Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Computer Vision and Learning Group

3D Consistent Implicit Generative Models of Human Appearance

Master's Thesis

Maxime Raafat
Department of Mathematics

Advisors:	Dr. Sergey Prokudin
Supervisor:	Prof. Dr. Siyu Tang

September 26, 2022

Abstract

We present a lightweight generative model for multi-view consistent full-body textured human avatars. Combining an explicit and expressive point-based module with a novel yet simple 3D aware GAN-based architecture, we develop a framework for capturing a diverse statistical distribution of clothed human appearance. While high-quality 3D generative modelling has recently achieved outstanding results, the extension on full-body humans has been subject to slow progress with only few attends at tackling the challenge. The method we propose offers several advantages over current rival techniques. First, the expressiveness of our point-based module guarantees robustness to novel body poses and camera views. Our generator is then trained on a large dataset of single-view photographs only, and synthesizes by design 3D consistent appearance textures in an implicit UV state without enforcing any view or identity preservation constraints. Finally, our pipeline enables rendering at impressive speeds on consumer hardware, enabling fast visualization and multiple real-time applications further down the line.

Acknowledgements

With my beginnings at the VLG in September 2021, I slowly discovered a team of passionate researchers and doctoral students, into which I had the chance of being integrated over the following year. Already months before my start, I was given the great opportunity to choose the direction of my semester project with my eventual supervisor, *Sergey Prokudin*. From then on until the end of my Master's Thesis, he provided essential advise, inspiration and expertise, which guided me all along the journey. I truly thank him for his valuable time, the effort he invested into me, and the friendship we built.

Throughout the year, I later on became acquainted with my professor, *Siyu Tang*, on a more personal level. Siyu has wonderfully taught two courses I followed with tremendous interest, Machine Perception and Virtual Humans; having her as a professor has been the best experience a student can get. I genuinely suggest any student to visit her lectures. I deeply appreciate the enthusiasm she has shown all along my Thesis, as well as encouragement and support I received from her within the last few months.

Alongside Siyu, I additionally got to know *Julien Valentin* - principal scientist at Microsoft Zurich - as an industry partner working with Sergey and Siyu. Julien and Siyu have both devoted much appreciated time and energy into weekly meetings, offering many helpful suggestions.

I would also like to express my gratitude towards any VLG members, who all showed amazing support and always treated me as an equal. Finally I want to thank my family, friends and partner, who listened for countless hours to the technical adventure I shared with them, even though presumably not understanding much of it.

To any person I mentioned and that helped me for this fabulous project, thank you.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Photorealistically rendering human avatars is an essential component of rapidly growing immersive technologies. Persuasive virtual and augmented reality, teleconferencing, virtual try-ons or animatable characters are just a small subset of all possible applications. Populating digital worlds will require novel and deformable clothed people, going beyond the well-studied reconstruction problem [4, 99]. We aim at capturing a diverse and representative distribution of high-quality human appearance via deep neural networks, enabling multi-view consistent realistic avatars with easy controllability.

While constrained environments of capture studios enable a considerable degree of realism using photogrammetry principles, the captured data comes at a high cost and is not scalable, therefore not suited for our generative purpose. Instead, we find inspiration in the recent 3D generative modelling literature and learn exclusively from single-view 2D photographs. Although traditional 2D and modern 3D synthesis [46, 17] have reached an unmatched image quality, combining them with human modelling unlocks new technical difficulties. Humans have many degrees of freedom in pose and identity, which a characteristic distribution should be able to incorporate. Only few works before ours have attempted solving this challenge [31, 13, 107], but none of them escaped the uncanny valley of realism. Their key idea lies in merging classical implicit human reconstruction methods with powerful 3D generators. On the other hand we take a more robust approach by leveraging recent tools and discoveries in explicit point-based neural rendering [100, 54].

In a first step, we introduce a new appearance model which extends on a family of skinned multi-person linear models [62, 78] with a point-based module. The point-based module represents appearance details as a UV texture map containing RGB colors of a point and its displacement along the normal. Second, we modify a powerful 2D lightweight GAN-based generator [57] to discriminate on rendered images rather than on the generator output right away. Precisely, we generate UV textures which we wrap around a posed body prior and then render via differentiable rendering. The rendered images then serve as fake labels, while our true labels are drawn from a large dataset of full-body human images [25]. In short, our framework approximates a distribution of UV appearance textures without ever explicitly seeing texture data. Combined, the point-based module and the lightweight architecture are capable of achieving high-quality 3D multi-view consistent results, but at a fraction of the time and computational cost of modern 3D generators.

Our contributions can be summarized as follows:

1. A fast, explicit and expressive point-based module for human appearance modelling;
2. A lightweight multi-view consistent GAN-based generator operating in an implicit UV state;

3. A rigorous evaluation of our method in the context of novel human appearance synthesis.

The manuscript is organised as follows. Chapter 2 covers the most relevant progress in the field of human modelling and other related work. Chapter 3 provides the fundamentals for the understanding of our method, reviewing parametric human body models, neural rendering and generative modelling. Our method is rigorously described in Chapter 4, with a diverse set of failure and successful experiments conducted in Chapter 5. The results of our experiments are more thoroughly discussed in Chapter 6, alongside a quick discussion on the limitations and implications of our work.

Chapter 2

Related Work

2.1 Full-body human appearance modelling

Accurately representing full-body human avatars has been a long standing problem. Our method is inspired from deformable 3D models [14, 7, 44] and particularly leverages SMPL-X [78] from the SMPL [62, 87] family. Several previous works build on top of SMPL to capture clothing deformations [65, 64], but omit the modelling of color appearance indispensable for photorealistic rendering. Earlier optimization-based methods [4, 3] focus on reconstructing detailed 3D textured body models of arbitrary people from a single monocular video using photogrammetry principles. Although yielding controllable avatars with detailed color appearance texture, these models are biased towards tight clothing and hair because of the constrained underlying mesh representation. Other works in the same direction have investigated image-to-image translation-based regression techniques [5, 2, 6, 55], considerably reducing the inference time. Those methods however oftentimes produce poor results and do not scale well as they are trained on limited data. Despite their advantages and potential ability to hallucinate occluded or unseen body regions, many of the existing regression-based methods typically require high resolution paired inputs of 3D scans [85, 22, 97, 96] for training.

In recent years, implicit representations have gained a lot of attention in human surface and appearance modelling [88, 89, 108, 42, 37]. Implicit volumetric rendering approaches [68] in particular have seen a rapidly growing interest because of their simplicity, expressiveness and the breakthroughs made by the community [58, 70, 28, 106, 105, 98, 9, 10, 67, 21, 69]. Many multi-view setups for human appearance rendering make use of neural radiance fields (NeRFs) as a backbone [61, 80, 75], and more advanced settings additionally incorporate a canonical pose-independent space with deformable human models [53, 79, 104, 59, 99, 43, 27, 19]. While resulting in impressive image quality, these methods regularly suffer from slower rendering speed and do not generalize well to extreme camera views or novel human poses when only few images or camera angles are available. In contrast, we adopt a different approach by combining point primitives [32] with recent techniques from differentiable rendering [100, 54, 1, 51, 20, 52]. Our method therefore enables efficient rendering and is robust to novel views and pose deformations of the underlying SMPL-X mesh.

2.2 3D generative modelling

3D models for high-quality multi-view consistent image synthesis have recently emerged, building on top of traditional 2D image generators. Generative adversarial networks (GANs) [30] especially have led the way in terms of 2D photorealistic image generation [83, 45, 47, 48, 46, 15, 110], offering for instance desirable qualities such as disentanglement of the so called StyleSpace [41, 47, 48, 46, 103]. Extending traditional 2D CNN-based generators to 3D settings enables the synthesis of voxel-based data [101, 26, 111, 38, 71, 72], but at high memory and computing costs with low output resolution. With the growth of implicit representations as described in Section 2.1, a few methods have tackled 3D generation based upon neural radiance fields [90, 74, 18], but suffer from slow forward passes which limit the training quality and outputs. Other works cleverly leverage NeRFs to yield impressive 3D aware image quality [33, 109] or even translate single images to full NeRFs [16]. Chan et al. [17] introduce a simple hybrid tri-plane representation taking advantage of both explicit and implicit methods, speeding up the training time and yielding state-of-the-art results.

Other approaches substitute NeRFs for signed distance fields (SDFs) [77], or replace GANs for recently emerging Diffusion Models [11, 40, 73, 23, 86]. Our method shares ideas with the Texturify framework by Siddiqui et al. [91], for instance by learning a distribution of texture maps. Unlike their work, we however do not reparameterize the input geometry and operate on deformable point clouds rather than static meshes.

2.3 Generators for full-body human appearance

Although essential to many virtual applications, synthesizing high-quality realistic full-body avatars is an unsolved challenge. Combining the knowledge from the two previous sections, we provide an overview on the sparse set of works which tackle generative modelling for full-body human appearance.

Grigorev et al. [31] pioneered virtual human synthesis by proposing StylePeople, a neural dressing module combined with a generative network inspired by StyleGAN [48]. The neural dressing module utilizes deformable SMPL-X meshes warped with a learned neural texture of deep features, and relies on a deferred neural rendering pipeline [94]. Although yielding controllable and view consistent avatars, the learned rasterizer is subject to inaccuracies, and combining it with low-quality data is the main reason for the mediocre generated results. We extend some of the ideas from StylePeople but enforce a more robust representation with a proper point rasterizer.

More recent research exploit state-of-the-art 3D GANs combined with deformable models. Generative neural articulated radiance fields [13] and AvatarGen [107] both merge a canonical pose-independent space with the tri-plane representation introduced in EG3D [17]. AvatarGen additionally substitutes radiance fields for SDFs for more realistic results. None of those methods however is capable of high-quality photorealistic generation. We believe this is not the correct approach, since already the projection step into and out of the canonical space is prone to imperfections or flawed learning. Instead our pipeline generates textures right onto the posed geometry and is thus by design consistent under novel poses and views.

Finally, Fu et al. [25] take a more data-centric approach and train a vanilla StyleGAN generator on a large-scale human image dataset. Despite not 3D aware, the generated results are highly photorealistic and significantly outperform the quality of all above-mentioned methods. Furthermore, the provided data that the authors make publicly available is a significant contribution to the community. An earlier work on generative interpretable faces [29] is also worth mentioning, in which a 2D generator is conditioned on controllable parameters to encourage disententglement between 3D relevant attributes. Extending [25] with conditioning on human model parameters is a conceivable suggestion to incorporate 3D knowledge into the generation process. To conclude this chapter, we refer the reader to Table 2.1 which presents a unified overview of the works we mentioned and consider the most important in their respective field.

Table 2.1: Unified overview of the presented related works. The photorealism column provides a qualitative and subjective assessment of each work in comparison to the others. We kindly ask the reader to take our assessment with a grain of salt.

METHOD	GENERATIVE	PHOTOREALISM	3D CONSISTENCY	CONTROLLABLE
PHORHUM [6]	No	Middle	Yes	Yes
HumanNeRF [99]	No	Middle	Yes	Yes
EG3D [17]	Yes	High	Yes	No
StylePeople [31]	Yes	Low	Yes	Yes
AvatarGen [107]	Yes	Low	Yes	Yes
StyleGAN-Human [25]	Yes	High	No	No

Chapter 3

Setting

This chapter covers the required material and concepts necessary for the understanding of this manuscript. Specific models or architecture technicalities will not be discussed unless relevant to the topic, in which case the particular technique will be thoroughly described in the according section. For further details on state-of-the-art methods, please refer to the related work in Chapter 2.

