduce the size of diffusion models, and *distillation* [5] to reduce the number of denoising steps. Unfortunately, both of these methods involve sacrificing visual fidelity or sample quality according to quantitative metrics. But as emphasized in [14], it is only the perceptual uniformity that matters in most cases and not the metrics.

2.2.1 Forward Process

From [10], the forward process is defined as a Markov chain that has T finite steps. This process begins with a clean frame $\mathbf{x}_0$, at each step t a new frame $\mathbf{x}_t$ is created by sampling a Gaussian distribution $q(\mathbf{x}_t|\mathbf{x}_{t-1})$ which is conditioned on the observation of the past frame $\mathbf{x}_{t-1}$. For large enough T, $\mathbf{x}_T$ is nearly indistinguishable from pure Gaussian noise. In practice, $\mathbf{x}_T$ is sampled from the standard Gaussian $\mathcal{N}(0, \boldsymbol{I})$.

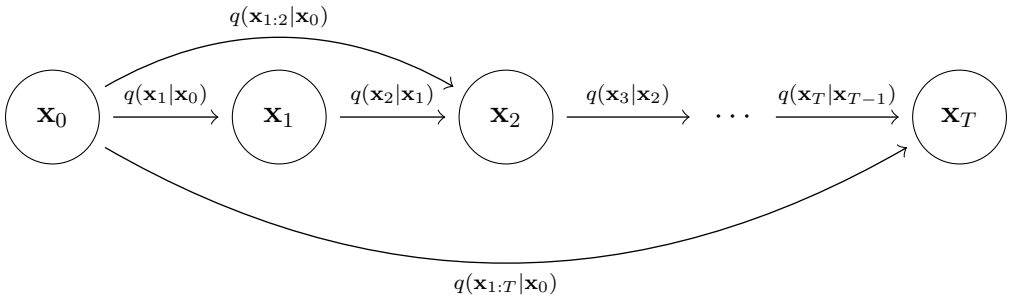

Figure 1: Illustration of the forward process.

In the forward process, sampling an $\mathbf{x}_t$ given an $\mathbf{x}_0$ can be described as sampling from a

conditional isotropic multivariate Gaussian distribution

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{1-\beta_t}\,\mathbf{x}_{t-1}, \beta_t \boldsymbol{I}\right), \tag{5}$$

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}) \tag{6}$$

where β_t is the variance at time t. These β_t are defined as hyperparameters that increase according to a linear schedule such that $0 < \beta_1 < \cdots < \beta_T < 1$.

The key insight is that with large T, each individual step has a very small variance β_t. As each step has a smaller variance, it gives a greater amount of certainty to the reverse posterior distribution of the forward step $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$. As a result, this allows us to assume that the learned prior distributions p_θ in the reverse process are also modelled by unimodal multivariate Gaussians. Later, this fact will allow us to reuse the reparameterization trick from [9] to find a tractable lower bound to the reverse posterior.

The Markovian forward process increasingly adds Gaussian noise at each step with increasing variance. A simple observation that can be made is to notice that the sum of Gaussians is a Gaussian in and of itself. By leveraging this property, it is possible to sample the entire forward process at an arbitrary timestep t with one step only. This thereby makes it possible to sample $\mathbf{x}_t$ knowing only the clean frame $\mathbf{x}_0$

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0, (1-\bar{\alpha}_t)\boldsymbol{I}\right), \tag{7}$$

where $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$.

2.2.2 Reverse Process

The prior distribution of the reverse process can be described with the joint probability $p_\theta(\mathbf{x}_{0:T})$ which is also a result of a from a Markov chain that starts with $p(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; 0, \boldsymbol{I})$. Simply, the reverse process is iteratively *removing* the noise that was added in the forward process in order to attain a new denoised clean frame $\mathbf{x}'_0$.

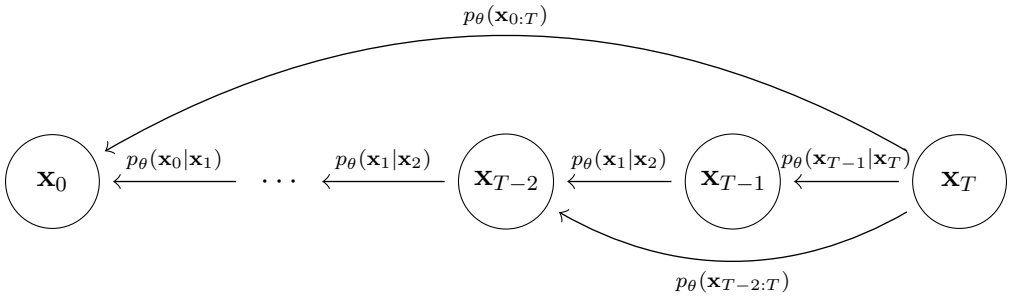

Figure 2: Illustration of the reverse process.

From [10], the reverse prior distributions are given by

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}\left(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)\right) \tag{8}$$

$$p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta\left(\mathbf{x}_{t-1}|\mathbf{x}_t\right), \tag{9}$$

where $\boldsymbol{\Sigma}_\theta$ is fixed according to a linear schedule, and $\boldsymbol{\mu}_\theta$ is the learning objective of the model. By using the reparameterization trick from [9], $\mathbf{x}_t$ can be rewritten as

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\,\boldsymbol{\epsilon} \tag{10}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$. This reparameterization allows the model to learn the noise $\boldsymbol{\epsilon}$ added rather than predicting the distribution; experimentally this yielded in better sample quality [10].

