Exploring the Capabilities of Automatic 3D Model Generation: A Comparative Study

Bachelor's Thesis

in the applied computer science course of the faculty of business informatics and applied computer science at the Otto-Friedrich-University of Bamberg

Faculty of Computer Graphics

Author: Andreas Franz Schwab

Examiner: Prof. Dr. Sophie Jörg

Abstract

The continued emergence of generative models, deep-learning architectures, and data-driven methods has opened a new era of technological opportunity, particularly in the area of automated 3D model generation. Such advances have become increasingly important in a number of sectors, including gaming, virtual reality, medicine and architecture. Given the increasing demand for 3D objects in these industries, a comprehensive analysis of the underlying technologies is of paramount importance. This thesis attempts to fill this scientific gap by providing a systematic examination of the various methods for automatic 3D model generation.

Based on generative algorithms, machine learning and artificial intelligence, the study examines various techniques and methods that have gained acceptance in this field. The goal is to objectively evaluate their efficiency in creating 3D models that are not only accurate but also visually compelling. This analytical framework is focused on evaluating methods such as Variational Autoencoders (VAEs), Generative Adversarial Networks (GANs), Diffusion Models, and Neural Radiance Fields (NeRFs), among others.

The study employs a carefully designed experimental setup and uses specified performance metrics to quantitatively and qualitatively analyze the strengths and weaknesses of these methods. Through this critical evaluation, the thesis aims to provide both an academic resource and a practical guide for subsequent research and applications in the field of automated 3D model generation. Thus, the contributions of this thesis go beyond a purely academic investigation; it serves as a foundational resource that can both deepen existing understanding and inform future computer graphics methodology.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In today's digital landscape, the demand for 3D models is steadily increasing, driven by the need for immersive and realistic visual experiences. Researchers and practitioners have leveraged generative AI techniques to develop innovative methods that automate the process of creating 3D models. These innovations have the potential to reshape how we interact with digital environments and facilitate the simulation, analysis, and visualization of complex real-world phenomena.

Starting out in 3D synthesis can be a challenging experience, particularly for novices with limited prior knowledge. While directly applying the models outlined in Chapter 3 may appear straightforward, acquiring a more in-depth understanding of the diverse methodologies and their foundational principles greatly enriches the learning process. This deeper insight not only improves the practical application of these models but also opens up possibilities for further advancements in the field of automatic 3D model generation.

This thesis provides a comprehensive examination of advances in automated 3D model generation. It delves into a variety of models and technologies, offering a detailed analysis of their mechanisms, capabilities, and limitations. The study focuses on evaluating the technologies' effectiveness, emphasizing their proficiency in creating functionally robust and aesthetically appealing 3D models. This research contributes significantly to the fields of computer graphics and artificial intelligence, serving as a valuable resource for novices in automatic 3D model generation and inspiring future researchers. The insights gained from this comparative study aim to inspire further innovation in this rapidly evolving field, pushing the boundaries of what is possible in 3D modeling and opening up new avenues of exploration and discovery in 3D synthesis.

To provide a foundation for this exploration, a comprehensive examination of the fundamentals of 2D generative AI is undertaken. Essential generative models such as Variational Autoencoders (VAEs) [Kingma and Welling, 2022; Rezende et al., 2014], Generative Adversarial Networks (GANs) [Goodfellow et al., 2020] and Diffusion Models [Yang et al., 2022; Ho et al., 2020; Sohl-Dickstein et al., 2015] are explained. Additionally, a brief introduction in Contrastive Language-Image Pre-training (CLIP) [Radford et al., 2021] and various forms of 3D data representation is given. These fundamental concepts provide the necessary foundation for a subsequent in-depth analysis of 3D model generation techniques.

The research further evaluates different approaches to generating 3D models, including methods based on images, text input, and video sequences. Each method presents unique

challenges and opportunities, and they are thoroughly examined through a comparative study. This investigation incorporates well-defined experimental setups and the application of rigorous performance metrics, such as accuracy, efficiency, and realism, to conduct comprehensive results analyses.

Additionally, this thesis addresses future opportunities in the field of 3D modeling. It highlights emerging trends and potential research directions that promise to redefine the 3D modeling landscape. The practical implications of these findings are also considered, underscoring their significance in real-world applications.

By providing a nuanced and detailed exploration of automated 3D model generation, this thesis contributes to the understanding and advancement of this dynamic and impactful field.

//TODO briefly explain test metrics / how evaluated.

Chapter 2

Basics

The chapter provides the groundwork necessary for the comparative analysis of automatic 3D model generation techniques by introducing the key technologies that drive these methods. It is essential to have a common understanding of the 2D generative models and 3D data representations that form the basis of this field.

The chapter begins by offering a brief overview of the most prevalent text-to-2D generative models, as they play a pivotal role in the process of 3D synthesis. This includes Variational Autoencoders (VAEs) [Kingma and Welling, 2022; Rezende et al., 2014], which are fundamental in providing probabilistic frameworks for learning complex data representations. Following this, Generative Adversarial Networks (GANs) [Goodfellow et al., 2020] are introduced, emphasizing their unique training mechanics involving both generator and discriminator components. Additionally, the chapter explores the realm of Diffusion Models, with a specific focus on Denoising Diffusion Probabilistic Models (DDPMs) [Ho et al., 2020; Sohl-Dickstein et al., 2015], while also touching upon Stochastic Gradient Methods (SGMs) [Song and Ermon, 2019] and Stochastic Differential Equations (SDEs) [Song et al., 2020, 2021].

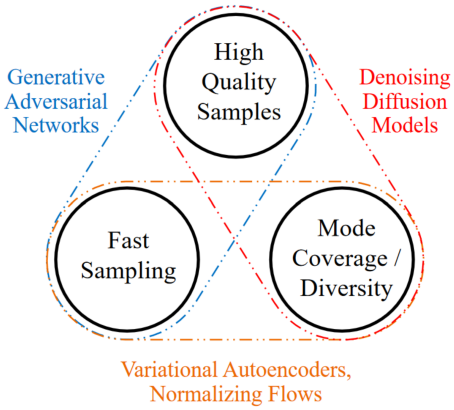

Figure 1: The Generative Learning Trilemma: Balancing Quality, Speed, and Diversity in Generative Models. [Xiao et al., 2022]

The figure above from Xiao et al. illustrates the "generative learning trilemma", effectively capturing the trade-offs among high-quality sample generation, fast sampling, and

mode coverage/diversity in these models. Generative Adversarial Networks are noted for their fast and high-quality samples, Denoising Diffusion Models excel in mode coverage/-diversity and sample quality, and Variational Autoencoders balance between fast sampling and diversity in their outputs.

Furthermore, the chapter delves into Contrastive Language-Image Pre-training (CLIP) [Radford et al., 2021], illustrating its effectiveness in bridging the gap between natural language and visual data, which is crucial for text-guided 3D model generation. The final part of the chapter is devoted to examining various forms of 3D data representation, such as Meshes, Point-Clouds, and Voxels. Understanding these representations is key to comprehending the structural makeup of 3D objects in computational settings.

//TODO check Final Diffusion Model

2.1 Variational Autoencoders – VAEs

Generative models play a pivotal role in the field of machine learning, particularly in tasks involving the synthesis and understanding of complex data distributions. Among the diverse types of generative models, Variational Autoencoders (VAEs) stand out for their unique approach and application. These models have gained prominence for their ability to efficiently generate new data samples while capturing the structures of complex data distributions. VAEs operate on the principle of encoding input data into a latent space, a compressed representation of the input data containing only the most imporant features, and then reconstructing the input from this space. This process is governed by a explicit probabilistic framework, where the model is trained to maximize the likelihood of the data under the learned distribution. This explicit modeling of data distribution in the latent space not only facilitates the generation of new data but also provides valuable insights into the data's inherent characteristics.

VAEs are essentially based on the architecture of Autoencoders, which apply an iterative process to identify the optimal encoder and decoder pair, aiming to minimize the reconstruction loss while preserving important information after dimensionality reduction. The encoder compresses the input data into a low-dimensional representation, or "code" [Hinton and Salakhutdinov, 2006; Goodfellow et al., 2016], which captures the most relevant features of the input, within a latent space. The decoder's role is to reconstruct the original input from this compressed latent representation. The reconstruction error, which is the discrepancy between the output and the original data, is then used to backpropagate and optimize the model's weights. This process, as described in works by Hinton and Salakhutdinov, Goodfellow et al., and Michelucci, strikes a balance between data compression and information preservation.

However, a notable limitation arises when the encoder and decoder become too closely matched. In such instances, the latent space becomes less controllable, leading to challenges in interpretability and regularity [Michelucci, 2022]. Autoencoders, by design, focus on approximate replication of input data rather than perfect replication. This approach necessitates the model prioritizing certain features of the input over others, often leading to the discovery of useful data properties [Goodfellow et al., 2016]. This dimensionality reduction proves beneficial in enhancing classification tasks' efficiency by reducing computational costs and memory overhead, and improving information retrieval in low-dimensional spaces [Goodfellow et al., 2016]. However, traditional autoencoders fall short

in generating new data due to the unregulated nature of the latent spaces.

To understand why traditional autoencoders struggle in generating new outputs, consider the analogy of a bag full of numbered balls. If one is tasked with randomly selecting a ball from this bag, the likelihood of picking a specific number is quite low, especially if the bag contains a large number of balls. Each ball in this analogy represents a vector in the latent space of an autoencoder. When an autoencoder extracts features from the input data, the resulting latent space is not perfectly structured; hence, many parts of this space might not correspond to meaningful or recognizable data patterns. This is akin to having many numbers in the bag that do not match the desired number. Thus, when an autoencoder attempts to generate new data by randomly sampling from this unstructured latent space, the likelihood of producing a meaningful output is similarly low. This challenge is further compounded by the fact that the structure of the latent space is influenced by various factors, including the distribution of the source space, the dimensionality of the latent space, and the architecture of the encoder. [Michelucci, 2022].