3.1 Parametric human modelling

Modelling realistic full-body digital humans is a complex task involving several challenges. Diversity in human shape, pose and appearance lead to near infinite possible combinations. Accurately representing each aspect in a single human is likely prone to imperfections in topology and geometry, and can suffer from inconsistencies between different subjects. Many models construct and incorporate a parametric body prior to capture the statistical variation within a human population, and minimize the inconsistencies between reconstructions. Below we provide a brief summary of the popular and widely accepted *SMPL* [62] parametric human model.

SMPL is a realistic 3D model of minimally clothed human bodies learned from thousands of 3D body scans, based on linear blend skinning (LBS) and blend shapes. Despite its rather simple pipeline (hence the name), it is capable of representing a large variety of surface level human topology by capturing a few parameters per subject only. SMPL renders body vertices $V \in \mathbb{R}^{3N}$ as a function of identity-dependent and pose-dependent mesh deformations, driven by two corresponding compact sets of parameters, shape $\boldsymbol{\beta} \in \mathbb{R}^{|\boldsymbol{\beta}|}$ and pose $\boldsymbol{\theta} \in \mathbb{R}^{|\boldsymbol{\theta}|}$:

$$M(\boldsymbol{\beta}, \boldsymbol{\theta}) : \mathbb{R}^{|\boldsymbol{\beta}| \times |\boldsymbol{\theta}|} \rightarrow \mathbb{R}^{3N}, \tag{3.1}$$

$$V = W(T_P(\boldsymbol{\beta}, \boldsymbol{\theta}), J(\boldsymbol{\beta}), \mathcal{W}, \boldsymbol{\theta}), \tag{3.2}$$

$$T_P(\boldsymbol{\beta}, \boldsymbol{\theta}) = \bar{\boldsymbol{T}} + B_S(\boldsymbol{\beta}, \mathcal{S}) + B_P(\boldsymbol{\theta}, \mathcal{P}), \tag{3.3}$$

where $T_P(\boldsymbol{\beta}, \boldsymbol{\theta})$ captures shape and pose deformation of the template mesh in the canonical pose $\bar{\boldsymbol{T}}$ via linear transformations B_S and B_P of the shape, respectively pose, blend shapes $\mathcal{S}$ and $\mathcal{P}$. The LBS function W takes as input the T-posed input template T_P and a set of shape-dependent K body joint locations $J(\boldsymbol{\beta}) \in \mathbb{R}^{3K}$, and deforms the template mesh with the blend weights $\mathcal{W}$ along the kinematic tree according to the pose $\boldsymbol{\theta}$ and shape $\boldsymbol{\beta}$.

To better understand the effect of the introduced deformations, let us consider a body model without LBS and blend shapes. Posing our template mesh consists in transforming each vertex according to the pose parameter $\boldsymbol{\theta}$. In our current simplified scenario, vertices are fully independent of one another and the deformed template will likely have sharp and unnatural articulations. LBS assigns vertices to a linear combination of joints, solving the sharp skinning artefacts problem by blending neighboring vertices. Although a posed mesh will now have smooth articulations, LBS cannot resolve unrealistic skin squeezing and stretching around joint locations. Pose blend shapes tackle this issue by displacing each vertex accordingly in canonical pose. Finally shape blend shapes, in analogy to pose blend shapes, displace vertices to model shape identity variation rather pose variation. Figure 3.1 provides an overview of the SMPL pipeline; note that the deformations are not applied in the same order as discussed here. For further details on the model, we refer the reader to the original publication [62].

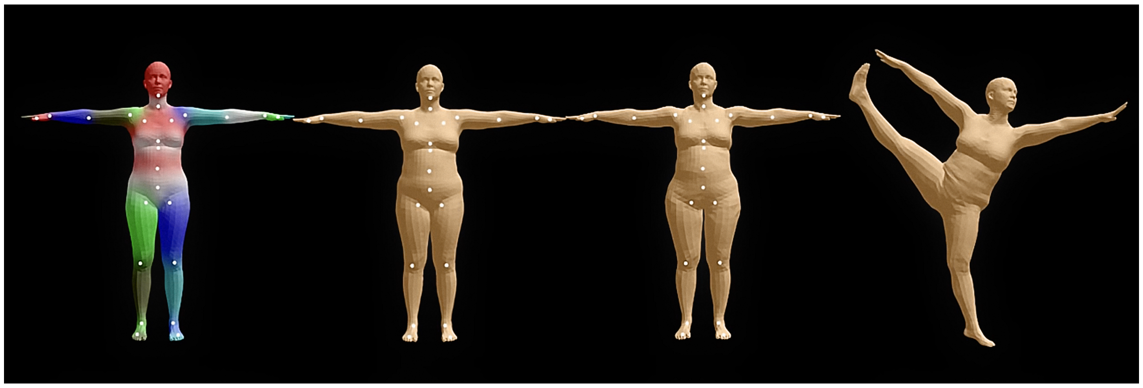


Figure 3.1: SMPL model pipeline (figure from the original publication [62]). From left (a) to right (d): template mesh with LBS shown via color coding (a), skinned template mesh displaced with shape blend shapes (b), shaped skinned template mesh displaced with pose blend shapes (c), final posed mesh with LBS and blend shapes (d). Joint locations are represented with white dots.

While we purposefully mentioned making use of SMPL in our method, we technically employ the extended model *SMPL-X* [78]. SMPL-X additionally introduces fully articulated hands and an expressive face via a similar learning pipeline and model parametrization.

3.2 Neural rendering

Classical Rendering. Rendering is an essential part of modern computer vision and graphics. It consists in producing an image, referred as the render, given the description of a 2D or 3D model. Classical 3D rendering methods can typically be divided into two major families of algorithms: rasterization and physically-based path tracing. The former trades accuracy for speed, while the latter yields highly realistic images but at a high computational cost. Rendering is at the center of many modern perception algorithms, but understanding its exact mechanisms is beyond the scope of this manuscript. For a thorough investigation on the topic, we recommend the book by Pharr et al. [81]. This section discusses recent rendering approaches in neural and differentiable rendering.

The rendering pipeline boils down to the following function r:

$$r : X \times \theta \times C \rightarrow I, \tag{3.4}$$

which maps graphics primitives X (meshes, points, voxels or implicit representations), rendering parameters θ (to not be confused with the pose parameter in SMPL) and camera parameters C to an image I.

Rendering parameters are conventionally editable in a rendering engine, but can also be incorporated into learnable weights of a neural network - more to this in the next paragraph. Camera parameters project the 3D scene into the image domain by first considering the camera's localization (position and rotation encoded into the extrinsics), then mapping 3D world coordinates onto a 2D image plane according to the camera's internal parameters (optical center and focal length encoded into the intrinsics). Rendering is commonly done with perspective projection in a physics-inspired pinhole camera model. Other projection methods such as orthographic projection do not necessarily require camera intrinsics. In orthographic projection, the depth of a 3D world point (distance along the z-axis of the point to the camera) does not affect its location in the projected image.

Differentiable Rendering. Differentiable rendering back-propagates gradients from an image through the rendering engine or neural network, all the way back to the rendering function inputs. This is of interest when modelling, capturing or synthesizing novel renders from image data. Differentiable rendering aims at minimizing the following objective by optimizing some or all of the parameters:

$$X^*, \theta^*, C^* = \underset{X,\theta,C}{arg\ min} \sum_{i=1}^{N} \mathcal{L}\big(r(X, \theta, C_i), I_i^{gt}\big), \tag{3.5}$$

where $\mathcal{L}$ is an image-based loss function and I_i^{gt} is the i^{th} ground truth image corresponding to camera C_i, among N available views per scene. The parameters can then simply be learned via traditional gradient-based optimization methods. Neural rendering, oftentimes mistakenly used interchangeably with differentiable rendering itself, assumes an additional neural shading module. In short, some component of the rendering pipeline is operated by neural networks to produce the final image.

Rendering however, depending on the considered graphics primitive, is usually not fully differentiable. Traditional mesh rendering in particular involves rasterization, a discrete sampling operation which prevents gradients to flow from pixels to mesh vertices. In the next paragraphs we explore the ideas behind differentiable mesh- and point-based rendering. Although not suffering from differentiability issues, we quickly examine volume-based methods for the sake of completeness, regardless.

Rendering Primitives. A mesh comprises a set of vertices (point locations) and faces, which stores the connectivity between vertices (typically three vertices forming triangles). Optionally, texture information can either be stored per vertex, or as an advanced texturing method such as UV texture maps. We previously mentioned the non differentiability of discrete mesh rasterization. Soft rasterization, abbreviated as *SoftRas* [60] (see Figure 3.2 extracted from the paper), overcomes the discretization obstacle by smoothly, or "softly", spreading out the coverage of each triangle face. Precisely, the influence of a triangle onto a pixel is positively correlated with the signed distance from the face to the pixel in question:

$$D(f_i, I_{uv}) = sigmoid\big(\ \delta(f_i, I_{uv}) \cdot \frac{d^2(f_i, I_{uv})}{\sigma}\ \big), \tag{3.6}$$

where $D(f_i, I_{uv})$ represents the influence of face f_i onto the pixel at location (u, v), the sign function $\delta(f_i, I_{uv})$ is $+1$ if $(u, v) \in f_i$, -1 otherwise, and σ corresponds to the face sharpness. The bigger σ, the larger the area face f_i covers. The influence of multiple faces on a pixel will finally be normalized after having been aggregated with respect to the face's depth.

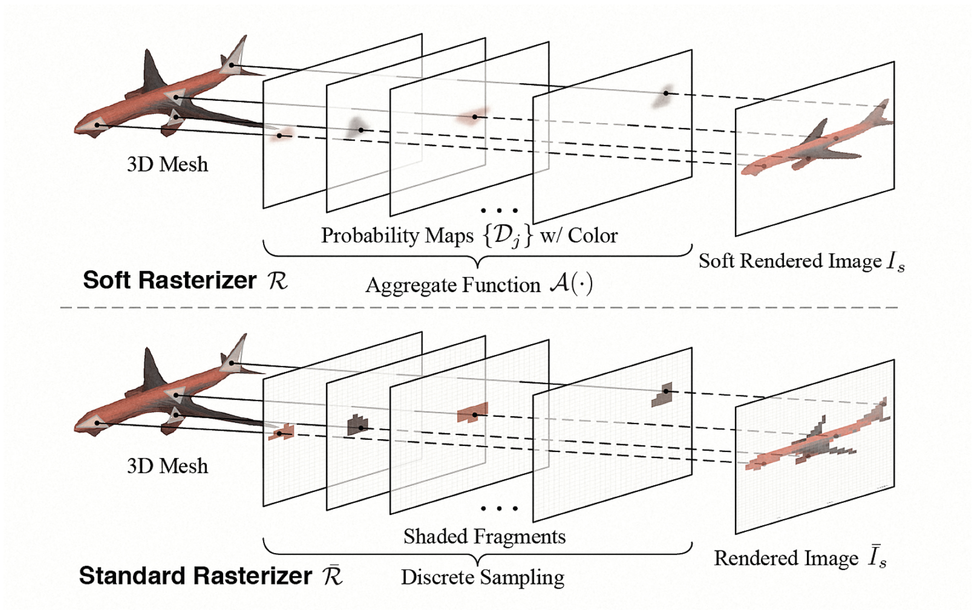

3D Mesh
Probability Maps $\{\mathcal{D}_j\}$ w/ Color
Soft Rendered Image I_s
Soft Rasterizer $\mathcal{R}$
Aggregate Function $\mathcal{A}(\cdot)$
3D Mesh
Shaded Fragments
Rendered Image $\bar{I}_s$
Standard Rasterizer $\bar{\mathcal{R}}$
Discrete Sampling

Figure 3.2: Soft rasterizer $\mathcal{R}$ (top) fusing per-triangle contributions $\mathcal{D}_j$ in a "soft" probabilistic manner, against non-differentiable standard rasterizer $\bar{\mathcal{R}}$ (bottom), which cannot flow gradients from pixels to geometry. Figure borrowed from the original publication [60].