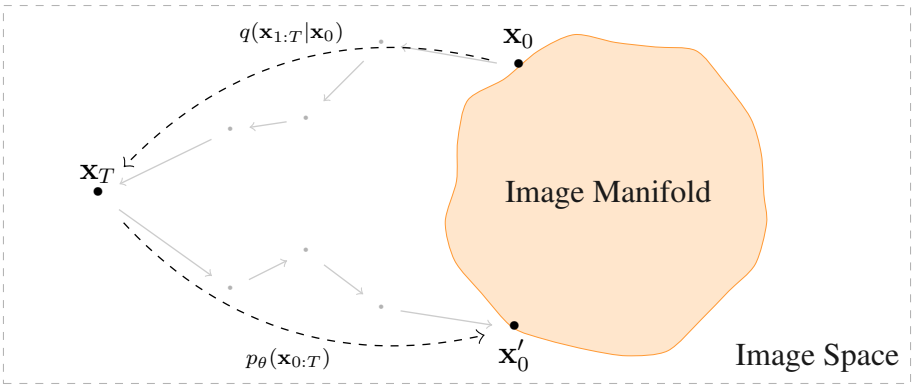

Figure 3: Illustration of random walks from the source image $\mathbf{x}_0$ and to the generated image $\mathbf{x}'_0$ on the image manifold in the image space corresponding to the forward and reverse processes.

As solving $p_\theta(\mathbf{x}_0) = \int p_\theta(\mathbf{x}_{0:T}\, \mathrm{d}\mathbf{x}_{1:T})$ is an intractable problem, we can instead optimize for the variational lower bound. Instead of using the true evidence lower bound, the usual reparameterization from (10) is used to get a simplified variational lower bound objective as given by [10]

$$\mathcal{L} = \mathbb{E}_{t,\mathbf{x}_0,\boldsymbol{\epsilon}} \left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta \left(\mathbf{x}_t, t \right)\|^2 \right], \tag{11}$$

where $t \sim \mathrm{Unif}\{1, \ldots, T\}$ and $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$.

The time t is passed into the model where it is embedded into the data using sinusoidal positional encodings (PEs) [15] given by

$$\mathrm{PE}_{t,2i} = \sin\left(\left(\frac{t}{10000} \right)^{2i/d} \right) \tag{12}$$

$$\mathrm{PE}_{t,2i+1} = \cos\left(\left(\frac{t}{10000} \right)^{2i/d} \right). \tag{13}$$

2.3 Pinhole Camera

Before discussing novel view synthesis, it is important to understand the image formation process in cameras. In essence, the light in a 3D scene will bounce off of every point in the scene, randomly in all directions. If we were to just take an image sensor and attempt to capture the light in the scene, the resulting image will appear blurry. This is a by-product of the light scattering in all possible directions. In a scene, there are many points which are able to reflect light towards the same spot on the image sensor. As a result, this is a many-to-one mapping from points in the scene to the image sensor.

Now, suppose we took a piece of paper and poked a very small hole through it. Looking through this pinhole, we are still able to see the scene. However, a lot of the light from the scene is getting blocked off. This means if we treat the pinhole as a singular point in space, then that means that there is exactly one way to reach this point from anywhere else in the scene. That is, for each point on the image sensor, only light from one direction will reach it rather than all possible angles. This gives us a one-to-one mapping of points in the scene to points on the image sensor, and we call the pinhole the center of projection, or the center of the camera.

With the pinhole camera, it is possible to trace the direction of the light rays that are hitting each point on the image sensor. It should be noted that the image sensor is also referred to as the image plane. If the image plane is placed in front of the center of projection, then the image formed on the plane will be inverted. As a result, the image plane is typically placed behind the center of projection in order to have an upright image which also has the benefit of simplifying the calculations.

The direction of a ray hitting the sensor can be recovered using the *intrinsic matrix* $K \in \mathbb{R}^{3\times3}$. Specifically, the intrinsic matrix can encodes information about the camera's focal length, skewing factor, offset, and aspect ratio. The direction of each ray from a camera with intrinsic matrix K can be given with

$$\mathbf{d}_{\text{camera}} = K^{-1}\mathbf{x} \tag{14}$$

where $\mathbf{x}$ is a point on the image plane, typically this point is chosen to be the center of a pixel on an image sensor.

However, the current direction vector we have is relative to the center of projection, or expressed in the camera space. This direction vector tells us nothing about its actual direction with respect to the world. To get the direction in the world, we require the *extrinsic matrix* often denoted as $[R|t] \in \mathbb{R}^{3\times4}$, this is also often referred to as the camera's pose. The extrinsic matrix describes the camera's rotation and translation relative to the world origin. By rotating our direction vector in camera space, we can get the direction in the world space

$$\mathbf{d} = R\,\mathbf{d}_{\text{camera}} \tag{15}$$

And the origin of the direction vector can be set to the center of projection, which in world coordinates is equal to the t vector.

Often in novel view synthesis, the key concept is how two different views relate to one another – or the relative pose. This is useful in instances where we aim to synthesize a novel view utilizing a reference view with no known absolute pose, as is the case in most real-life photos. To relativize the absolute camera rays from one camera pose to another, then the ray origins need to be transformed by the inverse extrinsic and the directions by the inverse rotation.

2.4 Novel View Synthesis

In novel view synthesis the objective is to infer how a given 3D scene will look like from a novel viewpoint with limited information about the scene. Some recent work into this area includes neural radiance fields (NeRF) [3] that are so-called "geometry-aware" models in that they synthesize views via volume rendering of an underlying 3D representation of a scene. More specifically, NeRF works by training a new model for a specific scene, then querying the model with novel views.