In essence, traditional autoencoders are not designed to generate new data; their main function is to copy and reconstruct the given input [Goodfellow et al., 2016].

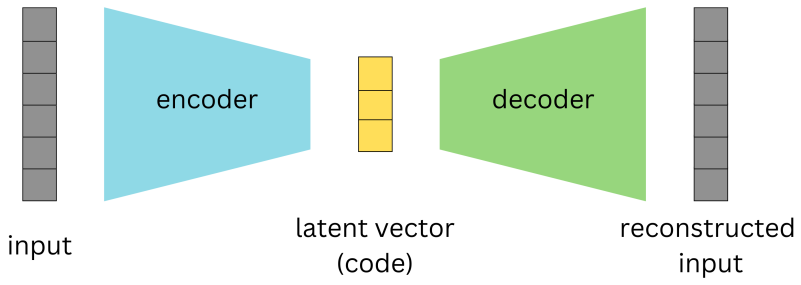

Figure 2: Schematic of an Autoencoder: Demonstrating the process of dimensionality reduction to a latent vector and subsequent reconstruction, aiming to minimize reconstruction loss. TODO ADD LOSS https://towardsdatascience.com/difference-between-autoencoder-ae-and-variational-autoencoder-vae-ed7be1c038f2

At the core of Variational Autoencoders is the handling of the latent space. Unlike traditional autoencoders that encode data to a single point, VAEs employ a probabilistic approach, encoding data into a distribution characterized by parameters μ (mean) and σ (standard deviation). This approach not only allows for nuanced manipulation of data within the latent space but also paves the way for data generation. VAEs address the limitations of traditional autoencoders by implementing enhanced regularization within the latent space during training. This is achieved by encoding a distribution over the latent space, instead of encoding to a single deterministic point [Doersch, 2016]. A key element in this process is the sampling of a point from this distribution to represent the latent variable z. This sampled point is then utilized in the reconstruction of the input data, with the resulting reconstruction error being backpropagated to update the model's weights.

To revisit the analogy of the bag full of numbered balls, VAEs transform the scenario.

Instead of a bag with a jumble of numbers, VAEs create a more organized bag where regions are filled with specific ranges of numbers. When a ball (a point in the latent space) is selected at random, there's a higher likelihood of it correlating with a meaningful and recognizable pattern, enabling the generation of coherent outputs. This organization within the latent space is key to VAEs' ability to generate new, meaningful data.

A primary objective of VAEs is the computation of data likelihood, $P(X)$, through strategic sampling of latent variables z. This aligns with the principle of maximum likelihood, which posits that a model likely to produce samples similar to the training set is preferable. The function $Q(z|X)$ plays a crucial role here, predicting the most probable latent variables z that could have generated a given data point X [Doersch, 2016]. The core equation of VAEs, as given by Doersch, encapsulates this relationship:

$$\log P(X) - D_{KL}\left[Q(z|X) \parallel P(z|X)\right] = \mathbb{E}_{z\sim Q}\left[\log P(X|z)\right] - D_{KL}\left[Q(z|X) \parallel P(z)\right]$$

This equation first aims to maximize the data's log likelihood ($\log P(X)$), indicating the model's effectiveness in accurately recreating data. This term reflects how well the model captures the actual data distribution. Alongside this, the Kullback-Leibler (KL) divergence D_{KL} serves as a crucial regularization factor. It ensures the model's approximate posterior distribution $Q(z|X)$ closely aligns with the true posterior distribution $P(z|X)$, maintaining the integrity of the latent space representation [Doersch, 2016]. By minimizing this divergence, the model ensures that its encoding of the input data into the latent space is as accurate as possible On the right-hand side, the equation assesses the expected log-likelihood of the reconstructed data from latent variables z, focusing on the model's decoding accuracy. This ensures that the model can effectively translate the latent space representation back into usable data. Simultaneously, the equation uses another KL divergence term between the posterior $Q(z|X)$ and the prior distribution $P(z)$. This divergence helps to regulate Q, ensuring that the latent space corresponds to the prior distribution, which is usually chosen to enable efficient learning and prevent overfitting.

The optimization of these elements is performed by stochastic gradient descent, where the parameters of Q are iteratively adjusted to find the optimal balance that allows precise encoding of the input data into the latent space while ensuring a high-quality reconstruction [Doersch, 2016]

Despite their advanced capabilities, VAEs have some limitations. A common observation, as noted by Goodfellow et al., is that the generated samples can often appear blurry. This blurriness might stem from the optimization process, specifically the minimization of the Kullback-Leibler divergence. This optimization might lead the model to assign high probabilities to training set points and other less distinct points, resulting in blurry images [Goodfellow et al., 2016]. The Gaussian distribution often used in VAEs for the generative model may also contribute to this effect, as it can ignore minor features in the input data [Goodfellow et al., 2016]. Additionally, VAEs typically utilize only a small portion of the latent space, potentially limiting the quality of generated images [Goodfellow et al., 2016]. The performance of the model is also sensitive to the choice of priors for the latent space, making hyperparameter tuning an essential aspect of working with VAEs [Kingma and Welling, 2022; Higgins et al., 2017].

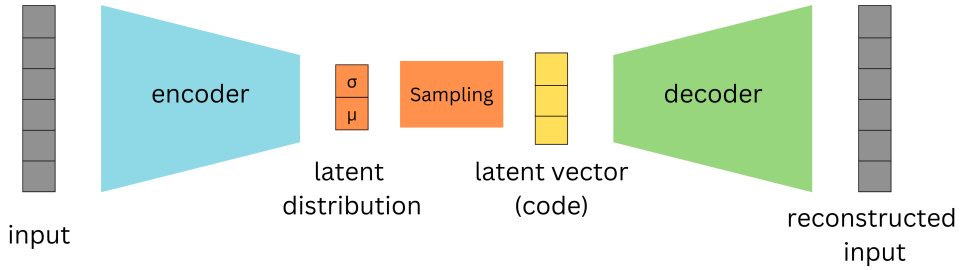

Figure 3: Functionality of a Variational Autoencoder: This figure illustrates the incorporation of a latent distribution, characterized by mean and standard deviation, for enhancing latent space regularization, enabling more effective and diverse data generation.

2.2 Generative Adversarial Networks – GANs

"When a deep neural network is used to generate data, the corresponding density function may be computationally intractable" [Goodfellow et al., 2020]. Unlike traditional generative models, implicit generative models do not require the explicit design of a density function to describe the patterns in the data. Instead, they use a sample generation process that produces new samples resembling the existing ones [Goodfellow et al., 2020]. Before Generative Adversarial Networks were introduced, the leading implicit generative model was the generative stochastic network, "which is capable of approximately generating samples via an incremental process based on Markov chains" [Goodfellow et al., 2020]. Markov chains are a way of describing a sequence of events or states, where probability of transition to the succeeding state is solely dependent on current states. This approach, however, can be time-intensive and may not always yield accurate results. GANs, on the other hand, directly generate high-quality samples in a single step, bypassing the gradual and often inefficient process of incremental generation.

The unique adversarial nature of GANs arises from the game-like competition between two neural networks: the generator and the discriminator. The generator is responsible for creating fake inputs or samples, which are then passed to the discriminator. The discriminator's role is to differentiate between real samples from the domain set and the fake samples generated by the generator.

In the initial training phase, the discriminator is trained on a dataset of real, unlabeled data, learning to identify the characteristics of authentic samples. As the discriminator becomes adept at recognizing genuine articles, the generator starts creating counterfeits, using random input vectors to produce imitations. The discriminator then evaluates these fakes and provides feedback. This feedback loop, involving sample creation and model adjustments, gradually refines the generator's output. Eventually, the generator becomes so proficient that its fakes are indistinguishable from real data, achieving what is known as a zero-sum game, where one network's gain is the other's loss [Goodfellow et al., 2020].

However, the utility of GANs extends far beyond just image generation. Their applications encompass a wide range of fields, demonstrating their versatility and significant impact. These applications include video frame prediction, which is essential in multime-

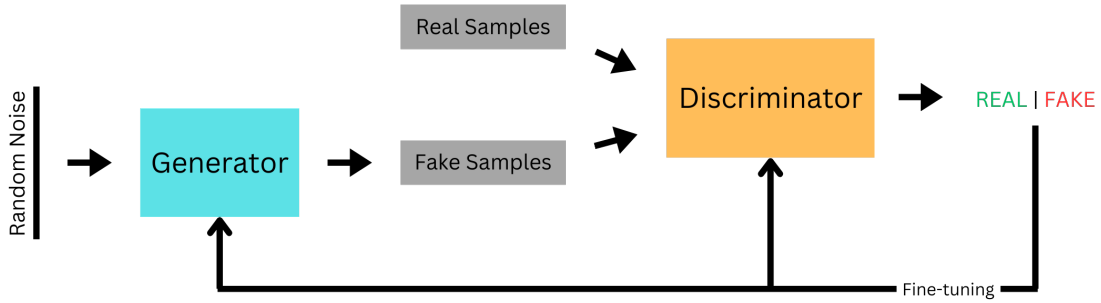

Figure 4: Illustration of the Generative Adversarial Network mechanism, highlighting the interplay between the generator and discriminator networks

dia applications; image enhancement, crucial in improving the quality and clarity of visual data; and encryption, where GANs contribute to the development of advanced security protocols [Goodfellow et al., 2020]. Additionally, GANs play a pivotal role in 3D object generation, the medical field,computer vision as well as traffic control systems, aiding in the development of smarter and more efficient transportation management [Aggarwal et al., 2021]. One of the most notable features of GANs is their ability to learn in an unsupervised manner, particularly through the generator network. Unlike traditional models that require a supervised learning set with labeled data, GANs can generate new data after training the discriminator with real examples. This capability allows GANs to produce realistic and varied outputs without direct exposure to or reliance on a large labeled dataset [Goodfellow et al., 2016],