Oftentimes referred to as *point clouds*, point primitives are a simplified version of meshes, in that they drop the need for faces. They accordingly only consist of point locations and store features, such as color or potentially neural features, in each point. Introduced by Wiles et al. [100] and later improved and accelerated by Lassner et al. [54], differentiable point rendering follows the same mechanics as SoftRas; it assigns the influence of a point on a pixel based on their distance to each other, and aggregates the influence of multiple points into the final pixel value.

In contrast to mesh and point rendering, volume-based rendering with voxels (volume pixels) is differentiable by construct. The method requires aggregating features from voxels hit by a ray marching algorithm, based on the voxels' visibility and density. None of those steps involve discrete operations, therefore yielding well-behaving gradients. Despite their expressiveness and ability to render semi-transparent surfaces, voxel primitives are memory hungry and scale with $O(N^3)$ in space complexity, with a volume side length of N features. They additionally often fail in compactly representing geometry, since most voxel cells tend to be empty. A recent and renowned solution [68] is to use implicit representations to approximate the geometry volume as a continuous function, commonly a simple MLP.

Our method presented in future chapters combines point rendering with generative adversarial networks to achieve novel 3D aware image synthesis. The next section concludes the current chapter by discussing the basics of generative modelling, while particularly focusing on GANs.

3.3 Generative modelling

Generative Adversarial Networks. Generative modelling aims at synthesizing new samples drawn from an underlying training distribution, given enough training data. For instance, an ideal generative model trained on a large diverse set of cat images will be capable of creating an unseen novel cat image, not part of the training data. Although the field of generative modelling has substantially grown over the last decade, we exclusively review GANs in the context of image generation due to their convincing capabilities, and only briefly mention alternatives.

Generative Adversarial Networks [30], or *GANs* for short, are a class of highly expressive likelihood-free algorithms for data synthesis. Similar to many generative models, they rely on a specialised network to transform random draws from a simple distribution - oftentimes normal - into realistic images. However, unlike most other classes of models, GANs do not aim to maximize the likelihood of the observed data with respect to the aforementioned specialised network. Instead they introduce a classifier whose task is to distinguish real images from fake generated ones. In the context of GANs, the specialized transformation network and the classifier are referred to as the generator G and the discriminator D. Formally, G maps any input from a latent space Z to the image space X by ideally following the data distribution. On the other side, D projects images to a real value between zero and one defined as the probability of being a genuine image sample:

$$G : Z \rightarrow X, \quad D : X \rightarrow [0, 1]. \tag{3.7}$$

Adversarial Objective. Training GANs boils down to finding a Nash equilibrium in a two-player game. In game theory, a Nash equilibrium represents a state in which the two players of a game do not benefit from changing their own strategy, knowing the opponent's strategy. In the current setting, this means converging to a state where the generator does not improve anymore, and the discriminator is not capable of telling the difference between true and fake labels. The original formulation [30] consecutively optimizes the generator and discriminator for opposite objectives which can be summarised as $arg\ \underset{G}{min}\ \underset{D}{max}\ V(G, D)$, with the loss:

$$V(G, D) = \mathbb{E}_{x \sim p_d(x)}\big[log\ D(x)\big] + \mathbb{E}_{z \sim p_z(z)}\big[log(1 - D(G(z)))\big], \tag{3.8}$$

where x and z are drawn from the true and generated distributions p_d, respectively p_z. Let us quickly inspect $V(G, D)$ by considering the possible outcomes for G and D. When training G, we hope to fool D into believing that the synthesized images are real. We want $D(G(z))$ to be one, i.e., the second term approaches negative infinity and minimizes the objective as expected. D on the other hand encourages correctly labeled true and fake samples, such that $D(x) = 1$ and $D(G(z)) = 0$. This results in the function's global maximum with both terms being zero, and concludes our sanity check to confirm the adequacy of the loss function. Given a fixed G, the objective's equilibrium is achieved for the optimal discriminator D^*:

$$D^*_G(x) = \frac{p_d}{p_d + p_z}. \tag{3.9}$$

Using D^* and a few calculus fundamentals, the minimax game can be reformulated into:

$$V(G, D^*) = -2 \cdot \bigg(log\, 2 + \cdot JS\big(p_d || p_z\big)\bigg), \tag{3.10}$$

where JS is the Jensen-Shannon (JS) divergence, a symmetric similarity measure for probabilistic distributions. The theoretical global minimum is achieved in $-log\ 4$ when $p_d = p_z$, i.e., when D reaches 0.5 everywhere. In a nutshell, the generator's performance will be evaluated by the discriminator and accordingly adapt its generative ability depending on D's decision until convergence. In this case, D cannot tell true from fake images apart anymore.

GAN Training. While a theoretical unique solution is guaranteed, the lack of theory and learning algorithms compared to explicit models makes GANs challenging to train. Training is fragile and suffers from many instabilities; this section discusses some popular well-known engineering tricks and tuning techniques which paved the way for photorealistic image synthesis. GANs are notoriously famous for two commonly encountered issues: a discriminator learning too fast and mode collapse.

At the beginning of training, D is generally capable of learning very quickly, while providing only small gradients to G. The discriminator consequently ends up perfectly labeling each sample, converging quickly to zero, and the generator produces meaningless information. Mode collapse on the other hand is an undesired saddle point in the dual energy landscape; making downhill progress for one player may move the other player uphill. The generator produces sub-optimal images, but updating it would hurt the discriminator. The result is an undesired convergence state from which both G and D cannot escape.

The above-described problems often occur in practice as a consequence of the GAN objective itself; many variants have been suggested [66, 76, 8, 35, 50, 102, 12] to mitigate the encountered instabilities. Wasserstein GANs with gradient penalty (WGAN-GP) [35] is one notable variant, in which the JS divergence has been replaced for the Wasserstein distance. Intuitively, the new metric measures the minimum effort required to transform one distribution into another one, and was introduced because of its many advantages over the JS divergence. The JS divergence typically correlates badly with sample quality and saturates as the discriminator gets better. This leads to vanishing gradients in the generator and potentially to mode collapse. The Wasserstein distance, from this perspective, is more stable since it proportionally grows or shrinks, when the distributions get further away, respectively closer to each other. The additional gradient penalty enforces a constraint on the gradients magnitude and stabilizes the overall training.

Further tricks such as independent learning rates for both G and D via a two time-scale update rule (TTUR) [39], one-sided label smoothing (occasionally randomizing the discriminators output) or instance noise [93] (adding progressively decreasing amounts of noise to both real and generated images before feeding them to the discriminator) can improve the GAN stability to a great extent. The nature of our method - described in the next chapter - introduces additional penalization to the gradients propagated from the discriminator to the generator. We therefore strongly rely on many of the above mentioned techniques to guarantee the generator's convergence. For an exhaustive and thorough analysis of GANs, we suggest the comprehensive review by Gui et al. [34].

Generative Alternatives. GANs fall in the category of implicit density generative models. Implicit density models generate a function to produce data instances from a learned distribution. The model will then draw samples and adapt accordingly until it resembles the true data distribution. On the other hand, explicit density models estimate the true probabilistic density function via a model whose parameters are optimized from true data samples. The biggest well-established players in the field of implicit density models are Variational Autoencoders (VAEs) [49], formed by an encoder and decoder. The encoder maps the image space to a latent space with reduced dimensionality, while the decoder then maps the latent back to the image space. VAEs approximate the posterior distribution by minimizing the ELBO (evidence lower bound) since the true posterior is not tractable. The ELBO comprises two terms: a reconstruction loss which maximizes the sample likelihood, and a Kullback-Leibler (KL) divergence term, encouraging the latent Gaussianess. Even though essential for consistency and interpolation between sampled points, the KL divergence introduces blurriness in the generated images since any point now belongs to a multitude of clusters in the latent space.

Recently, a new class of models inspired by non-equilibrium statistical thermodynamics [92] have gained a lot of attention. Diffusion Models [40, 73] have since then claimed producing better results than GANs [23]. The idea behind Diffusion Models is to slowly and iteratively destroy the structure of the data via noise addition in a forward diffusion process, and to restore structure in the data by learning the added noise through a reverse diffusion process. In the forward process, one can add noise drawn from a normal distribution in a single step by leveraging the properties behind normally distributed random variables. The reverse step then simply predicts the noise between two time steps from the forward process with a U-Net like neural network. Despite being effective, the reverse diffusion process requires small steps in each iteration, as estimating a slight perturbation in the image signal is more tractable than predicting the full distribution in one single step. In fact, Diffusion Models aim to maximize the likelihood of the posterior distribution similar to VAEs, but in a tractable way via clever reformulation of the variational lower bound (unlike the ELBO). Tractable likelihood-based models have desirable properties as they do not exhibit instabilities during training or mode collapse, which is why Diffusion Models have received a continuously growing interest since their discovery.

Albeit providing insights into the major components of our model, we are aware that this chapter might not cover all implemented aspects and tools. Further documentation on the machinery behind convolutional neural networks, the mathematics behind the rendering equation, conditional GANs or many more generative alternatives could have been explored to a greater extent. We decided to only provide intuition for the main building blocks of our pipeline.

Chapter 4

Methods

We present a lightweight generative model for 3D multi-view consistent full-body avatars. To this end, we first devise an expressive, flexible and powerful point-based appearance representation. In a second part, we then introduce a novel yet simple generative module, directly extending on traditional 2D GAN-based architectures.

4.1 Point-based appearance representation

Our point-based module builds on the SMPL-X [78] extension from the SMPL [62] family, thoroughly described in Chapter 3.1. We leverage the UV mapping from the minimally clothed mesh to define the detailed appearance of a subject. The UV field $A = [A_{rgb}, A_\delta] \in \mathbb{R}^{w_a \times h_a \times 4}$ consists of RGB colors and geometry information for every point on the surface. The geometry information is captured as a displacement of a point along the SMPL-X mesh surface normal. Here, (w_a, h_a) are the width and height of the appearance map. With this, we form a resulting point cloud $X = \{x = (x_{xyz}, x_{rgb}) \in \mathbb{R}^6\}$ by sampling points x^m_{xyz} from the surface of the mesh defined as in Equation 3.1, and obtaining their final 3D locations and colors using the UV map A:

$$x_{xyz} = x^m_{xyz} + \delta \cdot \mathbf{n}_{xyz}, \tag{4.1}$$

$$\delta = A_\delta[u, v], \tag{4.2}$$

$$x_{rgb} = A_{rgb}[u, v], \tag{4.3}$$

with UV coordinates of the surface point (u, v), and $\mathbf{n}_{xyz}$ is the normal direction of the surface mesh at location x^m_{xyz}. For reference, we will denote the full step of obtaining a human point cloud given the mesh vertices V and appearance map A as $h(V, A)$:

$$h(V, A) : \mathbb{R}^{3N} \times \mathbb{R}^{w_a \times h_a \times 4} \rightarrow \mathbb{R}^{6L}, \tag{4.4}$$

where L is the number of sampled points.