This class of geometry-aware models has the added benefit of guaranteeing a property called "3D consistency", i.e. each synthesized view of a scene will be consistent with one another.

To better understand this concept, think of a scene containing a ball. With no other references, it is difficult to judge the scale of the ball, i.e. it can be a small ball near the camera, or a large ball in the distance.

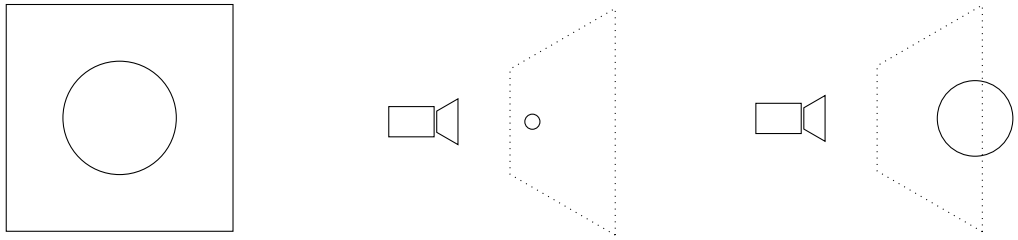

Figure 4: A visualization of the ambiguity arising from a single shot of a scene.

Another example from [1] is that there are many possible faces that can be associated with a given view of the back of a head. With 3D consistency, these properties remain the same across all views.

One issue with NeRF models in the few shot setting is that they must extrapolate information about occluded regions of a scene. This extrapolation often manifests as undesirable artifacts in synthesized views. As NeRF models are trained for a specific scene, they cannot leverage knowledge about other scenes to better extrapolate the information. However, there has been some work into optimization at test time to adapt off the shelf NeRF models to novel scenes in CodeNeRF [16].

An alternative is to use a "geometry-free" approach for novel view synthesis such as Light Field Networks (LFN) [17] and 3D Diffusion Models (3DiM) [1]. These models are focused on novel view synthesis in a "few-shot" setting, i.e. generating novel views with only a few ($\lesssim 10$) reference views.

Computing metrics such as the Fréchet Inception Distance (FID) on the geometry-free model outputs against the training dataset is incapable of scoring 3D consistency. As metrics like

FID measure the disparity between datasets. To address this [1] propose a new method called 3D consistency scoring. Their objective was to penalize inconsistent outputs but not outputs that are consistent but have deviated from the ground truth. Their solution was to generate dozens of views using the geometry-free models, then train a NeRF-like geometry-aware model. This NeRF-like model is then queried with novel poses to generate views, these views are then compared using the usual metrics against the ground truth.

2.5 X-UNet

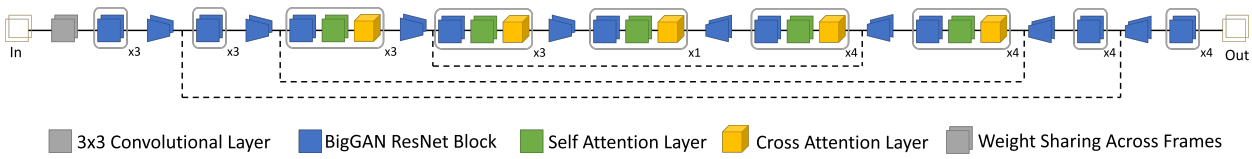

Figure 5: Diagram of the X-UNet architecture.

Most diffusion models like [10] [11] [12] [13] [1] use a UNet backbone [18] with ResNet blocks [19]. As well as some form of self-attention [15]. There has been prior work in conditional diffusion models like [11] [12], however, when applied to novel view synthesis, they were unable to synthesize views that were 3D consistent with one another. With [1], they introduce a new architecture called X-UNet based off of the conditional model in [13] which was originally designed for video synthesis.

Each residual block in X-UNet takes the feature maps and the positional embeddings as inputs like [10]. Unlike [10], however, the input frame $\mathbf{z}_k^{(\lambda_t)}$ and the clean frame $\mathbf{x}_k$ are allowed to have different noise levels. Specifically $\mathbf{x}_k$ has λ_{max} which corresponds to a no noise being added into the frame. Also following [10], each block is modulated with FiLM [20]. In X-UNet, the FiLM modulation occurs by summing both the pose and noise encodings rather than just the noise. Here, the pose encodings are the camera rays with some transformation applied to them, as a result their encodings have the same dimensions as the frames.

Each block also utilizes self and cross attention layers. The self-attention layer is defined

as calling the multihead attention [15] passing the same frame's feature vector for the query and key-value pairs. In essence, this allows the model to learn which portions of the input vector are important. The cross-attention layer is also based in the standard multihead attention, except the query and key-value pairs are from the opposite frames' feature vectors. By allowing each frame to query each others' feature maps, the model is able to leverage more information about the scene from the conditioning view.

2.6 Stable Diffusion

As mentioned previously, the UNet network architectures used in diffusion models are very large, especially when incorporating attention modules [15] to more effectively learn image representations. Stable diffusion [14] offers a different diffusion technique in order to reduce the size of such networks.

The key contribution from [14] is introducing an encoder/decoder processing step that encapsulates the diffusion process. Effectively, this enables the diffusion process to be operate in a low-dimensional latent vector space representation of an image rather than the actual high-dimensional image space.