Nevertheless, GANs pose a substantial challenge in their training process as they are hard to train [Goodfellow et al., 2020]. In addition, Brophy et al. highlight three important problems commonly associated with GANs, among others. These issues, namely non-convergence, diminishing or vanishing gradients, and mode collapse, contribute to the inherent instability experienced during GAN training. Non-convergence refers to the failure of a GAN model to stabilize and reach a state of equilibrium. Instead, it continuously oscillates and fails to converge to a satisfactory solution. As a result, the model does not learn the underlying patterns of the data and can even diverge, leading to poor performance [Brophy et al., 2023]. Diminishing or vanishing gradients occur when the gradients used to update the generator become extremely small or even vanish altogether. This phenomenon is often caused by an overly successful discriminator that becomes too adept at distinguishing real and fake samples. As a result, the generator struggles to learn from the feedback provided by the discriminator, impeding its ability to generate high-quality samples [Brophy et al., 2023]. Mode collapse happens when the generator collapses, meaning it focuses on producing only a limited set of samples or outputs, typically lacking diversity and variety [Salimans et al., 2016]. In such cases, the generator fails to capture the full range of patterns and characteristics present in the training data, resulting in uniform and repetitive samples that do not adequately represent the true distribution [Brophy et al., 2023].

2.3 Diffusion models

Diffusion models have emerged as a prominent method in generative modeling, offering distinct advantages over traditional models like Variational Autoencoders (VAEs) and Generative Adversarial Networks (GANs). Specifically addressing the limitations in data quality and generation efficiency that VAEs and GANs encounter, diffusion models present a novel approach. They operate by progressively perturbing data with noise and then learning to reverse this process, a methodology that enables the generation of highly realistic and diverse new samples.

Yang et al. [2022] distingueshe between three main approaches that dominate the study of diffusion models, which are going to be discussed shortly: Denoising Diffusion Probabilistic Models (DDPMs) [Ho et al., 2020; Sohl-Dickstein et al., 2015], Score-based Generative Models (SGMs) [Song and Ermon, 2019], and Stochastic Differential Equations (Score SDEs) [Song et al., 2020, 2021]. Each of these models has its own set of strengths and challenges, contributing uniquely to the development of Diffusion Models.

2.3.1 Denoising Diffusion Probabilistic Models

Central to the concept of Denoising Diffusion Probabilistic Models (DDPMs) are two Markov chains: the forward chain and the reverse chain, also known as the forward and reverse diffusion processes [Sohl-Dickstein et al., 2015]. These processes are illustrated in Figures 5 and 6.

The forward diffusion process, sharing some similarities with VAEs, focuses on a latent feature space of the initial data distribution. However, DDPMs differ in that the forward process in DDPMs "is fixed to a Markov chain that gradually [over a span of T steps] adds Gaussian noise to the data according to a variance schedule $\beta_1, \ldots, \beta_T$" Ho et al. [2020]. This process gradually pertubes the data's structure, eventually resulting in an image of pure noise, with the aim of gradually steering the data distribution towards a more manageable prior distribution [Yang et al., 2022; Poole et al., 2022].

The mathematical formulation of the forward process is given by Martínez et al.:

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I) \quad \text{where} \quad \sqrt{1-\beta_t}x_{t-1} = \mu_t \quad \text{and} \quad \beta_t I = \Sigma_t$$

In this equation, the model first adjusts the previous data point x_{t-1} to get x_t, the data point at the current step. This adjustment follows a Gaussian distribution and is done using the term $\sqrt{1-\beta_t}x_{t-1}$, which slightly reduces the intensity or strength of the previous data point [Sohl-Dickstein et al., 2015; Ho et al., 2020]. This controlled approach helps maintain a balance between the original data and the noise, ensuring that the noise doesn't overwhelm the data too quickly. Once this preparatory step is completed, the model then introduces noise. The level of noise added at each step is determined by the parameter β_t, where a higher value means more noise is added [Kingma et al., 2023]. The way noise is added is described by the covariance matrix $\beta_t I$, where I is the identity matrix [Croitoru et al., 2023]. This matrix ensures that noise is added to each element of the data in an independent and uniform manner, evenly distributing the noise across all parts of the data. The importance of this noise-adding process lies in its role in teaching the model the structure and characteristics of noise. By gradually adding noise to the

data, the model learns how images degrade step by step, knowledge that is crucial for the reverse process of DDPMs.

The process of adding noise over the entire sequence from the original data point x_0 to x_T is captured another formular by Martínez et al.:

$$q(x_{1:T}|x_0) = \prod_{t=1}^{T} q(x_t|x_{t-1})$$

Built upon the Markov property, the formula implies that each step depends solely on the previous step, allowing for a systematic and gradual transformation from x_0 to x_T [Martínez et al., 2023]. This methodical approach provides a detailed understanding of the data's evolution at each noise addition stage, giving a complete view of the transition probabilities throughout the forward diffusion process.

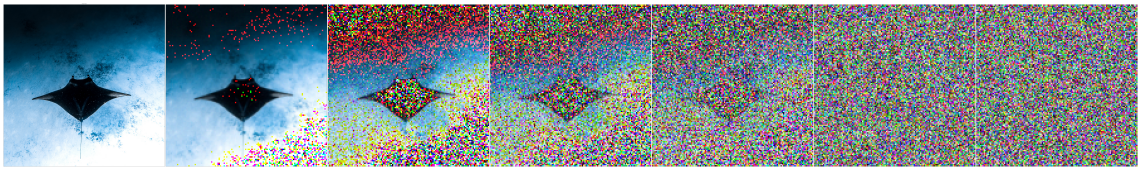

Figure 5: Illustration of the Forward Diffusion Process in DDPMs: This figure demonstrates the gradual addition of Gaussian noise to an image over multiple steps. Each subsequent image from left to right shows an increased level of noise, culminating in the far-right image, which represents a state of pure noise.

The reverse diffusion process, illustrated in Figure 6, employs a neural network parameterized by Θ to approximate the inverse of the forward process [Sohl-Dickstein et al., 2015; Yang et al., 2022]. It estimates the prior state of data points, x_{t-1}, from their current noisy state, x_t, using the probability distribution function $p_\theta(x_{t-1}|x_t)$, as given by Martínez et al.. This process is modeled as a normal distribution where the mean $\mu_\theta(x_t, t)$ and covariance $\Sigma_\theta(x_t, t)$ are determined by the neural network [Yang et al., 2022].

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t))$$

$$p_\theta(x_{0:T}) = p_\theta(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t)$$

The latter function $p_\theta(x_{0:T})$ is also taken from Martínez et al. and captures the probability of the entire data sequence under the reverse process, beginning with an estimate of the final noisy data point $p_\theta(x_T)$ and progressively reconstructing the data by removing noise at each step [Ho et al., 2020; Martínez et al., 2023]. Unlike the forward process that adds noise, the reverse process, starting from a state of random noise, uses the learned noise patterns to iteratively generate coherent images. This capability stems from the model's training to differentiate between noise and actual image features, enabling it to produce images that, while influenced by the training data, are not reproductions of the originals. The reverse process is also a Markov chain and involves the neural network learning to predict the reverse diffusion parameters Θ at each timestep [Yang et al., 2022]. The goal here is to ensure that the new samples it generates are statistically similar to

the original data it was trained on. This is done by maximizing the likelihood that these new samples belong to the same overall data distribution as the original set [Yang et al., 2022].

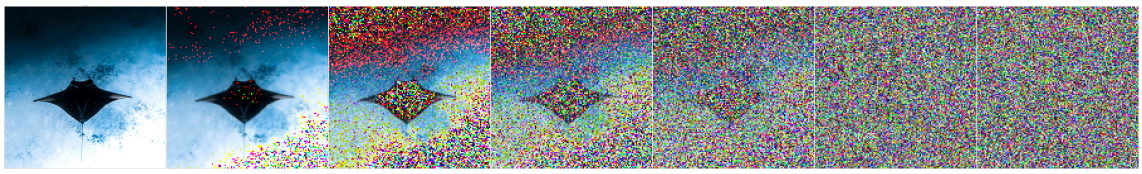

Figure 6: Visual Representation of the Reverse Diffusion Process in DDPMs: This figure illustrates the progressive removal of noise from a noisy state (right) back to the original or newly generated image (left), demonstrating the model's capability to reconstruct or create images by reversing the noise addition process.

In the reverse process, the neural network can be trained to predict one of three possibilities: the mean of the noise at each time step, the original image itself, or the noise of the image [Ho et al., 2020]. The second approach is not as advantageous as the "estimating small perturbations is easier than explicitly describing the entire distribution with a single, non-analytically normalizable potential function" [Sohl-Dickstein et al., 2015]. Focusing on the prediction of image noise is preferable because it allows a simple subtraction of the noise from the image, resulting in a less noisy version and thus also allowing an iterative generation of an image from the noise.

Despite their effectiveness, DDPMs are not without challenges. The most significant of these is the computational time required for generating new samples, which is due to "a Markov process [that] has to be simulated at each generation step, which greatly slows down the process" [Martínez et al., 2023].