Similar representations of geometry and appearance have been explored in prior works [3, 4, 2, 5, 6]. However, one key difference is that we treat the output of this stage as a point cloud rather than a watertight mesh, which potentially can also represent double-layered surfaces and other complex topology like glasses. Note that our point-based module is not restrained to minimally clothed meshes. Our model can indeed generalize to any geometry with UV coordinates, although in practice a close proxie will achieve better robustness to pose and shape deformations, and insure layout consistencies between different learned appearances.

4.2 Lightweight 3D aware implicit GAN

The above introduced point-based module enables optimization of both point colors and point locations stored into 2D UV texture maps on top of a given underlying mesh. We should therefore in theory be perfectly capable of learning surface geometry and appearance of human bodies using traditional 2D generators. We introduce *3DiGAN*, a **3D** aware **i**mplicit **G**enerative **A**dversarial **N**etwork for high-quality multi-view consistent appearance generation. Our GAN-based approach synthesizes images in an *implicit* UV state by extending an implementation of [57] by GitHub user *lucidrains* [63]. In particular, we augment the fast and lightweight convolutional GAN pipeline to rasterize the output of the generator G. Instead of feeding the discriminator D with the generated images right away, the images are first rendered with a point rasterizer on top of a SMPL-X mesh according to Equation 4.4, and only the renders are provided to D. The generator learns from gradients back-propagated all the way from D, through our point representation, up to the generated textures via the point-based neural renderer. The full pipeline is depicted in Figure 4.1.

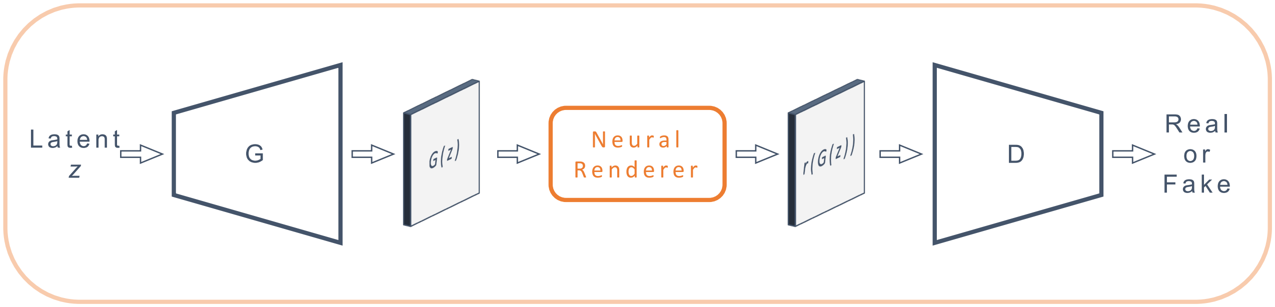


Figure 4.1: Forward pass of the 3DiGAN pipeline. A random sample z drawn from a latent normal distribution is passed through the generator G. The output $G(z)$ is then rendered on a SMPL-X mesh in the neural renderer r. The render $r(G(z))$ finally serves as fake labels during training, and is provided to the discriminator D alongside true samples from a dataset of full-body humans. If trained perfectly, $G(z)$ is a SMPL-X UV appearance texture map and $r(G(z))$ a realistic point cloud render of a deformed SMPL-X body with detailed color and geometry.

G is then trained on a dataset of full-body humans to generate SMPL-X UV appearance maps A in a simple yet clever architecture, as introduced by Liu et al. [57]. The generator (see Figure 4.2) enables gradients to robustly and quickly propagate through the network via Skip-Layer Excitation (SLE) modules. SLE modules present a low-cost redesigned skip-layer connection [36] to strengthen the gradients' flow between different feature layers. It convolves the spatially smaller feature map to match the number of channels of the larger feature vector and then perform a simple element-wise multiplication on each feature channel. Instead of optimizing the networks with the original loss function [30], the generator and discriminator are trained with a hinge adversarial loss [56, 95]:

$$\begin{aligned}\mathcal{L}_D &= -\mathbb{E}_{x\sim p_d(x)}\big[min(0,-1+D(x))\big] + \mathbb{E}_{z\sim p_z(z)}\big[min(0,-1-D(G(z)))\big],\\ \mathcal{L}_G &= -\mathbb{E}_{z\sim p_z(z)}\big[D(G(z))\big],\end{aligned} \tag{4.5}$$

where $\mathcal{L}_D$ is the discriminator loss and $\mathcal{L}_G$ equivalently the generator loss, and convergence state $\mathcal{L}_D = 2$ and $\mathcal{L}_G = 0$. Further information on the self-supervised discriminator pipeline and other technical details are available in the original publication [57].

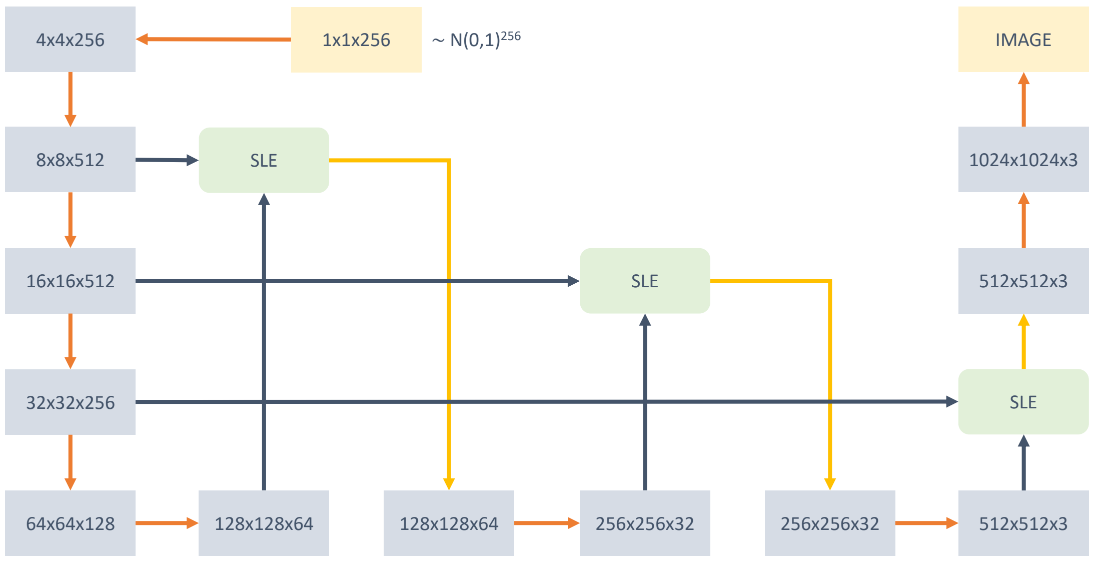


Figure 4.2: Generator architecture as presented in [57]. Yellow and blue boxes represent the input/output of the generator, respectively the feature layers of the generator. Up-sampling structures (mainly transposed convolutions) are shown with orange arrows. Finally, SLE modules are visualized as green rounded boxes with blue arrows as input layer and yellow arrows towards the recipient feature map. The feature dimensions are presented in each box as *width* $\times$ *height* $\times$ *number of channels*. Note that the pipeline is not bounded to generate images of resolution 1024×1024; one can drop the last few layers to shrink the network capacity and generate lower resolutions.

Figure 4.2 shows a generator for 3-channel RGB images. The UV field we are interested in however takes an additional fourth channel for point displacements. Although the generator can easily be extended to 4-channel image synthesis, we explicitly avoid doing this as it fails in capturing meaningful displacements - further explanations provided in Chapter 6. We instead introduce a very simple convolution-based image-to-image translator f, which we append to the generator. f takes in the current generated RGB UV texture and projects it towards a single-channel displacement UV map:

$$G(z) = A = [A_{rgb}, A_\delta], \quad A_{rgb} = G_{rgb}(z), \quad A_\delta = f(A_{rgb}), \tag{4.6}$$

where G_{rgb} is the generator per se as shown in Figure 4.2, and the notation for A is borrowed from Chapter 4.1. Since f becomes part of our generative process, it is trained in the same way as G_{rgb} with discriminator gradients.

Chapter 5

Experiments and Results

The focus of this chapter is to give valuable insights about the experiments completed during the thesis duration, alongside their respective results. The chapter is split in four sections: three of them review different scenarios, while the first section presents the tools and implementation details of our main method. The purpose of each experiment is to verify one hypothesis, where the previous outcome is oftentimes the trigger for a new hypothesis. All our conducted evaluations run on a single NVIDIA GeForce RTX 2080 Ti consumer GPU.

5.1 Implementation details

Dataset. Our final architecture trains on SHHQ [25], a dataset of high-quality full-body human images in a resolution of 1024×512. Although the authors promise a total of 230'000 images, only 40'0000 images are made publicly available at the time of writing. The dataset consists of raw images of fully visible mid-body aligned humans with their respective segmentation masks. We remove the raw images' background using the available masks, and reshape them to a square resolution of 1024×1024 by infilling the sides of the images with a white background, while keeping the original image centered. In its current stage, our pipeline only supports square image ratios for both generator output and discriminator input. Our method requires a set of SMPL-X meshes, such that when combined with an appearance texture, resembles a true distribution of full-body real humans. For that, we extract a SMPL-X mesh for each subject in our canonicalized SHHQ dataset with PIXIE [24], a regression model for expressive bodies from a single image. Since PIXIE outputs considerably more information than what we need, we run a script to only extract SMPL-X poses, shapes, expressions and orientations, and store all these parameters for the 40'000 images together into a NumPy *.npz* file. A visualization of the first 10 SHHQ subjects with their respective PIXIE output mesh is shown in Figure 5.1.

Neural Renderer. The overall GAN architecture we build on top of [63] mostly remains unchanged, and only few parts have been slightly modified to fit our purpose. Our method however leverages an additional rendering step at the intersection of our two adversarial networks. We previously mentioned employing a point-based renderer for the appearance representation. Precisely, we actually build on the Pulsar backend for differentiable sphere-rendering [54] as implemented in PyTorch3D [84]. During training, we sample 10^5 points from the SMPL-X mesh surface, and set the Pulsar transparency coefficient gamma to 0.001 and each sphere radius to 0.0008. Our method is fully modular, as loading data samples, the SMPL-X forward, and rendering in PyTorch3D are all batchable. Finally, rendering enables different resolutions for

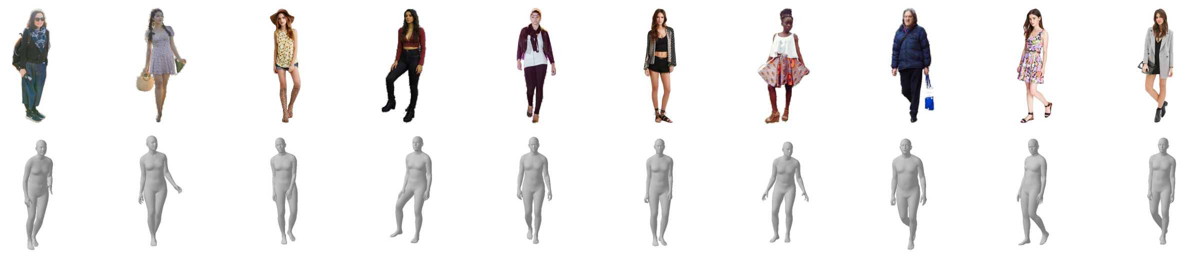


Figure 5.1: SHHQ samples (top) and their respective PIXIE mesh output (bottom).

the generator output and the discriminator input, allowing to discriminate on high resolution data while keeping the generation process low resolution and thus low cost. We nonetheless generate and discriminate on the same resolution of 256×256 for better results (see Chapter 5.3).