An RGB image $\mathbf{x} \in \mathbb{R}^{H \times W \times 3}$ is encoded by the encoder $\mathcal{E}$ into a latent representation $\mathbf{z} = \mathcal{E}(\mathbf{x})$, and $\mathbf{z}$ can be decoded back into the image space with the decoder $\bar{\mathbf{x}} = \mathcal{D}(z) = \mathcal{D}(\mathcal{E}(\mathbf{x}))$. The latent image representation $\mathbf{z}$ is in the vector space $\mathbb{R}^{h \times w \times c}$, where, importantly, the spatial resolution $h = H/f, w = W/f$ has been down sampled by a constant factor f. In practice, the scaling factor $f = 2^m$ where m is some positive integer. It is important to note that this is a lossy operation, i.e. the recovered $\bar{\mathbf{x}}$, while very similar, is not necessarily the same image as the input image $\mathbf{x}$.

There is a risk of having a latent space with an arbitrarily high variance. Some learned vector regularizations such as KL-reg or VQ-reg can be applied to reduce the variance. In KL-reg, the vector space is biased towards a standard Gaussian akin to VAEs [9]. Whereas in VQ-reg, a vector quantization layer is used inside the decoder.

Beginning with a clean image $\mathbf{x}$, the stable diffusion process then uses the encoder to get

its compressed latent $\mathbf{z}$. This latent is then then be passed as an input into a diffusion model, where the usual diffusion processes occur. The output of the diffusion model is then another vector $\bar{\mathbf{z}}$ inside the latent space, this vector is then passed into the decoder to retrieve the generated image $\bar{\mathbf{x}} = \mathcal{D}(\bar{\mathbf{z}})$.

The stable diffusion model also provides a method of conditioning the latent diffusion process. This involves a processing the conditioning information y with a domain specific encoder τ_θ that projects y into an intermediate representation. Then, this intermediate representation is used to condition the diffusion process by either being passed into a cross attention module in each UNet block or concatenated onto the latent vector.

The X-UNet model is a relatively large model. For example, an X-UNet model for images with a spatial resolution of 128×128 contains around 400 million parameters, 256×256 over 2.8 billion, and 1024×1024 over 30 billion. As a result, there is a strong incentive to finding methods to reduce model size.

With stable diffusion, the target spatial resolution can be down sampled by a factor of f. The space savings in terms of spatial resolution alone is directly proportional to f^2. With all other things equal, an $f = 2$ will reduce the space requirements for 256×256 from 2.8 billion to just 700 million. In actuality, the number of parameters can be even lower than 700M when considering that there will be fewer X-UNet layers.

3. METHODS

3.1 3DiM

3DiM aims to provide a simple framework using an image-to-image diffusion model to perform novel view synthesis. Following [5] and [1], given a distribution $q(\mathbf{x}_1, \mathbf{x}_2)$ for pairs of views for a given scene and poses $\mathbf{p}_1, \mathbf{p}_2 \in \mathrm{SE}(3)$, the forward Gaussian process from (5) is modified to increasingly add noise to the image as with the log signal-to-noise-ratio $\lambda = \log\left[\alpha^2/\sigma^2\right]$ with a decreasing cosine schedule $\lambda_{\text{min}} = \lambda_T < \lambda_{T-1} < \cdots < \lambda_0 = \lambda_{\text{max}}$ as in [21].

$$q\left(\mathbf{z}_k^{(\lambda)} | \mathbf{x}_k\right) = \mathcal{N}\left(\mathbf{z}_k^{(\lambda)}; \sqrt{S(\lambda)}\,\mathbf{x}_k, S(-\lambda)\boldsymbol{I}\right) \tag{16}$$

where $S(\cdot) : [\lambda_{\min}, \lambda_{\max}] \mapsto \mathbb{R}$ is a sigmoid function, and $\mathbf{z}_k^{(\lambda)}$ is a noised frame with a different pose with respect to the reference view $\mathbf{x}_k$. Again using the properties of the Gaussian like in (10), we reparameterize the noised frame

$$\mathbf{z}_k^{(\lambda)} = \sqrt{S(\lambda)}\,\mathbf{x}_k + \sqrt{S(-\lambda)}\,\boldsymbol{\epsilon} \tag{17}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$. Modifying the simplified training objective from (11) yields

$$\mathcal{L} = \mathbb{E}_{q(\mathbf{x}_1, \mathbf{x}_2)} \mathbb{E}_{\lambda, \boldsymbol{\epsilon}} \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta\left(\mathbf{z}_2^{(\lambda)}, \mathbf{x}_1, \lambda, \mathbf{p}_1, \mathbf{p}_2\right) \right\|_2^2 \tag{18}$$

where $\boldsymbol{\epsilon}_\theta$ is now a model trained to denoise frame $\mathbf{z}_2^{(\lambda)}$ given a clean reference frame from another pose $\mathbf{x}_1$.

Ideally the frames of a 3D scene can be modelled with the decomposition

$$p(\mathbf{x}) = \prod_i p(\mathbf{x}_i | \mathbf{x}_{<i}), \tag{19}$$

without any assumptions about conditional independence. Practically, we can at best generate a

view using a k-Markovian model that considers only k conditioning frames. Experimentally, [1] found that k small yields the best sample quality and decided to set $k = 2$.

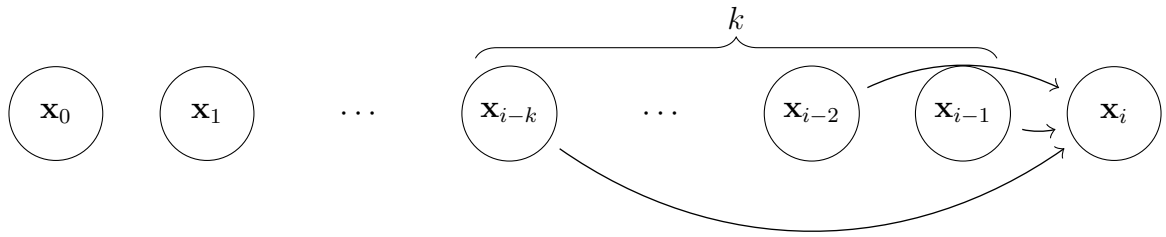

Figure 6: Illustration of generating a frame $\mathbf{x}_i$ with k-Markovian model.