2.3.2 Score-Based Generative Models

In the domain of generative modeling, Score-Based Generative Models (SGMs), as introduced by Song and Ermon distinguish themselves by prioritizing the learning of a score function using the Stein score [Liu et al., 2016]. The score is "the gradient of the log-density function at the input data point" [Song and Ermon, 2019], serving as a directional guide towards higher data density regions [Song and Ermon, 2019]. The score function is central in SGMs, akin to how DDPMs use a forward process of adding noise, but with a different purpose. In SGMs, the score guides the model to areas in the data space where the probability of finding similar data points to the training set is higher.

SGMs begin their learning process with a set of training data. Common data distributions indicate areas of high probability, while rare inputs usually result in low represented data regions. The role of SGMs is to learn how to work with that data space. In practice, the model $s_\theta(x)$ is trained to replicate the score function $\nabla_x \log p(x)$ of a data distribution $p(x)$ using so called score matching [Hyvärinen and Dayan, 2005]. This method, as Song describes, offers architectural flexibility by not requiring a normalizing constant. Song states that training of such models is achieved by "minimizing the Fischer divergence between the model and the data distributions" and represents this approach as:

$$\mathbb{E}_{p_{(x)}} \left[\|\nabla_x \log p(x) - s_\theta(x)\|_2^2 \right]$$

This divergence assesses how closely the model's score aligns with the actual data score by calculating their squared differences. The integration of score matching into this process means that no precise knowledge of the true data value is required [Song, 2021]. The goal here is to align the model's predictions with the real data's log-likelihood gradient as closely as possible. Under specific conditions, this method can reliably ensure that the model's score accurately reflects the true score of the data distribution [Song and Ermon, 2019].

In the generation of new samples, SGMs start with a random or noise image. Using Langevin dynamics, an iterative process described by [Roberts and Tweedie, 1996], the model samples from a distribution $p(x)$ using its score function [Song, 2021]. Following Song, it starts with a random prior distribution $x_0 \sim \pi(x)$, and iteratively updates the chain by:

$$x_{i+1} \leftarrow x_i + \epsilon \nabla_x \log p(x) + \sqrt{2\epsilon} z_i,$$

where each iteration updates the sample x_i by a small step size ϵ in the direction of the data's score function, guiding the sample towards the target data distribution. This process theoretically converges the distribution to $p(x)$ as ϵ becomes smaller and iterations increase [Song and Ermon, 2019; Song, 2021].

A notable challenge in SGMs arises when estimating the score function in less represented data regions. Inaccuracies here may hinder the convergence of Langevin Dynamics [Song and Ermon, 2019]. To mitigate this, Song and Ermon recommend introducing Gaussian noise to the data and estimating scores for these noise-altered distributions, an approach termed Noise Conditional Score Networks (NCSNs). This strategy prevents the data distribution from collapsing into a lower-dimensional structure, ensuring more robust sample generation [Song and Ermon, 2019].

The core idea of Noise Conditional Score Networks is to generate realistic data samples by effectively navigating through a series of noise-perturbed data distributions that gradually converge to the true data distribution. This process is facilitated by introducing Gaussian noise σ into the data at multiple levels [Song and Ermon, 2019]. The NCSN, represented $s_\theta(x, \sigma)$, is trained to estimate scores for these noise-perturbed distributions [Song and Ermon, 2019]. This training allows the network to adjust its predictions based on different noise levels to ensure accurate and relevant point estimates. Generating new samples begins with a state of pure noise and uses annealed Langevin dynamics in order to gradually reduce the noise level σ [Song and Ermon, 2019]. This method transforms the initial noise into realistic data points by iteratively gradually reducing noise, ending when a sufficiently low noise level yields a sample that accurately represents the modeled data [Song and Ermon, 2019].

2.3.3 Score-Based Generative Modeling through Stochastic Differential Equations – SDEs

Song et al. aim to combine both DDPMs and SGMs (NCSNs) using Stochastic Differential Equations (SDEs). The idea is to create a continuous diffusion process, indexed by time, that transforms a data distribution into a more tractable prior distribution. This is described through the following function by Song et al.:

$$dx = f(x, t)dt + g(t)dw,$$

The process is governed by two coefficients: a drift coefficient $f(x,t)$, governing the deterministic properties of the stochastic process, guiding how data evolves over time, and a diffusion coefficient $g(t)$, which scales the random noise introduced by Brownian motion dw (Wiener process) [Song et al., 2020]. This Brownian motion represents the random movement of particles in a fluid as they collide with fast-moving molecules in the fluid.

For generating new samples, a principle from Anderson [Anderson, 1982] comes into play. It states that "the reverse of a diffusion process is also a diffusion process, running backwards in time and given by the reverse-time SDE:" [Song et al., 2020].

$$dx = \Big[f(x,t) - g(t)^2 \nabla_x \log p_t(x)\Big]\, dt + g(t) d\bar{w}$$

This equation by Song et al. describes the process of recovering data from noise by moving backward in time. "Once the score of each marginal distribution, $\nabla_x \log p_t(x)$, is known for all t, we can derive the reverse diffusion process from [the above equation] and simulate it to sample from p_0" [Song and Ermon, 2019]. This score function essentially captures the essence of the data's probability distribution at various stages of noise addition.

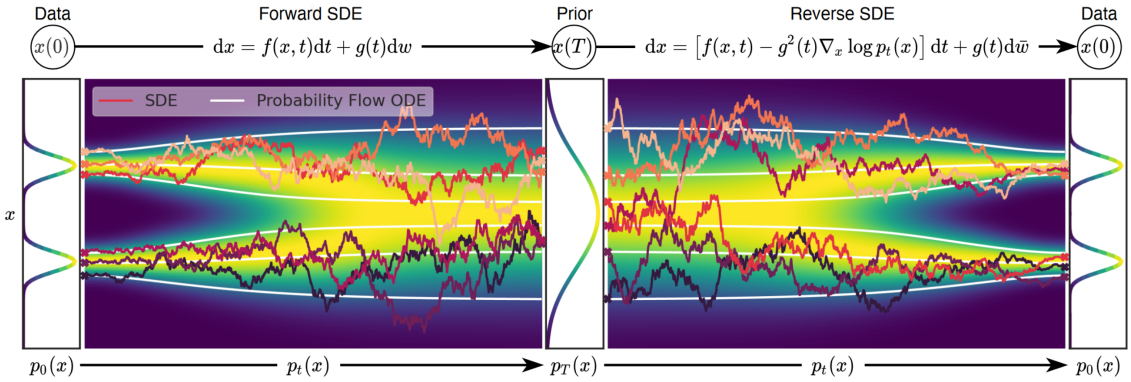

Figure 7: This figure illustrates the two-fold process in score-based generative modeling through SDEs. On the left, the Forward SDE represents the gradual transformation of data into noise, guided by the drift and diffusion coefficients. On the right, the Reverse SDE depicts the process of reconstituting original data from noise, leveraging the known score of each marginal distribution [Song et al., 2020]

Training a model to accurately estimate the score functions at various noise levels is a fundamental aspect of this process. To achieve this, the model is trained through score matching [Hyvärinen and Dayan, 2005], which involves fine-tuning the model to closely approximate these score functions across a spectrum of noise levels [Song et al., 2020]. The training objective, as stated below by Song et al., is formulated to find optimal model parameters θ^* that minimize the expected discrepancy between the estimated and true scores over time, which are influenced by the noise.

$$\theta^* = \arg\min_\theta \mathbb{E}_t \left\{ \lambda(t) \mathbb{E}_{x_0} \mathbb{E}_{x_t|x_0} \left\| s_\theta(x_t,t) - \nabla_{x_t} \log p_{0t}(x_t|x_0) \right\|_2^2 \right\}$$

The expectations are taken over time and modulated by a time-varying weighting function $\lambda(t)$. "With sufficient data and model capacity, score matching ensures that

the optimal solution for the above equation, denoted by $s_{\theta^*}(x,t)$ equals $\nabla_{x_(t)} \log p_t(x)$ for almost all x and t" [Song and Ermon, 2019]. The score matching process matches the output of the score network with the true gradient of the log-likelihood over the course of the SDE, enabling the generation of realistic data samples from complex distributions [Yang et al., 2022].

2.4 Contrastive Language-Image Pre-training – CLIP

Radford et al. address the limitations of traditional computer vision models, which are restricted by their training on a fixed set of object categories and lack adaptability to new tasks or concepts. To overcome these limitations, they propose a novel method called Contrastive Language-Image Pre-training (CLIP), which is an "efficient and scalable method of learning from natural language supervision" [Radford et al., 2021]. This process allows the model to learn a representation of the image that is grounded in natural language, enabling it to understand the content and context of the image.

Unlike traditional computer vision models that rely solely on annotated image datasets, CLIP leverages a large corpus of text and image pairs from the internet. It learns to associate images and their corresponding textual descriptions, allowing it to understand the relationship between visual and textual data. CLIP is built on a transformer-based architecture, which has proven highly effective for natural language processing tasks [Radford et al., 2021]. It consists of two main components: an image encoder and a language encoder. The image encoder processes images using a modified version of ResNet50 [He et al., 2016] or as a second approach was build upon the Vision Transformer (ViT) [Dosovitskiy et al., 2021], while the language encoder uses another modified transformer-based model to process textual descriptions [Vaswani et al., 2023]. By learning to associate images and text, CLIP acquires a generalized understanding of visual concepts and language semantics.

One of the remarkable aspects of CLIP is the ability for zero-shot capability. It can perform tasks without task-specific training. For example, given a natural language prompt, CLIP can recognize objects in images, generate captions, or perform classification tasks. Furthermore, CLIP has been shown to be very effective on a variety of tasks. It outperforms state-of-the-art models on the ImageNet classification task, and it achieves state-of-the-art results on the Visual Genome dataset for object detection and question answering [Radford et al., 2021].