5.2 Multi-view single scene, RGB only

Learning a distribution of UV textures from real images in a coherent manner comes with challenges, and a clear understanding of the constraints is a necessity. To develop our comprehension of the subject, we start with a simplified RGB only multi-view setting for a single scene. In particular, we conduct a set of experiments on a cow mesh [82] made available by PyTorch3D. We render the mesh with a single point light from 2'000 uniform random views on a circle around the mesh and allow for arbitrary negative and positive spherical elevations of up to 30 degrees (i.e., the rendered camera angles are always sampled from a narrow region on a sphere with fixed radius). These views are rendered at a resolution of 1024×1024 and constitute the training data for this section. Four random samples are depicted alongside the target ground truth UV texture in Figure 5.2.

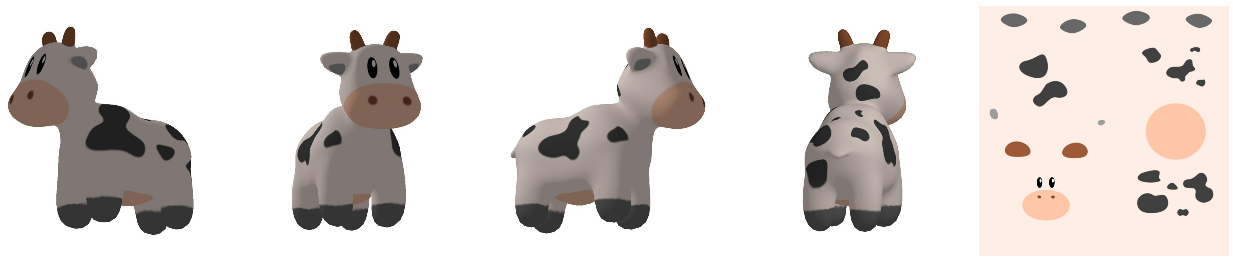
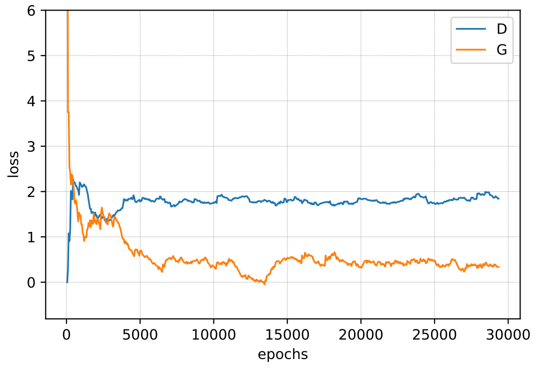


Figure 5.2: Random samples from the cow mesh dataset and target ground truth UV texture.

In the conducted experiments, we assume knowledge of the camera views distribution and, furthermore, render directly on the ground truth mesh. We consequently render the ground truth geometry from similar views as in the training data at a resolution of 256×256, for which we hope to accurately reconstruct the corresponding RGB UV texture with same learned resolution, but with a GAN loss rather than a typical photometric loss. For each experiment we provide two figures: a plot for the discriminator and generator training losses, alongside a set of generated UV textures and their respective renders from a random view for the last model checkpoint. Note that the plots are capped at 8'000 epochs (with the exception of the first more demonstrative experiment). All runs will either have reached their maximal wall-clock time by then, or will have been interrupted due to clear convergence or divergence of the losses. Please pay attention to the scale of the y-axis, as it fluctuates to get the most out of the available space for each figure.

Each experiment runs for 24 hours, if not interrupted before. We unconventionally disable any data augmentation, as our goal is reconstruction rather than learning a distribution. The batch size is fixed to 8 and we additionally accumulate gradients for 4 epochs (replicating batching without any further memory requirements). Lastly, the generator and discriminator are optimized with Adam and an identical learning rate of 0.0002 for both networks. These parameters stay unchanged for all our scenarios, unless stated otherwise.

Default Architecture. Before blindly diving into the multi-view setting, we first confirm the generator's ability to converge onto a single image without any rendering, i.e., with a typical GAN model. We select an arbitrary image from our constructed dataset, namely we opt for a render of a side facing cow, and utilise this single image as our training set. We then let the model train for around 30'000 epochs to guarantee convergence. Despite not perfectly reaching the convergence state of $D = 2$ and $G = 0$, the generator is still capable of flawlessly reconstructing the training image, as seen in Figure 5.3.



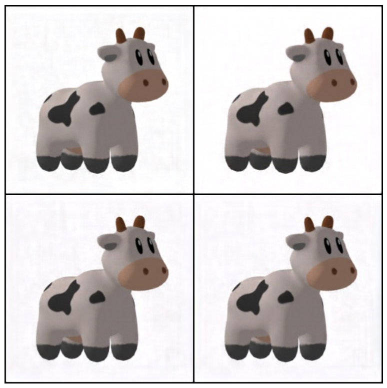


Figure 5.3: GAN training loss for default architecture without rendering (left) and corresponding sampled generated results at convergence (right).

Default Architecture with Rendering. Now that we validated the generator's capacity to overfit to a single scene, we extend it with the previously discussed neural renderer. Our early evaluations operate with the default PyTorch3D point renderer [100] instead of Pulsar, with 10^5 sampled points from the mesh surface. Albeit showing a promising trend early during training, the generator quickly diverges and steadily gets worse. Unsurprisingly on the other hand, the discriminator is flawlessly capable of distinguishing real from fake samples. Figure 5.4 shows both sampled results at the last model checkpoint (middle figure) and at the global generator loss minimum around 1500 epochs (right figure). Even though the synthesized quality at optimality is considerably better than later during training, it still suffers from blurriness and other undesired artefacts. Notice that future experiments do not include intermediary results, as they are less informative about the generation quality. Can we achieve better convergence by guiding our networks with more precise camera poses similarly to regular reconstruction methods?

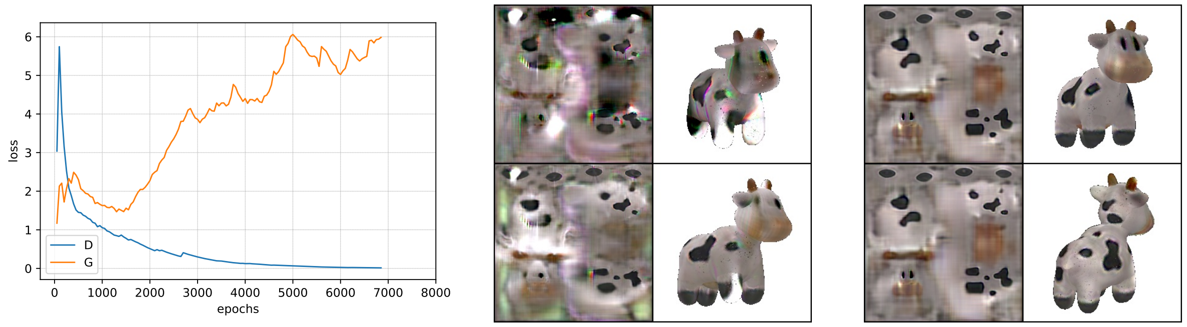


Figure 5.4: GAN training loss for default architecture with rendering (left), corresponding sampled generated results at generator optimality (middle) and at training abortion (right).

Camera Conditioning. We borrow ideas from traditional photogrammetry by taking advantage of camera poses for each training sample. Instead of optimizing a pixel-wise loss for specific target views, we condition both our networks on camera azimuths and elevations. We condition in a similar fashion to StyleGAN [48], via a simple fully connected mapping from the labels towards the generator latent space and towards the discriminator image space. From Figure 5.5, we can conclude that additional camera labelling only hurts the generator. One possible explanation is that we learn a view-indepedent UV texture; thus providing view-dependent information only further confuses G. Moreover, as each camera angle has the same likelihood of occuring, the discriminator should also be view-independent. A possible remedy could be to constrain the generator to a single unique latent vector instead of learning a full distribution.

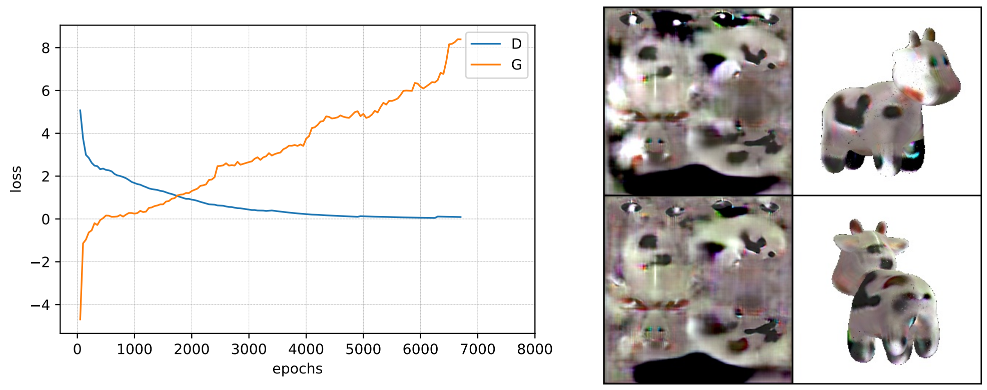


Figure 5.5: GAN training loss for camera conditioning method (left) and corresponding sampled generated results at training abortion (right).

Single Latent Code. In order to enforce a unique learned UV texture and potentially help our generator, we suggest a single optimized latent code. Instead of sampling from a normal distribution, we fix the latent to a constant vector which we then optimize alongside the generator. Figure 5.6 however clearly invalidates our hypothesis with a slowly worsening generator. Indeed for a fixed generator, slight changes in the latent can have drastic effects in the image space. By optimizing both G and the latent code, the generator weights might be constantly chasing after the optimal latent and vice-versa, inevitably resulting in the discriminator taking the lead. We discard this idea, as we anyways already demonstrated the default GAN's ability to overfit to a single target. Note that the two shown textures are identical down to the pixel, as only a single latent is available at each time. By investigating the conducted evaluations so far, we realize that the common denominator is a discriminator converging too quickly. The next experiments all focus on penalizing D with various popular regularization techniques to give the generator a chance of catching up.