Instead of sampling frames from the Markov process, [1] instead proposes a new method *stochastic conditioning*. In this process, we begin with an initial set of conditioning views $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_k\}$ with k small. As $\hat{\mathbf{x}}_{k+1}$ and $\mathbf{z}_{k+1}^{(\lambda_t)}$ are both known, the reverse of the forward process can be obtained by using Bayes' on the posterior distribution:

$$\hat{\mathbf{x}}_{k+1} = \left(\sqrt{S(\lambda_t)}\right)^{-1} \left(\mathbf{z}_{k+1}^{(\lambda_t)} - \sqrt{S(-\lambda_t)}\, \boldsymbol{\epsilon}_\theta \left(\mathbf{z}_{k+1}^{(\lambda_t)}, \mathbf{x}_i, \lambda_t, \mathbf{p}_1, \mathbf{p}_2\right)\right) \tag{20}$$

$$\mathbf{z}_{k+1}^{(\lambda_{t-1})} \sim q\left(\mathbf{z}_{k+1}^{(\lambda_{t-1})} \middle| \mathbf{z}_{k+1}^{\lambda_t}, \hat{\mathbf{x}}_{k+1}\right), \tag{21}$$

where each denoising step involves resampling $i \sim \text{Unif}\{1, \ldots, k\}$.

The key distinction from [10] is that each denoising step is conditioned against a different and random frame from the set $\mathcal{X}$, where as [10] conditions against the same clean frame at each step. After generating the new frame $\mathbf{x}_{k+1}$, we let $\mathcal{X} := \mathcal{X} \cup \{\mathbf{x}_{k+1}\}$, and repeat. Another deviation from [10] is that [1] uses only 256 denoising steps rather than 1000, and a bottle-neck with spatial resolution 8×8 instead of 4×4. But alike regular diffusion, the noised frame at λ_T can be given by $\mathbf{z}_i^{(\lambda_T)} \sim \mathcal{N}(0, \boldsymbol{I})$ and at the frame at λ_0 is sampled noiselessly.

Following [5], the Bayesian reverse posterior in (21) is given by

$$q\left(\mathbf{z}_{k+1}^{(\lambda_{t-1})}\middle|\mathbf{z}_{k+1}^{\lambda_t},\hat{\mathbf{x}}_{k+1}\right)=\mathcal{N}\left(\mathbf{z}_{k+1}^{(\lambda_{t-1})};\tilde{\boldsymbol{\mu}}_{t-1|t}\left(\mathbf{z}_{k+1}^{(\lambda_t)},\hat{\mathbf{x}}_{k+1}\right),\left(\tilde{\sigma}_{t-1|t}^2\right)^{1-\gamma}\left(\tilde{\sigma}_{t|t-1}^2\right)^{\gamma}\boldsymbol{I}\right) \tag{22}$$

where

$$\tilde{\boldsymbol{\mu}}_{t-1|t}\left(\mathbf{z}_{k+1}^{(\lambda_t)},\hat{\mathbf{x}}_{k+1}\right)=e^{\lambda_t-\lambda_{t-1}}(\alpha_{t-1}/\alpha_t)\mathbf{z}_{k+1}^{(\lambda_t)}+\left(1-e^{\lambda_t-\lambda_{t-1}}\right)\alpha_{t-1}\hat{\mathbf{x}}_{k+1} \tag{23}$$

$$\tilde{\sigma}_{u|v}^2=\left(1-e^{\lambda_v-\lambda_u}\right)\sigma_u^2 \tag{24}$$

and γ is a hyperparameter that controls the amount of noise added during the sampling procedure as in [21]. Finally, $\mathbf{z}_{k+1}^{(\lambda_{t-1})}$ can be sampled from the reversed posterior via

$$\begin{aligned}\mathbf{z}_{k+1}^{(\lambda_{t-1})}&=\tilde{\boldsymbol{\mu}}_{t-1|t}\left(\mathbf{z}_{k+1}^{(\lambda_t)},\hat{\mathbf{x}}_{k+1}\right)+\sqrt{\left(\tilde{\sigma}_{t-1|t}^2\right)^{1-\gamma}\left(\tilde{\sigma}_{t|t-1}^2\right)^{\gamma}}\boldsymbol{\epsilon}\\&=e^{\lambda_t-\lambda_{t-1}}(\alpha_{t-1}/\alpha_t)\mathbf{z}_{k+1}^{(\lambda_t)}+\left(1-e^{\lambda_t-\lambda_{t-1}}\right)\alpha_{t-1}\hat{\mathbf{x}}_{k+1}+\sqrt{\left(\tilde{\sigma}_{t-1|t}^2\right)^{1-\gamma}\left(\tilde{\sigma}_{t|t-1}^2\right)^{\gamma}}\boldsymbol{\epsilon}\end{aligned} \tag{25}$$

where $\boldsymbol{\epsilon}\sim\mathcal{N}(0,\boldsymbol{I})$.

3.2 Experiment

Again, the basic idea of this work is to take the existing 3DiM model proposed by [1] and reimplement it in the PyTorch framework instead of Jax in order to evaluate its performance under different conditions and hyperparameter settings. Some questions would be (1) comparing the model's performance across different resolutions and (2) determining the size and speed of the model to see if or how they can be improved. An added bonus would be accomplishing 3D (or view) consistency between synthesized views à la the stochastic conditioning method as described in [1].