However, there exist several limitations to CLIP. "The performance of zero-shot CLIP is often just competitive with the supervised baseline of a linear classifier on ResNet-50 features" [Radford et al., 2021]. This means that CLIP is not significantly better than a model that is trained on labeled data for the specific task at hand. In addition, the authors estimate that achieving SOTA performance across their evaluation suite would require significantly more computational resources, approximately a 1000-fold boost in computational power. Current hardware capabilities cannot accommodate such demands, emphasizing the requirement for advancements in hardware technology to effectively train zero-shot CLIP models.

2.5 Multilayer Perceptron – MLP

Multilayer Perceptrons (MLPs) form a fundamental class of artificial neural networks in machine learning, characterized by their layered structure, including input, hidden, and output layers [Noriega, 2005; Murtagh, 1991]. Central to their function is the activation function, typically the Logistic Sigmoid Function in MLPs, which transforms the weighted inputs into node outputs in a continuous and differentiable manner, facilitating gradient-based optimization [Murtagh, 1991; Noriega, 2005]. The learning process in MLPs is supervised, involving initial random weight assignment, followed by training through pattern presentation, output comparison, and backward error propagation, typically using gradient descent methods to minimize errors [Noriega, 2005]. This process is guided by the generalized delta rule or backpropagation, which adjusts network weights based on the error between predicted and actual outputs, allowing the network to effectively learn from its environment [Murtagh, 1991]. The architecture of an MLP, including the number of layers and nodes, along with the activation methods and learning techniques, significantly influences its performance. Balancing network complexity and the risk of overfitting is crucial in MLP design [Murtagh, 1991]. MLPs are versatile, capable of performing tasks like regression, mapping input vectors to values, and supervised classification, where input patterns are trained to produce specific output classifications [Murtagh, 1991]

2.6 Representation forms of 3D Data

2.6.1 Meshes, Point-Clouds, Vocels

//TODO

2.6.2 Neural Radiance Fields – NeRFs

Neural Radiance Fields (NeRFs), as introduced by Mildenhall et al., represent a significant advancement in 3D scene representation, particularly when compared to traditional methods which often struggle with complex geometries and varying lighting conditions. Emerging in response to these challenges, NeRFs employ a novel volumetric representation, capturing the spatial and angular distribution of light more accurately [Mildenhall et al., 2020].

Classic deep learning methods often require a comprehensive dataset comprising various scenes and their representations. NeRFs, on the other hand, are trained to specialize in a single, unique scene [Mildenhall et al., 2020]. The underlying neural network, MLP F_θ, consists of Fully Connected Layers (FCs) with ReLU activations, specifically designed to encode the volumetric details of that particular scene, effectively creating a dedicated neural network for each scene [Mildenhall et al., 2020].

The neural network accepts two types of input: a position in a given coordinate system, often expressed as a 3D vector x, y, z, and a viewing direction represented by two angles θ, ϕ. To better understand the concept of the latter two inputs, imagine a scenario where a flashlight is held in the middle of a dark room. The angle θ would represent how much the flashlight is tilted up or down. Similarly, the angle ϕ would represent how much the flashlight is rotated about the vertical axis while pointed outward. These angles help the neural network understand from which direction the beam is being cast to a point in

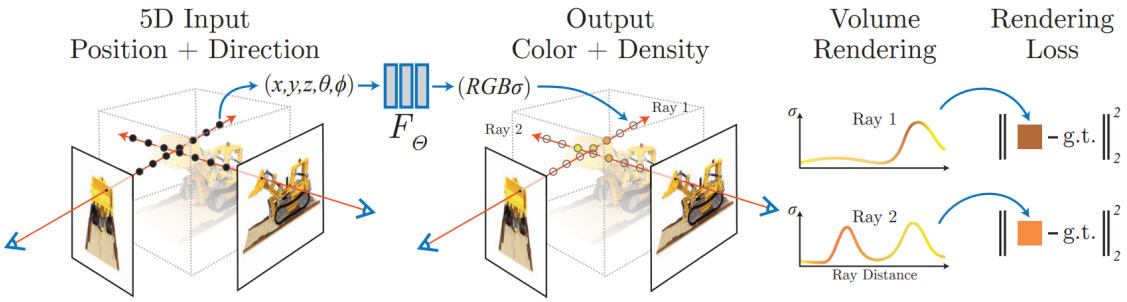

Figure 8: Sumamrized workflow of a NeRF: 5D input (Position + Direction) is processed by the MLP F_θ to output color and density. Volume rendering integrates predictions along rays to generate an image from the specified viewpoint. The rendering loss, comparing the rendered and actual images, guides the network's training and refinement process Mildenhall et al. [2020]

3D space. The network's output consists of the color c and density σ at that particular location [Mildenhall et al., 2020].

Central to NeRF's operation is the process of volume rendering, which involves casting rays through the scene and gathering color and density information at multiple points.

$$C(r) = \int_{t_n}^{t_f} T(t)\sigma(r(t))c(r(t), d)dt \quad \text{where} \quad T(t) = \exp\left(-\int_{t_n}^{t} \sigma(r(s))ds\right)$$

This formula, as outlined by Mildenhall et al., enables the rendering of a 3D scene by casting rays and aggregating their properties from a near boundary (t_n) to a far boundary (t_f). The equation essentially builds up the final image by integrating the effects of light interaction with the scene's material over the length of each ray. The volume density σ at a point in the scene indicates the amount of light-blocking material at that specific location. A higher density suggests a greater likelihood of a light ray being obstructed, signifying more material presence at that point. The accumulated transmittance $T(t)$ reflects the cumulative effect of density along a ray's path. It quantifies the extent to which light from the start of the ray can travel through the scene to a given point without being absorbed or scattered by the material. Its value decreases along the ray's path sa it encounters areas of higher density. Moreover, the term $c(r(t), d)$ denotes the color at a point along the ray, given the viewing direction d. The interplay of density and transmittance at each point along the ray determines if a ray continues through space or concludes upon encountering an object, with the color at this final point contributing to the rendered image [Mildenhall et al., 2020].

Volume Rendering enables capturing visual phenomena like lighting, reflections, and transparency, which are often challenging to model with traditional 3D reconstruction techniques. For each pixel in a desired image frame, the neural network is queried at multiple points along a ray projected through the scene to produce a curve representing the density of objects along the ray's path, as seen in Volume Rendering in Figure 8. The point at which this density curve rises significantly usually corresponds to an object in the scene, and the color at this point is what is rendered for that particular pixel. These

density and color curves can be visualized using graphs to illustrate how density and color vary along the ray's path [Mildenhall et al., 2020].

To address multi-view consistency, NeRF predicts color as a function of both location and viewing direction, while density depends solely on location. This separation acknowledges that density, unlike color, remains consistent regardless of viewing angle [Hu et al., 2023].

An integral aspect of setting up NeRFs involves solving the problem of identifying the camera's position and direction for each input image. Methods like Structure-from-Motion (SfM) and Simultaneous Localization and Mapping (SLAM) can address this issue [Wei et al., 2021]. Once these parameters are identified, new views can be synthesized by querying the neural network for color and density information along rays projected through the scene [Gerats et al., 2023].

Training a NeRF model, however, presents its own challenges. Without explicit density data, the model learns by minimizing the loss between predicted and observed values from input images. This is facilitated by the differentiable nature of the entire rendering pipeline, including ray casting, sampling, and color computation [Yariv et al., 2020].

Although naive NeRF models may not provide photorealistic results due to the lack of detail, several optimizations have been introduced to improve their performance. One notable improvement is the use of positional encoding techniques that deterministically map 3D coordinates and view directions to a higher dimensional space. This is achieved by using high-frequency features before inputting them to the multilayer perceptron (MLP), which helps optimize the neural radiation fields to better represent high-frequency scene content [Mildenhall et al., 2020]. Hierarchical volume sampling is another important optimization. This strategy involves two neural network systems, one coarse and one refined, which are jointly optimized during training. Initially, the coarse network sparsely samples the rays in the 3D scene, and based on this initial sampling, the refined or "fine" network is guided to perform a more detailed sampling. This hierarchical approach is critical because dense sampling is very computationally expensive. A two-tier system therefore helps manage computational resources while allowing detailed sampling along rays to obtain the density and color of the sampled points [Arandjelović and Zisserman, 2021].

One of NeRFs key advantages is its memory efficiency. For example, rendering a single scene with NeRF requires only about five megabytes of memory, which is in stark contrast to voxel grid renderings that require over 15 gigabytes for a comparable scene [Mildenhall et al., 2020]. This mismatch in memory requirements underscores NeRF's superior efficiency in terms of data storage and transfer, and represents a compelling advantage over traditional 3D rendering techniques [Mildenhall et al., 2020]. Remarkably, the memory requirement of the rendered scene is even smaller than that of the input images, making the model extremely efficient in data storage and transfer [Mildenhall et al., 2020].

However, the NeRF model is not free of limitations. A major challenge is the computational cost associated with training the neural network. The optimization process for a single scene may require about 100,000 to 300,000 iterations, equivalent to a training period of about one to two days, to converge, assuming the use of a single NVIDIA V100 GPU [Mildenhall et al., 2020]. While this does not require a data center, it does require a significant amount of time, making NeRF less suitable for scenarios that require rapid implementation. In addition, NeRF rendering is prone to sampling and aliasing issues

that can lead to significant artifacts in the synthesized images [Rabby and Zhang, 2023]. These artifacts arise from the limited sampling of the radiation field, resulting in inaccurate reconstruction of certain features, especially in scenes containing sharp edges or textures [Rabby and Zhang, 2023].