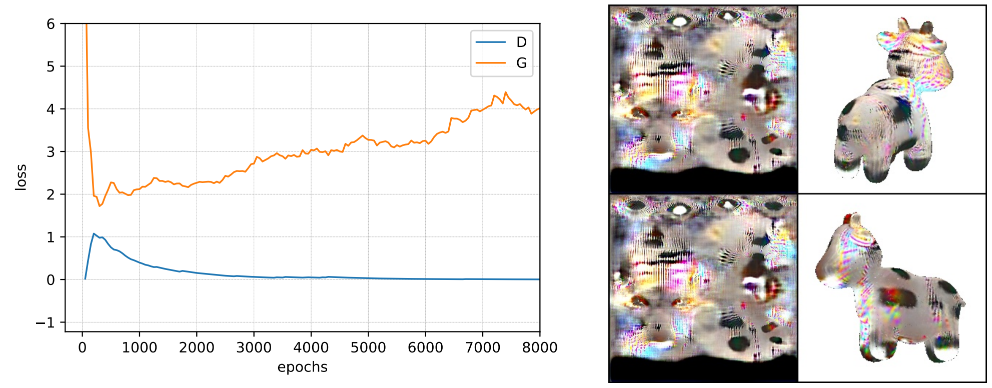


Figure 5.6: GAN training loss for a single optimized latent code (left) and corresponding sampled generated results at training abortion (right).

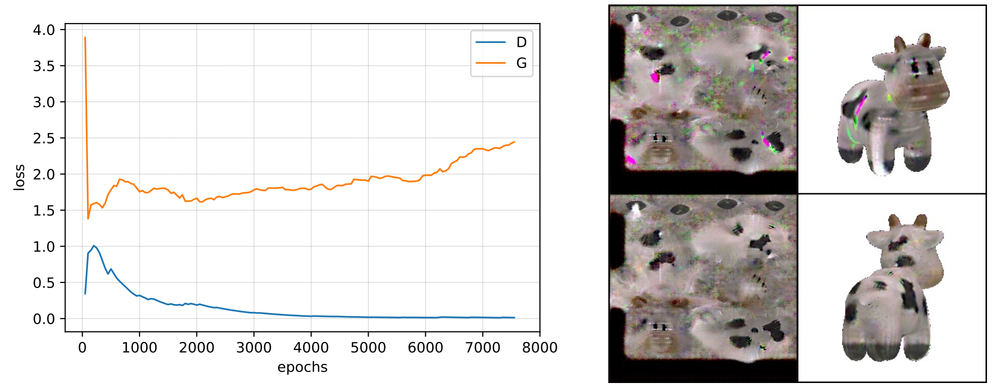


Figure 5.7: GAN training loss with TTUR coefficient set to 0.5 (left) and corresponding sampled generated results at training abortion (right).

Two Time-Scale Update Rule. The learning rate is an essential hyperparameter that regulates the optimization step size in learning-based algorithms. By default our GAN uses the same learning rate for both G and D, but a two time-scale update rule (TTUR) [39] allows to manipulate each learning rate independently. The TTUR coefficient proportionally scales the discriminator learning rate up or down, and consequently slows down or speeds up the learning for D. We evaluate three different scenarios by setting the TTUR coefficient to 0.5, 0.1 and 2.0 (see Figures 5.7, 5.8 and 5.9 respectively). We expect better results for the two first examples, and worse performance for the larger coefficient. In addition to TTUR, we also introduce label smoothing by occasionally randomizing the discriminators output with 20% probability, and accumulate gradients for a longer period of 16 epochs (previously 4), as larger batch sizes correlate positively with GAN convergence. These techniques will be applied in future experiments as well. As expected, slowing down the discriminator allows for the generator to produce better results, despite producing striped artefacts (see face region in Figure 5.8) and still not converging towards the desired state. Our observations lead us to think that a low TTUR coefficient of 0.1 is adequate and yields the best quantitative and qualitative results so far.

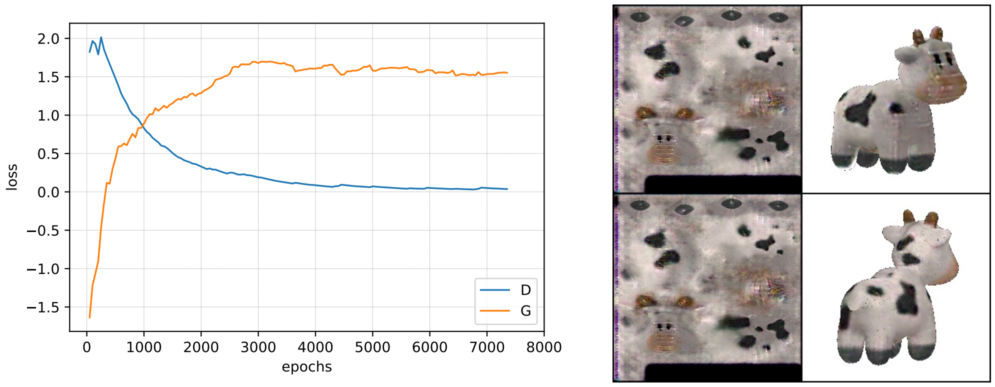


Figure 5.8: GAN training loss with TTUR coefficient set to 0.1 (left) and corresponding sampled generated results at training abortion (right).

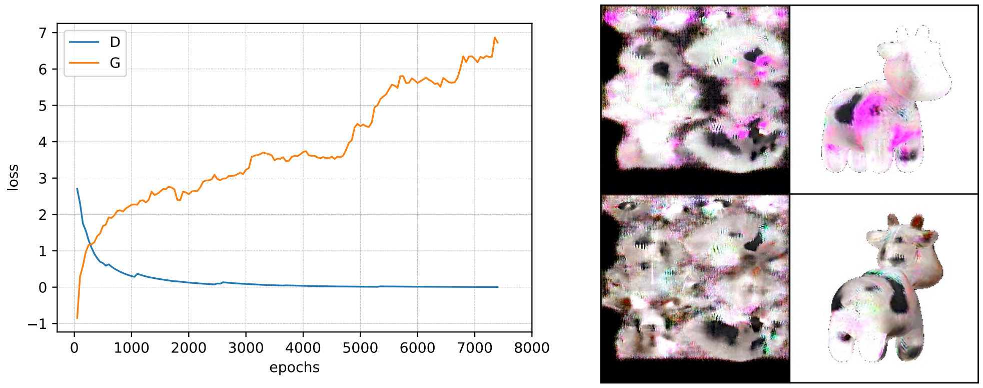


Figure 5.9: GAN training loss with TTUR coefficient set to 2.0 (left) and corresponding sampled generated results at training abortion (right).

Gradient Penalty. Although our current implementation already penalizes gradient magnitudes larger than one, we evaluate the effect of stronger regularization by scaling the gradient penalty weight [35] up to 40 (set to 10 by default). As stated in the last paragraph, we maintain a TTUR coefficient of 0.1, accumulate gradients every 16 epochs, and only slightly raise the label smoothing probability to 40%. Results are depicted in Figure 5.10, but seem to be invariant to the gradient penalty scaling. We believe that the default gradient penalty coefficient is already enough to restrain the gradients' magnitude.

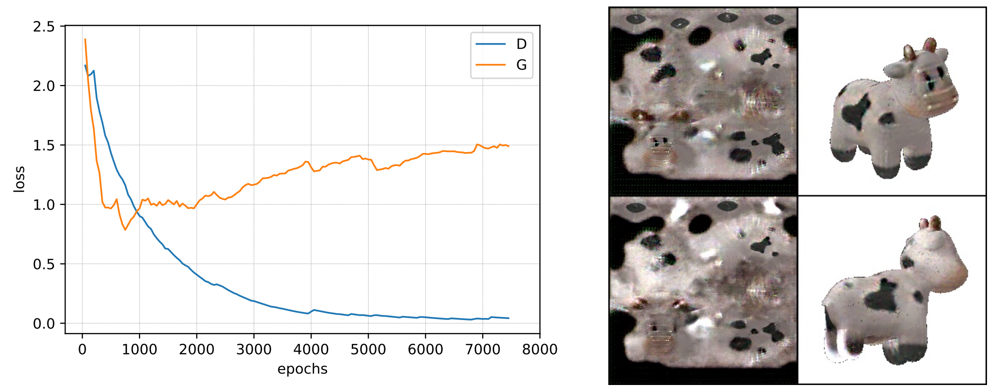


Figure 5.10: GAN training loss with scaled gradient penalty (left) and corresponding sampled generated results at training abortion (right).

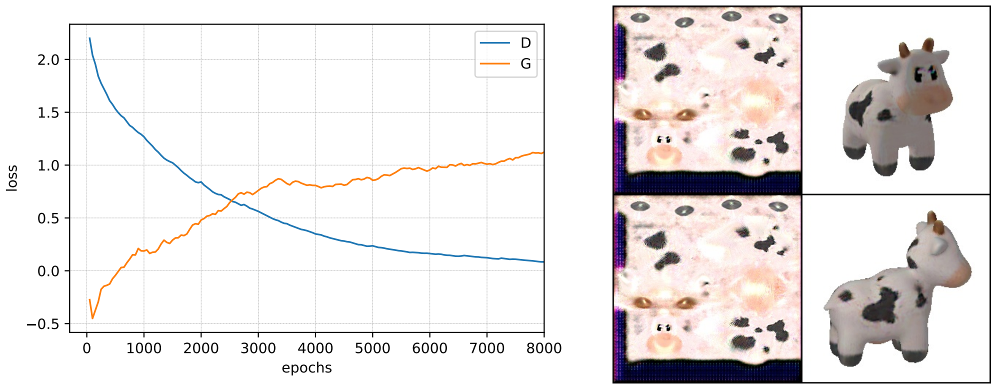


Figure 5.11: GAN training loss with discriminator dropout (left) and corresponding sampled generated results at training abortion (right).

Discriminator Dropout. Having experimented with many unsuccessful settings, we take a step back and wonder whether the point rasterizer itself might not be the problem. From close inspection of the renders, one can notice that the edges of our geometry are not smooth nor continuous. Switching back to mesh rendering might resolve this issue, as the described artefacts might be a side effect of the rasterization algorithm. We decide to simplify our setting even more by dropping the point sampling step and directly rendering the default mesh with a single point light at the same location as in the training renders. To further penalize the discriminator, dropout layers with probability 0.2 are added in the discriminator between each convolution and LeakyReLU activation function. Label smoothing is maintained at probability 0.4 and the TTUR coefficient at 0.1. Here the results in Figure 5.11 are interesting, since G produces the best qualitative results so far (see the sharpness around the face region in the UV space) but still does not converge as desired. The change in brightness and saturation of the UV textures is not a direct product of mesh rendering, but instead of the added point light. In fact, illumination effects were previously baked into the textures. The results are nevertheless still considerably better than with the point rasterizer. On a side note, keep in mind that mesh rendering is not suited for our method, as it does not support displacements in the same precision as a point-based approach. We later circumvent this by substituting the default point renderer for the faster and more accurate Pulsar backend. With all this in mind, can we finally achieve convergence with precise UV reconstruction?

Discriminator Scheduling. For our last experiment we hinder our discriminator one more time with a scheduling on its learning, by updating the network only every 10 epochs rather than at each iteration. Everything else stays unchanged from the previous paragraph (dropout, label smoothing and TTUR). Our model is finally capable of converging in the multi-view and single-scene scenario but not exactly towards the expected convergence state. Compared to the previous experiment, the generated images unexpectedly seem to be more noisy and have worse overall reconstruction fidelity. We therefore discard the scheduling approach in future single-view experiments. Results are depicted in Figure 5.12.