3.3 Implementation

3.3.1 Architecture & Hyperparameters

The implementation of the X-UNet architecture was largely based off of the details and code provided with the supplementary material and in the appendices of [1]. The provided code was originally written for use in conjunction with the Jax [22] library. Due to unfamiliarity with Jax, it was decided that it would be best to port the base code and recreate the model using the PyTorch library instead. With PyTorch it was easier to refer to existing open-source implementations of other UNet and diffusion models to quickly bootstrap and verify the X-UNet model.

By and large, the PyTorch implementation stayed true to the Jax code in [1]. One major deviation, in terms of implementation, is that this implementation uses the built-in PyTorch `MultiheadAttention` module. The authors of [1] opted to write a custom attention module utilizing the generalized dot product attention from [15]. Reviewing the code in [1] did not reveal any obvious difference from the standard description of multi-head attention.

The model's hyperparameters again followed that of [1]. For images with a spatial resolution of 128×128, 4 X-UNet layers were used with channel depths of $256 \times [1, 2, 2, 4]$ were used. Each X-UNet layer down samples the input by a factor of 2 and eventually leads to a bottle-neck with spatial resolution of 8×8, which was noted by [1] to produce perceptually acceptable results.

The signal-to-noise ratio is defined using the cosine-shaped schedule from [5] with parameters $\lambda_{\text{min}} = -20$ and $\lambda_{\text{max}} = 20$:

$$\lambda_t = -2 \log \tan(at + b), \quad b = \arctan \exp\left(-\tfrac{1}{2}\lambda_{\text{max}}\right), \quad a = \arctan \exp\left(-\tfrac{1}{2}\lambda_{\text{min}}\right) - b, \tag{26}$$

where t is the time-step normalized to the unit interval $[0, 1]$. It should be noted that α_t and σ_t from [5] and used in the ancestral sampler (25) is given by

$$\alpha_t = \cos(at + b), \sigma_t = \sin(at + b). \tag{27}$$

3.3.2 Data-set & Training

All images used in training the model for novel view synthesis were generated by rendering the SRN ShapeNet data-set [7]. Due to computation and time constraints, it was decided to not follow [1] in training the model against *all* of ShapeNet, and rather focus on a specific subset of the taxonomy.

To validate that the PyTorch code was able to perform diffusion, it was first trained on the CelebA-HQ data-set [8]. In this experiment, the model synthesized faces at a 128×128 resolution with the standard hyperparameters and the cameras' intrinsics and poses were set to the identity.

Initially, the model was trained on 64×64 spatial resolution images, obtained from the NMR renderings [6] of the ShapeNet data-set. This was to validate that the ported PyTorch code was able to perform diffusion, and potentially synthesize novel views. Each model in the NMR data-set had 24 rendered views moving in a circular fashion around the object. Due to file storage limitations on our clusters, we decided to limit ourselves to just the "vessels" taxonomy of the data-set.

After validating the model's correctness, data-sets with larger spatial resolutions were used. This involved rendering the ShapeNet models at the desired custom resolution with Blender through the Stanford ShapeNet Renderer script. Minor modifications were made to export the intrinsic K and extrinsic $[R \mid t]$ matrices from the cameras in the Blender scene. With this script, 12 views in circular fashion of each model belonging to the "chairs" taxonomy were rendered; the camera poses of the 12 views remained consistent for each model. No changes were made to the cameras' calibration between models, meaning that the entire data-set shared the same camera intrinsics. The models in this data-set were randomly split with a 83/12/5 ratio for training, validation, and testing.

After getting subpar results from the low resolution vessels experiment, it was decided to use the chairs ShapeNet category instead. The chairs category had approximately 4 times as many models when compared against the pre-rendered NMR vessels. It was hypothesized that having a larger data-set from the chairs would improve sample quality in addition to raising the spatial

resolution of the images.

An experiment done in [1] but not here was to perform hue modulation on the inputs to the network to reduce the over-fitting to the training data.

All training of the models and rendering of the 3D ShapeNet scenes were performed using the Texas A&M HPRC clusters on A100 GPUs. All of results shown are synthesized from the same clean reference frame, instead of using the stochastic conditioning method and incrementally synthesizing views from smaller changes in camera pose. Also, classifier-free guidance for training the model was not implemented at the time of writing.

4. RESULTS

All synthesized results are generated using reference views inside of the testing split of the data-set. Additionally, no real-world scenes were used due to the model's extremely limited resolution.

First, the validation with 128×128 CelebA [8] data-set. This model was to verify that the PyTorch implementation was able to perform standard diffusion with a standard 2D image data-set, and as a result it was neither trained to completion nor with the optimal set of hyperparameters for this type of diffusion task which explains the lower perceptual quality of these results. The following faces are synthesized using the standard 1000 denoising steps from prior work [10], instead of the 256 used in [1] for novel view synthesis.