2.6.3 Deep Marching tetraheda – DMTet

In 3D model generation, there are two primary methods: implicit and explicit 3D representations [Shen et al., 2021]. Implicit methods, like Neuronal Radiance Fields (NeRFs) [Mildenhall et al., 2020], use mathematical functions to define shapes, capturing complex details such as the shape's orientation and volume. These representations, while rich in detail, usually require conversion into 3D meshes for practical use. Conversely, explicit methods involve directly defining 3D shapes using specific coordinates, like in the case of meshes, voxels, or point clouds. This approach is able to capture and show geometrical details. However, it can be less ideal for computational tasks, especially in machine learning, due to its irregularities [Michalkiewicz et al., 2019].

DMTet, or Deep Marching Tetrahedra [Shen et al., 2021], emerges as a method representing a pivotal step in converting neural implicit representations into explicit mesh forms. Unlike traditional approaches that often struggle with detailed 3D structures, DMTet leverages advanced algorithms to produce high-fidelity 3D meshes in "a new differentiable shape representation [...]" [Shen et al., 2021]. The conversion is not only limited to implicit to explicit, but also contains explicit to explicit, as for example the conversion of coarse voxel or noisy point clouds to detailed meshes [Shen et al., 2021]. DMTet uniquely combines the benefits of both implicit and explicit 3D representations, leveraging a novel hybrid 3D representation approach.

The method propesed by Shen et al. produces a deformable and differentiable tetrahedral grid based on a given Sign Distance Field (SDF). The first step is therefore always to convert the original input into such a form. Point clouds are transformed directly, while voxels undergo initial conversion into point clouds through sampling points on their surfaces, which can then be used for the calculation. In the tetrahedral grid, each vertex receives an initial predicted SDF value and a deformation offset to represent the surface using an implicit function [Shen et al., 2021]. The surface is then refined using subdivision and further "converted into an explicit [triangular] mesh with a Marching Tetrahedra (MT) algorithm, which we show is differentiable and more performant than the Marching Cube" [Shen et al., 2021]. The final step involves refining the mesh into "a parameterized surface with a differentiable surface subdivision module" [Shen et al., 2021].

DMTets allows for supervision on the surface which produces better results in contrast to NeRFs [Shen et al., 2021]. Moreover, it is efficient in terms of inference speed and output quality, producing explicit meshes suitable for interactive graphic applications [Shen et al., 2021].

Chapter 3

Models

There exist several different approaches to generate 3D Models from a 2D template. In this chapter I will provide detailed insights in some of these approaches.

3.1 3D from Text input

//TODO

3.1.1 Dreamfusion

DreamFusion, as introduced by Poole et al., marks a significant advancement in the field of 3D modeling. Utilizing Neural Radiance Fields (NeRF), DreamFusion employs a novel technique known as Score Distillation Sampling (SDS) to generate coherent 3D objects and scenes from a variety of text prompts. This approach diverges from traditional methods that depend on pre-existing images from multiple angles, as DreamFusion dynamically generates these images during training using a diffusion model.

Central to DreamFusion is the use of Differentiable Image Parameterization (DIP), as described by [Mordvintsev et al., 2018]. This technique enables the generation of images x through parameters θ and a differentiable generator g, offering refined optimization capabilities even at the pixel level [Poole et al., 2022]. This marks a shift from the conventional approach of diffusion models, which usually produce outputs similar to their training data. In DreamFusion, the parameters θ define 3D volumes, with g functioning as a volumetric renderer.

In the Score Distillation Sampling (SDS) method, the initial step involves identifying the best parameters θ for the model ϕ. This involves minimizing the loss in relation to a datapoint created by a model. Poole et al. use the formular

$$\theta^* = \arg\,\min_\theta \mathcal{L}_{\text{Diff}}(\phi, \mathbf{x} = g(\theta))$$

This essentially means the model is adjusted to find the parameters θ that bring its output as close as possible to the desired result based on textual input. The primary SDS function by Poole et al. is:

$$\nabla_\theta \mathcal{L}_{\text{SDS}}(\phi, \mathbf{x} = g(\theta)) \triangleq \mathbb{E}_{t,\epsilon}\left[w(t)\left(\hat{\epsilon}_\phi(\mathbf{z}_t; y, t) - \epsilon\right)\frac{\partial \mathbf{x}}{\partial \theta}\right]$$

In this equation, $w(t)$ acts as a weighting factor, modifying the influence of different components within the formula. The term $\hat{\epsilon}_\phi(\mathbf{z}_t; y, t)$ represents the score function predicted by the diffusion model ϕ. This function estimates the noise adjustments needed based on the noisy image z_t, the text embedding y, and the noise level t. The actual noise at time t is denoted by ϵ. This function is key in determining how the noise should be adjusted during the diffusion process to align the generated image with the text input. Additionally, the term $\frac{\partial \mathbf{x}}{\partial \theta}$ indicates how changes in the model's parameters θ affect the image $\mathbf{x}$, guiding the optimization process. SDS adds controlled noise to an image x at each timestep t, and then adjusts it based on the model's scoring, efficiently updating the image to closely match the text descriptions without the need for traditional backpropagation [Poole et al., 2022].

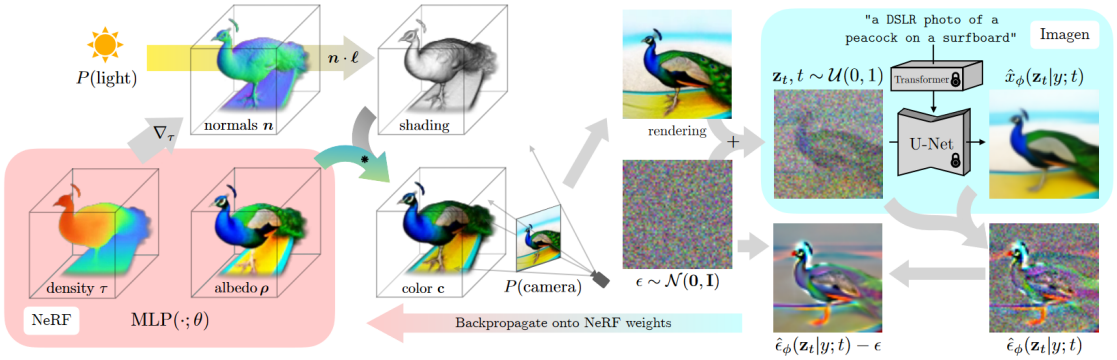

Figure 9: Summatized functionality of Dreamfusion, image by [Poole et al., 2022]

DreamFusion offers an innovative approach to transforming text into 3D models, as depicted in Figure 9. This process employs several key components: natural language captions for directional guidance, Google's Imagen as the text-to-image diffusion model [Saharia et al., 2022], an imporved version of Neural Radiance Field, the mip-NeRF 360 [Barron et al., 2022] "that reduces aliasing" [Poole et al., 2022], and the Score Distillation Sampling (SDS) for the loss function.

At the center of the rendering process is the Neural Radiation Field, which is represented as MLP$(\cdot;\theta)$, where the dot $\cdot$ denotes the input of 3D coordinates and viewing directions. This input is important to determine how light and color interact in 3D space. θ symbolizes the parameters or weights of the MLP that are fine-tuned during training. These parameters determine how the MLP interprets its input (the 3D coordinates and viewing directions) to produce the final output, such as the color and density at each point in the 3D model.

The creation of 3D scenes begins with this parameterization of the NeRF MLP, followed by randomly choosing camera angles and point light positions. This randomness ensures realistic representations from various perspectives [Poole et al., 2022]. Shading plays a crucial role in adding depth and realism, driven by the interaction of light with surface normals, computed from the density gradients and the light position l. Additionally, the inherent color of objects, or albedo, is generated during the NeRF rendering phase. By combining this albedo with the effects of shading, the NeRF accurately renders the final color for each scene point. The result is a detailed visual representation from the selected viewpoint.

After rendering, DreamFusion evaluates the scene against the diffusion model's predictions, assessing diffusion loss to evaluate the match with the expected outcome. The rendered image is then diffused and reconstructed using a static conditional Imagen model, which adds predicted noise $\hat{\epsilon}_\phi(z_t|y;t)$ into the rendering, to improve fidelity but also increasing variance [Poole et al., 2022].

The model refinement phase involves subtracting this predicted noise, resulting in a direction with reduced variance. This direction informs the backpropagation through the rendering process, crucial for updating the NeRF's MLP parameters in a manner that more accurately reflects the scene described by the text.

Despite DreamFusion's promising results, it is not without limitations. The model tends to exhibit a lower level of detail, partly due to its reliance on a 64×64 image model. Furthermore, while SDS is an effective loss function, it sometimes leads to "oversaturated and oversmoothed results [. . .]" [Poole et al., 2022]. Another aspect to consider with SDS is the mode-seeking behavior, which potentially limits the variety of results generated. This limitation is strengthened by the use of KL divergence, "which has been previously noted to have mode-seeking properties in the context of variational inference and probability density distilaltion"[Poole et al., 2022]. This tendency of the model to prioritize the most frequent patterns can lead to a trade-off between accuracy and diversity, so "it may be unclear if minimizing this loss will produce good samples" [Poole et al., 2022]. This statement highlights a major challenge in machine learning, especially with generative models such as DreamFusion. Minimizing loss, a standard method for improving model performance, does not always lead to high-quality or diverse results. There is a risk of overfitting, where the model can reproduce the training data very well, but is less able to generate new and diverse results.