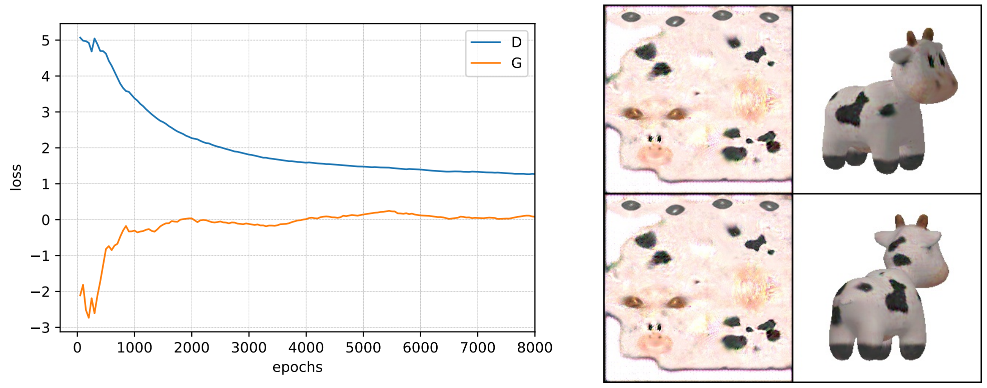


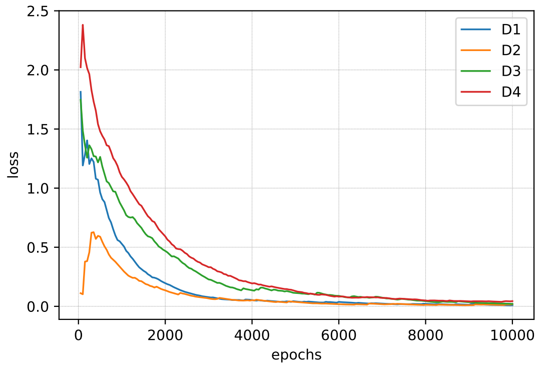
Figure 5.12: GAN training loss with discriminator scheduling (left) and corresponding sampled generated results at training abortion (right).

Table 5.1: Quantitative evaluation of regularization techniques in the context of multi-view single scene RGB only generation. The metric of evaluation is the FID score.

	REGULARIZATION	FID ↓
1	Default	189.36
2	1 + Label smoothing	177.47
3	2 + TTUR coefficient 0.1	228.87
4	3 + Dropout	200.17

Ablation and Overview. The above scenarios have been evaluated over a long period of time with several iterations and modifications; it can be difficult to pinpoint which approaches were actually beneficial or detrimental. Hence this last paragraph aims at summarizing the important results in a short and structured ablation study. We exclusively focus on the effects of regularization, particularly label smoothing, TTUR and dropout, as they seem to correlate the most with convergence improvements. For a fair comparison, we employ the Pulsar backend instead of default point or mesh rendering. Pulsar has a larger memory footprint which is why we decrease the batch size to 2, while still accumulating gradients for 16 epochs. We set the

Pulsar transparency coefficient gamma to fully opaque (0.00001), the sphere radius to 0.01, and sample 10^5 points from the mesh surface. We start with the default architecture and one-by-one add each regularization trick to the previous experiment. The study runs for 10'000 epochs to ensure potential convergence. FID scores after 10'000 epochs, GAN training plots and qualitative results are available in Table 5.1, Figure 5.13 and 5.14.



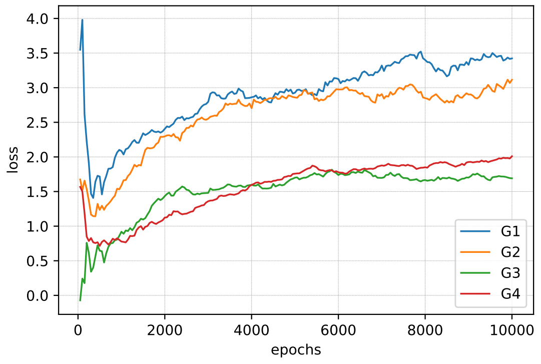


Figure 5.13: Discriminator (left) and generator training losses (right) for ablation study on the cow mesh. Numeration as is Table 5.1.

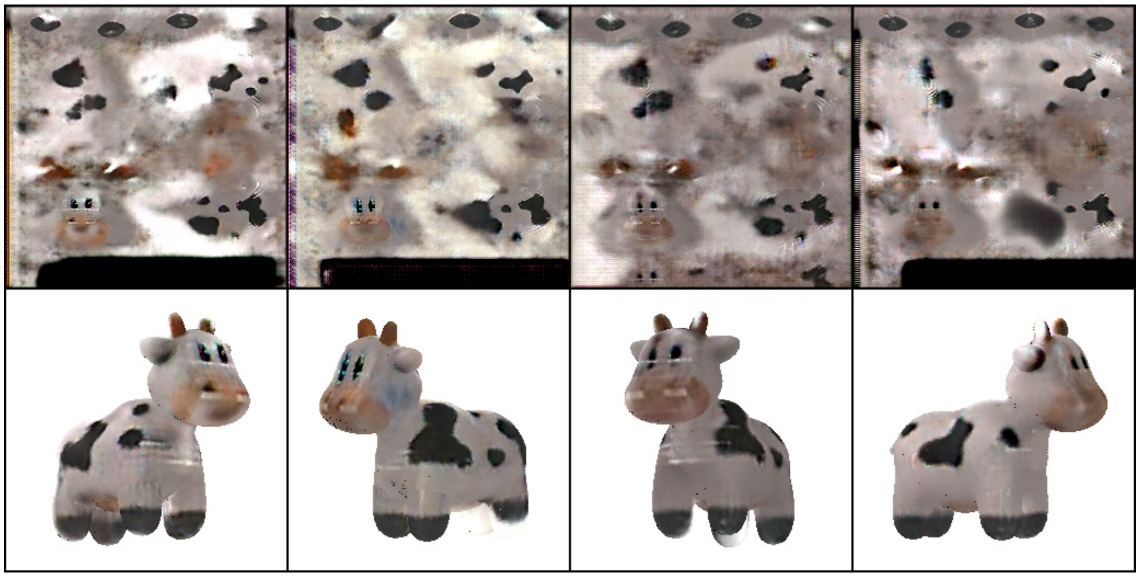


Figure 5.14: Sampled UV generated results (top) and corresponding renders (bottom) at 10'0000 epochs for each ablation study experiment. Left (1) to right (4) following the numeration from Table 5.1.

Against our expectations, convergence does not necessarily correlate with better quality. In fact, restraining the discriminator evens out the chances for the generator to catch up, but likely at the cost of less meaningful gradients. The label smoothing FID score validates this hypothesis, as it does not penalize the discriminator at all but rather randomizes the loss instead. We can additionally confirm that the qualitative improvements of our previous dropout experiments were a consequence of mesh rendering rather than the dropout itself. Justifying why some techniques have a larger impact on the overall quality remains however an open question. Chapter 6 will later discuss why our generative pipeline fails to perfectly reconstruct single scenes.

5.3 Single-view scenes

So far, we have explored UV generation given the target geometry and multiple views from the same scene. We now assess our method on one-shot full-body human photographs, first for a single-view, then a large set of independent single-views from the SHHQ dataset [25]. We extend our pipeline to be fully batchable by performing a SMPL-X forward on one-time loaded parameters, instead of loading meshes as *.obj* files individually. For all experiments, we use Pulsar with the same hyperparameters as discussed in the *Neural Renderer* paragraph in Chapter 5.1 and a batch size of 2, while accumulating gradients for 16 epochs. For evaluations on scene reconstruction, we again disable data augmentation. All other experiments on the full SHHQ dataset augment each sample with 25% probability.

Single-view Reconstruction. This section examines preliminary evaluations for the final framework on multiple single-view photographs. The goal is to fit the appearance of the person in the left of Figure 5.15, given the PIXIE [24] output mesh on the right side. We purposefully select a subject with little aberration from the corresponding minimally clothed body to facilitate our investigation. Furthermore, we now enable discriminating at a different resolution than generation, since rendering of the synthesized UV textures can be done at any scale.

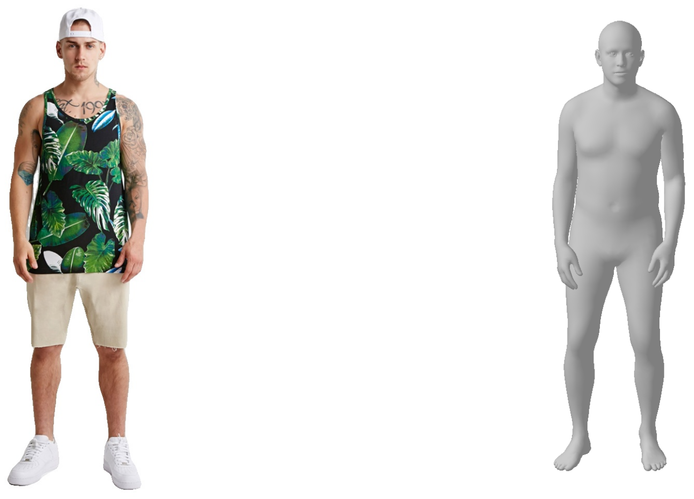


Figure 5.15: Sample from the SHHQ (left) and corresponding PIXIE prediction as a SMPL-X mesh (right).

Six different scenarios are evaluated. We start by comparing the effects of different learning rates while applying the label smoothing trick with probability 0.2 at a resolution of 256×256 (Figure 5.16). In a second stage, we repeat the same evaluations, but render at a resolution of 1024×1024 (Figure 5.17). The generated UV textures remain low resolution, but the discriminator now manipulates high resolution images. These four settings exclusively operate on color appearance. The last two experiments introduce a new channel for displacements in two distinguish ways. First we naively extend the generator to synthesize

4-channel images. Later we map the learned RGB UV textures through a simple image-to-image translator f as discussed in Chapter 4.2.

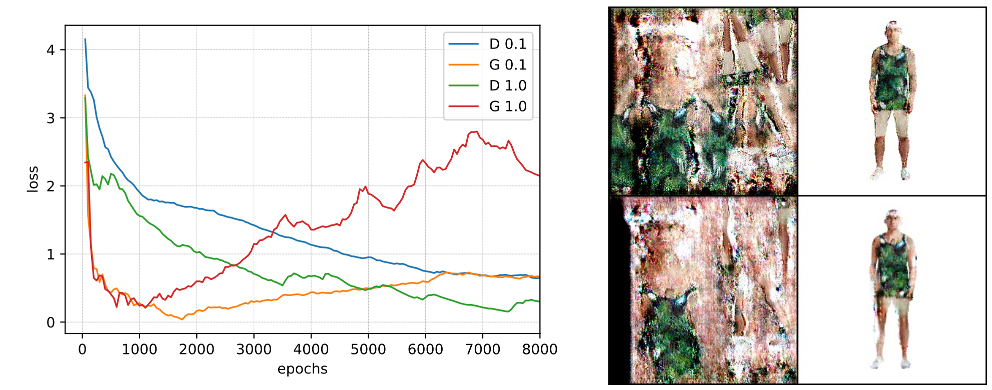


Figure 5.16: GAN training losses for TTUR comparison (left) and corresponding sampled generated results at training abortion (right), resolution 256×256. TTUR coefficient 0.1 (top) against default 1.0 (bottom).