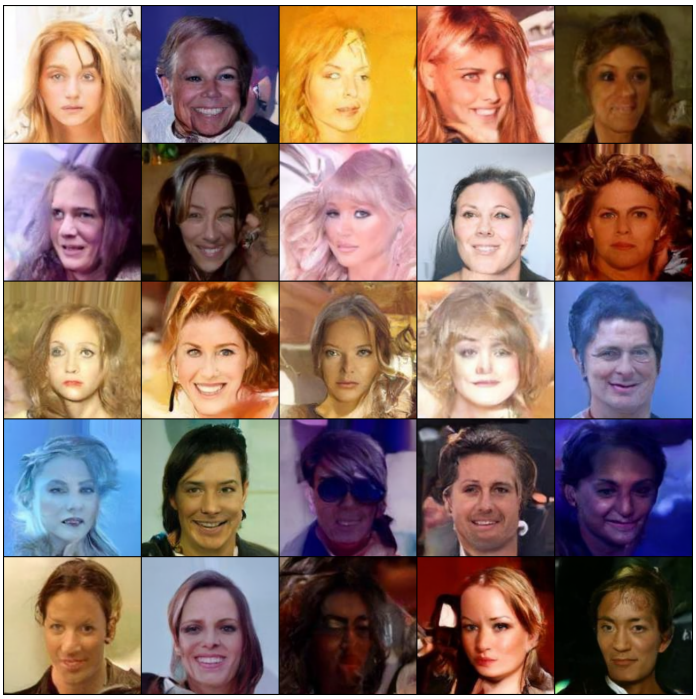

Figure 7: Synthesized faces with diffusion and the X-UNet with CelebA dataset. Pure Gaussian noise was used as the conditioning frame, and the pose inputs $\boldsymbol{p}_1 = \boldsymbol{p}_2 = \boldsymbol{I}_{3\times4}$.

Next, a model trained at 64×64 with the "vessels" ShapeNet category. It is speculated that due to the limited spatial resolution of the model, it is hard to discern and pick out features of any object. As a result, many of the synthesized views at this resolution were not perceptually coherent. The synthesized views that appear to be some-what acceptable in quality generally do not represent the original object, such as in the case of Figure 8. In that figure, the conditioning view is almost at top-down with the vessel facing the viewer. From this pose and with the limited resolution, it is hard – even for a human – to discern what kind of vessel it is and what kind of materials it is composed of.

One experiment that was not conducted, but could possibly improve results, is to instead synthesize a novel view starting with $K = |\mathcal{X}| > 1$, or in a few-shot setting instead of a single-shot. Ideally, this experiment will allow the model to leverage more information about the scene when limited by spatial resolution. Doing this will require fine-tuning of many hyperparameters, as the authors of [1] have found that higher K correlates with lower sample quality.

Finally, the results of 3DiM at a resolution of 128×128 on the "chairs" ShapeNet category. The increased resolution gives more information about the scene for the model to work with. This is likely the reason why in these chair samples are able to preserve shape and texture when compared against the vessels.

Initially, the model was being trained on renders with transparent backgrounds that were later interpreted as a black background by the model. This led to strange artifacts in the results, possibly due to the fact that there was very little contrast between the shadowed regions of a model and the black background. However, some images were still able to yield acceptable sample quality Figure 9.

(a) Conditioning view $\boldsymbol{x}_0$ **(b)** Target view

(c) Illustration of the diffusion denoising process to generate a novel view.

(d) Other attempts in synthesizing the same view with different input noise.

Figure 8: Synthesis of novel views of a vessel.

Unfortunately, this implementation of the model often yields low quality results as seen in Figure 10. There are many factors that may contribute to this, such as training time, batch sizing, incorrect hyperparameters or implementation. Due to time and computing constraints, not all of these possibilities were able to be ruled out. Regrettably, again due to time, we were unable to train a model of the vessels category at 128×128 resolution to directly compare against the smaller 64×64 model to definitively answer the question about sample quality being negatively impacted by low resolutions.

Figure 9: Synthesized novel views of various chairs.

Figure 10: Failures in synthesizing a novel view of chairs.

5. CONCLUSION

5.1 Remarks

This implementation of the 3DiM method is able to synthesize some novel views given a single view of an existing 3D scene. However, there are a lot of areas that can still be improved, such as sampling quality and model size.

5.2 Limitations

This implementation had many limitations, such as a very limited image resolution and the slow sampling process that is typical of diffusion models. Another limitation that was not addressed is the large model size which was the major limiting factor in training.

While we did not test the sample qualities with differing batch sizes, we suspect that there may be some correlation between batch-size and

5.3 Future Work

5.3.1 Stable Diffusion

Diffusion models like the X-UNet used in 3DiM are relatively large, often requiring hundreds of millions or billions of learnable parameters for relatively low resolution images. One technique previously discussed, stable diffusion, aims to reduce the dimensionality of the input image by passing it through an encoder and performing diffusion in the resulting latent space. This latent representation has the benefit of being some squared factor f^2 less in spatial dimensionality compared to the original input image. This would yield some benefits in reducing the size of the UNet models used in diffusion.

A consideration to make are that these latent representations are inherently lossy. This can manifest in artifacts with high frequency details such as reflections or text to being lost in the process. Overall, this method may only work with novel view synthesis if beginning with a very large resolution and small scaling factor like $f = 2$.

5.3.2 Distillation

Typically in diffusion, it takes many sampling steps to synthesize an image – typically on the order of thousands or tens of thousands. Some recent work has been made in "distilling" these diffusion models [5], in which a teacher model with N steps can be reduced to new student model requiring only $N/2$ steps. In the extreme, a model with N steps can be reduced to as few as 4 or even a single step process with minimal loss to the overall sample quality when compared against the original model.

This method is an attractive method of speeding up the X-UNet model used in the 3DiM process. The construction of the model is already based on the log signal-to-noise ratios as defined in progressive distillation by [5]. Then, it should be trivial to apply the teacher-student model to reduce the number of steps from 256 to a smaller number. However, in the 3DiM process, stochastic conditioning is a required step in maintaining 3D consistency between synthesized novel views. In this stochastic conditioning, each denoising step is conditioned against a different clean frame of the same scene. This requirement may prevent the ability to significantly reduce the number of steps, but even a reduction by a factor of 2 is very attractive.