3.1.2 Fantasia3D

The model proposed by Chen et al. takes a different approach to generating 3D models from text input, in particular by disentangling geometry and appearance in the generated 3D models. This method offers a more detailed rendering quality compared to conventional Neural Radiance Fields (NeRFs), which use volume rendering to combine the learning of surface geometry with pixel colors. This conventoional approach limits effective surface recovery, lacking the capability to track the surface of an object and tune detailed material and texture. In contrast, Fantasia3D achieves more realistic outputs with its hybrid scene representation of DMTet, "which maintains a deformable tetrahedral grid and a differentiable mesh extraction layer; deformation can thus be learned through the layer to explicitly control the shape generation" [Chen et al., 2023]

In the geometry stage, Fantasia3D relies on a Deformable Mesh Tetrahedralization (DMTet), which parametrizes the 3D geometry as a Multi-Layer Perceptron (MLP) Ψ. Initially, Fantasia3D renders and encodes surface normals and object masks extracted from DMTet. However, in later stages, the model refines its approach by using only the rendered normal map for shape encoding [Chen et al., 2023]. The default initialization of DMTet is an ellipsoid, but the model also accepts custom inputs.

Appearance Modeling involves training another MLP Γ, which is responsible for applying the Bidirectional Reflectance Distribution Function (BRDF) [Chen et al., 2023] to a pre-learned DMTet. This function is crucial for "predict[ing] parameters of surface material and supports high-quality 3D generation via photorealistic rendering" [Chen et al.,

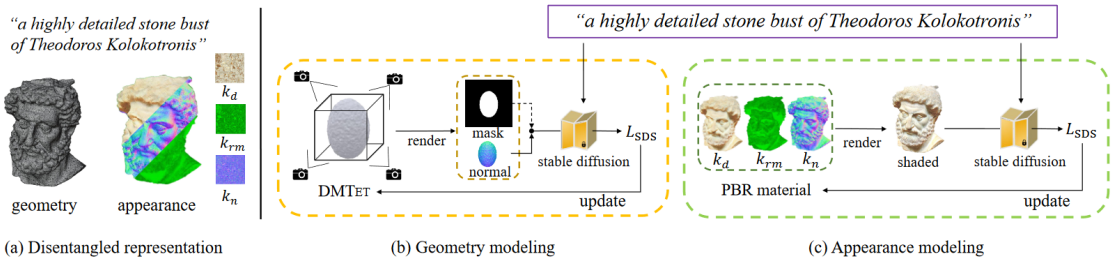

Figure 10: Outline of the method Fantasia3D. Figure taken From [Chen et al., 2023]

2023]. The BRDF focuses on the following parameters to achieve accurate shading of the geometry: the diffuse value k_d, the combined roughness and metallic properties k_{rm}, and the normal variation k_n. These are predicted using the formula $(k_d, k_{rm}, k_n) = \Gamma(\beta(p); \gamma)$ by Chen et al., where $\beta(p)$ represents the surface properties at point p and γ denotes network parameters.

Both MLPs (Γ for appearance and Ψ for geometry) undergo a refinement process, using a pre-trained Stable Diffusion model [Rombach et al., 2021]. This model improves the capabilities of Γ and Ψ, ensuring that they accurately interpret and render 3D shapes and textures. A key aspect of this refinement is the use of Score Distillation Sampling (SDS) loss [Mildenhall et al., 2020] for optimization. In this process, SDS loss functions by comparing the true image with the one generated by Fantasia3D. This comparison is achieved through ray casting, where rays are projected for each pixel in the scene. The model then renders the color of each ray, effectively translating the 3D model into a 2D image. This step allows the model to evaluate and adjust its rendering based on how accurately it replicates the true image of Stable Diffusion. By iterating this process, the model continually improves its accuracy in rendering photorealistic images, ensuring that the final output is as close to the actual image as possible.

Fantasia3D offers a high degree of user interactivity, permitting the incorporation of both custom and predefined generic 3D shapes, thus greatly enhancing the versatility and user engagement in the content creation process. The separation of geometry and appearance generation also ensures compatibility with widely-used graphics engines [Chen et al., 2023]. Despite its capabilities in creating high-quality 3D models from textual descriptions, Fantasia3D encounters specific challenges. One notable limitation is its struggle with accurately generating complex geometries like hair, fur, and grass [Chen et al., 2023]. Furthermore, the model is currently not able to generate complete scenes as focus is currently lying on individual object generation [Chen et al., 2023].

3.1.3 Magic 3D

Shap-E involves training an encoder which "produces a latent representation of a 3D asset" [Jun and Nichol, 2023]. In a second step, a diffusion prior on the obtained latent representations is trained, conditioning it on images or text descriptions to capture additional information [Jun and Nichol, 2023].

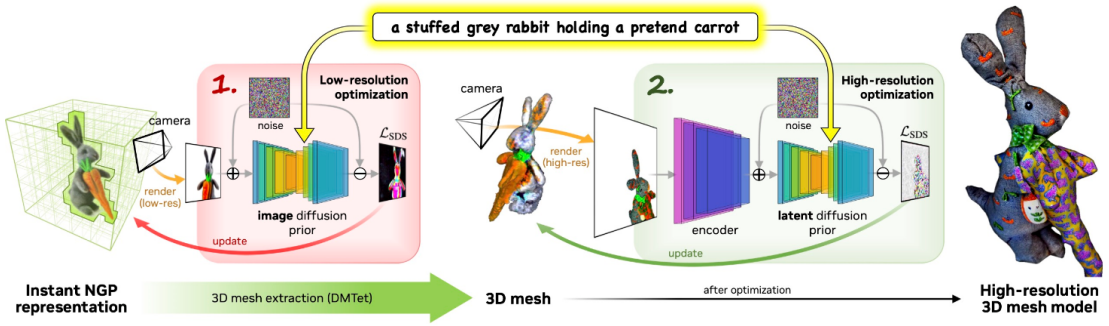

Figure 11: Summatized functionality of Magic3D

3.2 3D from Image

//TODO

3.2.1 Magic 123

//TODO

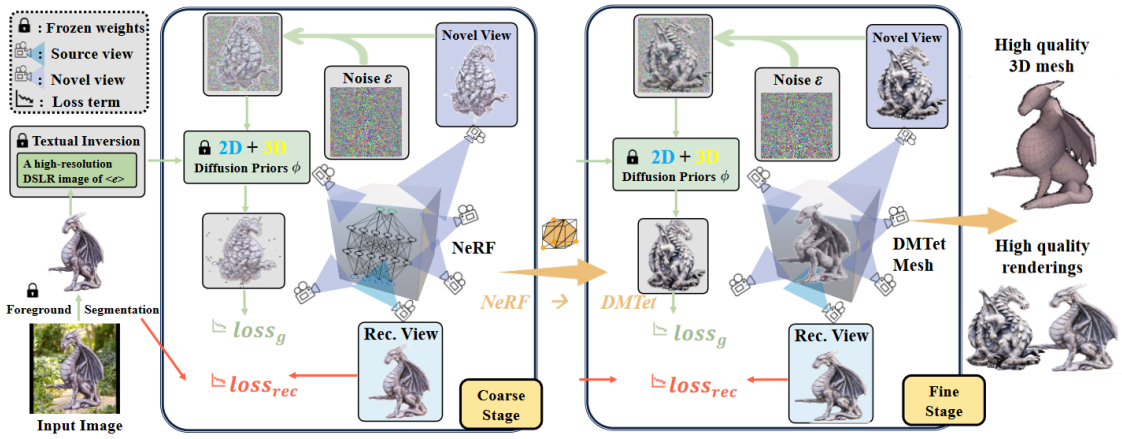

Figure 12: Summatized functionality of Magic123

3.2.2 Wonder 3D

//TODO

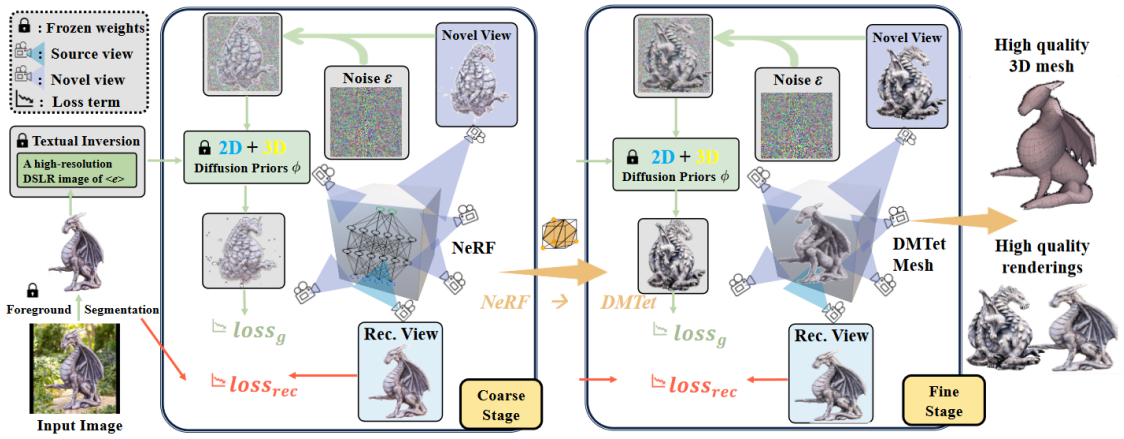

Figure 13: Summatized functionality of Wonder3D

3.3 3D from Video

//TODO

3.3.1 NERfs

//TODO

Chapter 4

Comparative Study

4.0.1 Experimental Setup

details about data collection, preprocessing, and model training

4.0.2 Performance Metrics

discussion of evaluation criteria used in the comparative study

4.0.3 Results and Analysis

present findings and their interpretation

Chapter 5

Future Directions

The exploration of automatic 3D model generation has revealed a dynamic and rapidly evolving field. While this thesis has covered several significant models, it's important to acknowledge that these represent just a fraction of the ongoing advancements. The beauty of this field lies in its cumulative progress, where each new model not only builds upon existing knowledge but also introduces innovative approaches to enhance the state-of-the-art. This chapter will highlight some of the most promising research trends and potential directions for future exploration, focusing on both computational efficiency and quality enhancement.