5.3.3 In-painting

In other works involving novel view synthesis such as MPI methods like [4], a reference view is warped using some transformation and only the previously occluded regions are generated via some guided in-painting process.

This is contrast to the focus of this work, which was to synthesize an entirely new view given a reference frame of a 3D scene. This required synthesizing all $H \times W$ pixels of an image of the scene, rather than some smaller subset of pixels. By selectively in-painting a subset of pixels in the scene, this can reduce the amount of time required to synthesize a novel view.

There has already been work in applying diffusion with in-painting tasks [12], so it may be possible to apply those findings in the context of novel view synthesis.

5.3.4 Classifier-Free Guidance

Due to time constraints, this work did not implement classifier-free guidance that is commonly used in guided or conditional diffusion models. In VAEs or flow models, it is possible to do truncated or low temperature sampling with decreased variance of the noisy inputs. This has the effect of sacrificing some mode coverage of the model while boosting individual sample quality. However, simply applying this same method of reducing variance to diffusion negatively impacts the sample quality – often producing results that are blurry and low in quality.

This trade-off is particularly attractive in diffusion models, as they can already generate high quality samples and have a larger mode coverage when compared against GANs or flow models. Even if the diffusion model's mode coverage is restricted, the model can potentially create extremely high quality samples with more coverage compared to other existing generative models.

Classifier-free guidance [23] is a method that attempts to accomplish the same effects as low temperature sampling with diffusion models, without the loss of sampling quality. This is a simple method to implement for the 3DiM method. For a small probability $\sim 10 - 20\%$ of the time, pure Gaussian noise should be passed as the condition instead of a clean reference frame of the scene in order to unconditionally synthesize a view. The model's weights with conditional and unconditional synthesis can then be mixed in order to maximize sample quality.

In contrast to [1] where they trained against the entirety of the ShapeNet data-set, this work only trained the the 3DiM model against a single class of ShapeNet objects, i.e. only chairs or vessels. Applying classifier-free guidance in this case might not have yielded the same benefits in our case due to the limited nature of the data-set.

REFERENCES

[1] D. Watson, W. Chan, R. Martin-Brualla, J. Ho, A. Tagliasacchi, and M. Norouzi, "Novel view synthesis with diffusion models," 2022.

[2] E. R. Chan, C. Z. Lin, M. A. Chan, K. Nagano, B. Pan, S. D. Mello, O. Gallo, L. Guibas, J. Tremblay, S. Khamis, T. Karras, and G. Wetzstein, "Efficient geometry-aware 3D generative adversarial networks," in *arXiv*, 2021.

[3] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," in *ECCV*, 2020.

[4] R. Tucker and N. Snavely, "Single-view view synthesis with multiplane images," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[5] T. Salimans and J. Ho, "Progressive distillation for fast sampling of diffusion models," 2022.

[6] H. Kato, Y. Ushiku, and T. Harada, "Neural 3d mesh renderer," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

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

[8] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[9] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings* (Y. Bengio and Y. LeCun, eds.), 2014.

[10] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, 2020.

[11] C. Saharia, J. Ho, W. Chan, T. Salimans, D. J. Fleet, and M. Norouzi, "Image super-resolution via iterative refinement," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 4, pp. 4713–4726, 2023.

[12] C. Saharia, W. Chan, H. Chang, C. Lee, J. Ho, T. Salimans, D. Fleet, and M. Norouzi, "Palette: Image-to-image diffusion models," in *ACM SIGGRAPH 2022 Conference Proceedings*, SIGGRAPH '22, (New York, NY, USA), Association for Computing Machinery, 2022.

[13] J. Ho, T. Salimans, A. Gritsenko, W. Chan, M. Norouzi, and D. J. Fleet, "Video diffusion models," 2022.

[14] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10684–10695, June 2022.

[15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, Curran Associates, Inc., 2017.

[16] W. Jang and L. Agapito, "Codenerf: Disentangled neural radiance fields for object categories," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12949–12958, 2021.

[17] V. Sitzmann, S. Rezchikov, B. Freeman, J. Tenenbaum, and F. Durand, "Light field networks: Neural scene representations with single-evaluation rendering," in *Advances in Neural Information Processing Systems* (M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, eds.), vol. 34, pp. 19313–19325, Curran Associates, Inc., 2021.

[18] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015* (N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, eds.), (Cham), pp. 234–241, Springer International Publishing, 2015.

[19] S. Zagoruyko and N. Komodakis, "Wide residual networks," in *Proceedings of the British Machine Vision Conference (BMVC)* (E. R. H. Richard C. Wilson and W. A. P. Smith, eds.), pp. 87.1–87.12, BMVA Press, September 2016.

[20] E. Perez, F. Strub, H. de Vries, V. Dumoulin, and A. C. Courville, "Film: Visual reasoning with a general conditioning layer," in *AAAI*, 2018.

[21] A. Q. Nichol and P. Dhariwal, "Improved denoising diffusion probabilistic models," in *Proceedings of the 38th International Conference on Machine Learning* (M. Meila and T. Zhang, eds.), vol. 139 of *Proceedings of Machine Learning Research*, pp. 8162–8171, PMLR, 18–24 Jul 2021.

[22] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, and Q. Zhang, "JAX: Autograd and XLA." Astrophysics Source Code Library, record ascl:2111.002, Nov. 2021.

[23] J. Ho and T. Salimans, "Classifier-free diffusion guidance," 2022.

APPENDIX: Additional Results