The realm of 3D model generation is marked by continuous advancements, with new findings emerging on a near-weekly basis. Although keeping pace with these developments is challenging, certain areas within this domain warrant ongoing attention and improvement. This chapter will also speculate on future possibilities, acknowledging the dynamic nature of this field where today's cutting-edge advancements may soon become tomorrow's standard practices.

5.0.1 Emerging Trends in 3D Model Generation

explore current developments and new directions in the field. Gaussian Splatting Lumia AI - Genie

Virtual production is becoming increasingly popular, allowing for more creative and cost-effective backdrops. This rise in demand is expected to continue, driving growth in the market for production-ready 3D assets. The concept of the metaverse is also gaining traction, with significant investments from major companies. This virtual environment is set to redefine experiences ranging from virtual concerts to digital clothing, opening new avenues for 3D modeling applications.

Another emerging trend is the standardization of universal modeling standards. As 3D assets are required to function across various virtual platforms, the need for standardization becomes crucial. Tools like TurboSquid's StemCell are paving the way for more versatile and universally compatible 3D models.

Overall, these trends indicate a shift towards more integrated, AI-driven, and standardized approaches in 3D model generation, pointing towards a future where 3D modeling is more accessible, versatile, and deeply ingrained in various aspects of digital interaction.

5.0.2 Potential Research Directions

suggest areas for future research based on study's findings. Reduce Computational cost

The future of 3D modeling and additive manufacturing (AM) holds substantial potential for innovation across various sectors. One key area of research is exploring the use of AI in additive manufacturing. With technologies like ChatGPT and Nvidia's AI tools demonstrating the ability to create 3D models from text input, there's a significant opportunity for AI applications to become more prevalent in additive manufacturing.

The healthcare industry is increasingly adopting AM to deliver personalized healthcare solutions. This includes the creation of patient-specific implants and surgical tools customized to individual anatomies. The integration of AM in healthcare facilities for various applications, such as orthopedics, dental, and surgical instrumentation, offers an exciting avenue for research, potentially revolutionizing patient care and treatment.

In industrial markets, the use of metal AM for the production of end-use parts, especially in the aerospace and energy sectors, is experiencing rapid growth. Researching new AM solutions specifically designed for mass production, which combine various printing and finishing technologies, could significantly enhance manufacturing workflows and output

Bioprinting is another promising field, with significant strides being made in using bioprinted human tissue models for drug discovery and development. Researching the ability to simulate human responses to drugs in a laboratory setting using bioprinted models could dramatically streamline drug development processes and potentially eliminate the need for animal testing.

Moreover, there are opportunities to address challenges such as intellectual property protection, managing the complexity of large-scale 3D models, and overcoming adoption barriers due to advanced 3D modeling tools and techniques. These challenges present potential research directions that could significantly impact the field's growth and adoption.

Chapter 6

Conclusion

//TODO

6.0.1 Summary of Findings

provide a concise overview of the main results

6.0.2 Contributions to the Field

highlight the significance of the thesis

6.0.3 Implications and Practical Applications

discusses real-world impact of the findings.

Bibliography

Aggarwal, A., Mittal, M., and Battineni, G. (2021). Generative adversarial network: An overview of theory and applications. *International Journal of Information Management Data Insights*, 1(1):100004.

Anderson, B. D. (1982). Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326.

Arandjelović, R. and Zisserman, A. (2021). Nerf in detail: Learning to sample for view synthesis.

Barron, J. T., Mildenhall, B., Verbin, D., Srinivasan, P. P., and Hedman, P. (2022). Mip-nerf 360: Unbounded anti-aliased neural radiance fields.

Brophy, E., Wang, Z., She, Q., and Ward, T. (2023). Generative adversarial networks in time series: A systematic literature review. *ACM Computing Surveys*, 55(10):1–31.

Chen, R., Chen, Y., Jiao, N., and Jia, K. (2023). Fantasia3d: Disentangling geometry and appearance for high-quality text-to-3d content creation.

Croitoru, F.-A., Hondru, V., Ionescu, R. T., and Shah, M. (2023). Diffusion models in vision: A survey.

Doersch, C. (2016). Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. (2021). An image is worth 16x16 words: Transformers for image recognition at scale.

Gerats, B. G. A., Wolterink, J. M., and Broeders, I. A. M. J. (2023). Dynamic depth-supervised nerf for multi-view rgb-d operating room images.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2020). Generative adversarial networks. *Communications of the ACM*, 63(11):139–144.

Goodfellow, I. J., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press, Cambridge, MA, USA. `http://www.deeplearningbook.org`.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., and Lerchner, A. (2017). beta-VAE: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*.

Hinton, G. E. and Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507.

Ho, J., Jain, A., and Abbeel, P. (2020). Denoising diffusion probabilistic models. *ArXiv*, abs/2006.11239.

Hu, S., Zhou, K., Li, K., Yu, L., Hong, L., Hu, T., Li, Z., Lee, G. H., and Liu, Z. (2023). Consistentnerf: Enhancing neural radiance fields with 3d consistency for sparse view synthesis.

Hyvärinen, A. and Dayan, P. (2005). Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(4).

Jun, H. and Nichol, A. (2023). Shap-e: Generating conditional 3d implicit functions.

Kingma, D. P., Salimans, T., Poole, B., and Ho, J. (2023). Variational diffusion models.

Kingma, D. P. and Welling, M. (2022). Auto-encoding variational bayes.

Liu, Q., Lee, J., and Jordan, M. (2016). A kernelized stein discrepancy for goodness-of-fit tests. In Balcan, M. F. and Weinberger, K. Q., editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 276–284, New York, New York, USA. PMLR.

Martínez, G., Watson, L., Reviriego, P., Hernández, J. A., Juarez, M., and Sarkar, R. (2023). Towards understanding the interplay of generative artificial intelligence and the internet.

Michalkiewicz, M., Pontes, J. K., Jack, D., Baktashmotlagh, M., and Eriksson, A. (2019). Deep level sets: Implicit surface representations for 3d shape inference.

Michelucci, U. (2022). An introduction to autoencoders.

Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., and Ng, R. (2020). Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*.

Mordvintsev, A., Pezzotti, N., Schubert, L., and Olah, C. (2018). Differentiable image parameterizations. *Distill*, 3.

Murtagh, F. (1991). Multilayer perceptrons for classification and regression. *Neurocomputing*, 2(5):183–197.

Noriega, L. (2005). Multilayer perceptron tutorial. *School of Computing. Staffordshire University*, 4(5):444.

Poole, B., Jain, A., Barron, J. T., and Mildenhall, B. (2022). Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*.

Rabby, A. S. A. and Zhang, C. (2023). Beyondpixels: A comprehensive review of the evolution of neural radiance fields.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., and Sutskever, I. (2021). Learning transferable visual models from natural language supervision.

Rezende, D. J., Mohamed, S., and Wierstra, D. (2014). Stochastic backpropagation and approximate inference in deep generative models.

Roberts, G. O. and Tweedie, R. L. (1996). Exponential convergence of langevin distributions and their discrete approximations. *Bernoulli*, pages 341–363.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. (2021). High-resolution image synthesis with latent diffusion models.

Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E., Ghasemipour, S. K. S., Ayan, B. K., Mahdavi, S. S., Lopes, R. G., Salimans, T., Ho, J., Fleet, D. J., and Norouzi, M. (2022). Photorealistic text-to-image diffusion models with deep language understanding.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., Chen, X., and Chen, X. (2016). Improved techniques for training gans. In Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

Shen, T., Gao, J., Yin, K., Liu, M.-Y., and Fidler, S. (2021). Deep marching tetrahedra: a hybrid representation for high-resolution 3d shape synthesis.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. (2015). Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR.

Song, Y. (2021). Score-based generative modeling through stochastic differential equations. `https://yang-song.net/blog/2021/score/`. Accessed: [26.11.2023].

Song, Y., Durkan, C., Murray, I., and Ermon, S. (2021). Maximum likelihood training of score-based diffusion models. *Advances in Neural Information Processing Systems*, 34:1415–1428.

Song, Y. and Ermon, S. (2019). Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. (2020). Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2023). Attention is all you need.

Wei, Y., Liu, S., Rao, Y., Zhao, W., Lu, J., and Zhou, J. (2021). Nerfingmvs: Guided optimization of neural radiance fields for indoor multi-view stereo.

Xiao, Z., Kreis, K., and Vahdat, A. (2022). Tackling the generative learning trilemma with denoising diffusion gans.

Yang, L., Zhang, Z., Song, Y., Hong, S., Xu, R., Zhao, Y., Shao, Y., Zhang, W., Cui, B., and Yang, M.-H. (2022). Diffusion models: A comprehensive survey of methods and applications. *arXiv preprint arXiv:2209.00796*.

Yariv, L., Kasten, Y., Moran, D., Galun, M., Atzmon, M., Basri, R., and Lipman, Y. (2020). Multiview neural surface reconstruction by disentangling geometry and appearance.

Erklärung

[TODO: Fügen Sie hier die Erklärung zur selbstständigen Bearbeitung gemäß Ihrer Themabestätigung ein]

Datum	Unterschrift