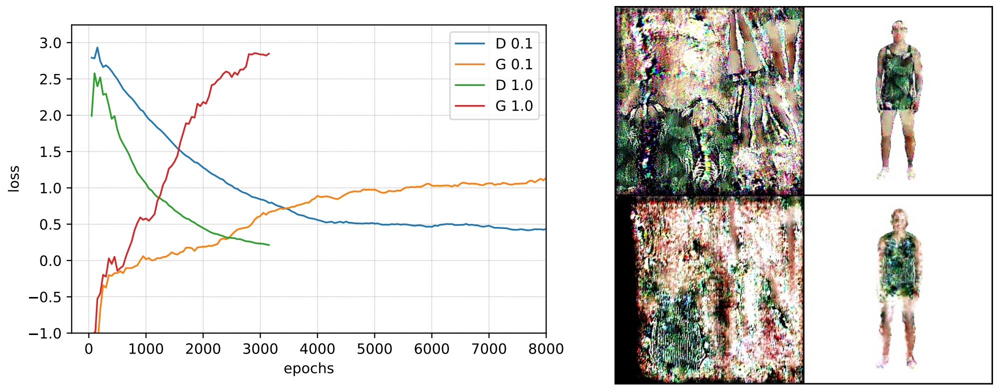


Figure 5.17: GAN training loss with TTUR comparison (left) and corresponding sampled generated results at training abortion (right), resolution 1024×1024. TTUR coefficient 0.1 (top) against default 1.0 (bottom).

For both high and low resolutions, the effects of a smaller learning rate on the discriminator are evident. The experiments with TTUR coefficient 1.0 at a resolution of 1024×1024 were interrupted early, as divergence was undeniable. One might think that discriminating at higher resolutions should help the generator to synthesize improved quality data. While in essence this is a valid assumption, we believe it fails in practice since two neighboring pixels in high resolution renders have a strong likelihood of colliding into a single pixel on lower resolution renders. Indeed on larger resolution renders, the discriminator is capable of perceiving a larger pixel surface area of the underlying UV texture. D might therefore only propagate meaningful gradients to the visible regions, discarding and deteriorating large portions of the generated UV map.

We finally investigate the results with enabled displacements, both via naive extension to 4-channels and with an image-to-image translator. We set the TTUR coefficient to 0.1 and render at a resolution of 256×256.

Training plots and qualitative results are shown in Figure 5.18, where only the first three RGB channels are visualized. We interrupt the run for naive displacement learning early, as it undoubtedly fails in capturing geometry accurately. This behavior is a consequence of convolution-based learning, when entangling all output channels with a shared receptive field. Our image-to-image function does only a slightly better job with more regulated displacements, but eventually breaks down and progressively diverges.

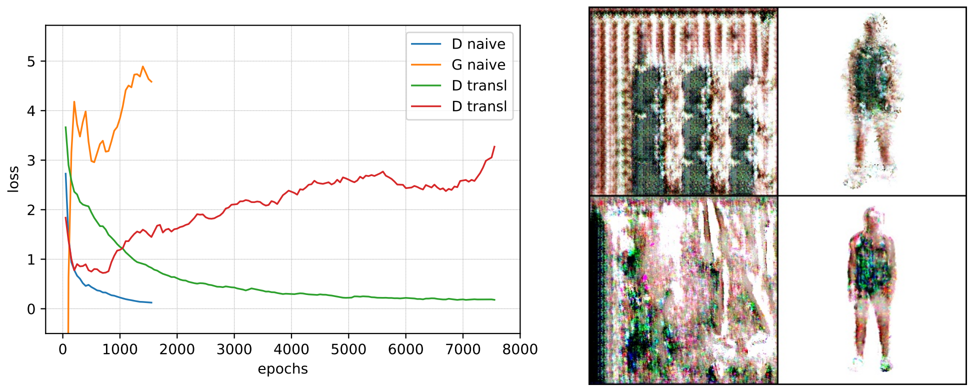


Figure 5.18: GAN training loss with displacements (left) and corresponding sampled generated results at training abortion (right), resolution 256×256. Naive displacements implementation (top) against image-to-image translator (bottom).

3D Aware Synthesis. To capture a realistic distribution of human appearance, we need a strong model for both surface color and geometry. In its current form, our framework only supports color; further investigation on displacement optimization is required. This last paragraph nonetheless evaluates our pipeline (RGB channels only) on the full available SHHQ dataset. When training the generator, we sample SMPL-X meshes from the PIXIE regressions as underlying geometry for the UV textures. We only provide a qualitative analysis, as our generator is far from converging. After 14'000 epochs, we achieve the results depicted in 5.19. Our model is still far away from convincing synthesis, but we can already observe a few desirable qualities. Our generator is capable of learning body attributes in their correct location, and therefore conforms to the SMPL-X UV layout. Despite a few samples with a similar style (for instance the green stain on the shirt), we also achieve adequate diversity in clothing and skin color. Some harmful artefacts can on the other hand also be noticed, such as the generator compensating for the lack of geometry by inpainting some parts of the texture with a white background. Due to time constraints, these results are only preliminary and incomplete. An exhaustive analysis of the generator's ability to synthesize novel data and an inspection of the latent space are necessary.

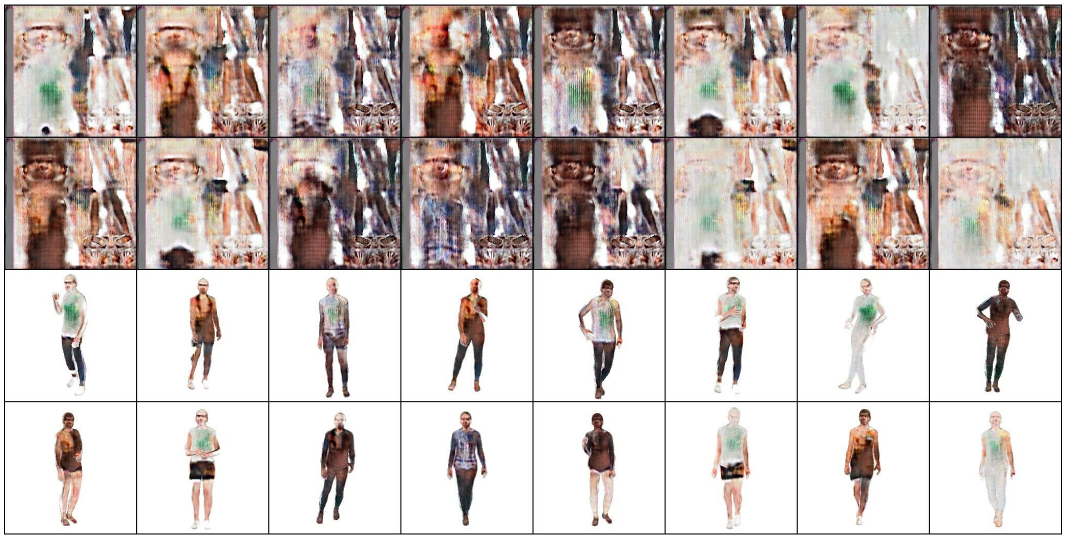


Figure 5.19: Sampled UV generated results (top) and corresponding renders (bottom) trained for 14’0000 epochs on SHHQ with 3DiGAN.

Chapter 6

Discussion

This short chapter analyses the limitations of our pipeline and failure cases encountered in the previous sections. We also briefly discuss implications and challenges of our method.

Downsides of a GAN-based Approach. GANs are notoriously bad at training in a stable fashion. They require a lot of hyperparameter tuning and regularization at different levels. Our method introduces an additional UV mapping step, in which we project the learned images onto a mesh. This mapping is well-defined, but marks a clear separation between generator output and observed true data, as G won't explicitly imitate what the discriminator sees. This weakens the propagated gradients from D to G, and makes it hard for the generator to learn correctly.

On another note, our framework builds on top of classical convolution-based GANs. By construction, convolutions locally rely on neighboring information. Spatial consistency in a generative forward pass is thus primordial for data synthesis. Notice that the observed checkerboard-like patterns in the UV textures of our experiments are a typical signature of transpose convolutions. Although there is a direct correspondence between the 3D space and the texture space, this correspondence does not apply so nicely to the UV projection step. UVs tend to break seams, introduce cuts, stretch and disproportionally reshape the laid out geometry. While learning RGB values is a simple pixel to texel (texture pixel) projection, a major drawback appears for displacements. In order to correctly optimize for changes in geometry, we precisely need to capture distances in the image space, and map those distances as values to the texture space. In summary, measuring distances in the image space with locally dependent convolutions is the main reason why our pipeline fails for displacements. Competing frameworks building on 3D generators [17, 107] fully capture spatial information into separate channels, and therefore do not need to learn any distances in the image space. Replacing our GAN-based approach for more robust methods less susceptible to errors during the UV projection step, could potentially lead to better overall results in the synthesized appearances.

Reconstruction Quality and Convergence. The main focus of our experiments lied in approximating single scenes via UV texture fitting. While the learned appearances might be convincing to a certain degree, we did not achieve high fidelity reconstruction capable of fooling the discriminator. Capturing a single scene with GANs is a counterintuitive task because of the non-deterministic nature of the latent space. Indeed, the generator starts from a Gaussian distribution, but we force it to project any sampled input to the same output. Formally, we try to transform the input distribution to a single spike distribution with zero standard deviation. This fragile procedure makes convergence complicated, as any slight deviation in the generator's weights will have a large effect on the synthesized UV image domain.

Framework Limitations Unlike modern reconstruction methods [68], our framework bakes environment lighting effects into the appearance textures. A solution to mitigate this would be to condition our generator on predicted light intensity and position. This however makes real-time applications impractical, since then a single texture cannot be stored in memory. There might also be entanglement between directions in the latent space, and controlling a single attribute might slightly change the appearance's identity. Note that not only illumination, but in fact any possible attributes are baked into the textures (pose, shape, expression or other environmental factors), but we assume these to be negligible in our texture space.

Ethical Implications. Although our method does not produce photorealistic results yet and still lacks technical innovations, it could in the future be exploited for defamatory content creation. We do not condone the misuse of generative techniques for the synthesis of convincing and deceptive imagery, especially in the context of social misinformation. We also recognize that SHHQ has a potential bias and lack of diversity, which is reflected in the learned distribution of appearances.

Chapter 7

Conclusion

Synthesizing novel photorealistic human avatars will enable populating digital thriving worlds. With this work, we have taken several steps towards this goal. We first proposed 3DiGAN, a lightweight generative model for full-body human appearance synthesis from single-view 2D photographs only. While our framework still has a few limitations, our simple approach differs from modern more complex 3D volumetric-based generators and has a high potential to produce competitive results. We additionally examined various relevant parallel works, and provided a detailed outline of the fundamental building blocks of our method. This review constitutes an informative starting point for many research directions in human modelling, neural rendering and image generation. Later, we investigated our contributions via several conducted experiments, evaluations and discussions in the context of single-scene reconstruction. These pointed out the limitations of our pipeline, but also clarified which steps can be taken to improve its current performance. Our method serves as a strong basis for future work in photorealistic image synthesis of both full-body human avatars and other geometries.

Appendix A

The First Appendix

We will provide the code for 3DiGAN, related experiments and the PIXIE outputs on SHHQ. The code will also later be made publicly available under : `https://github.com/maximeraafat/3DiGAN`.

Bibliography

[1] Kara-Ali Aliev, Artem Sevastopolsky, Maria Kolos, Dmitry Ulyanov, and Victor Lempitsky. Neural point-based graphics. In *European Conference on Computer Vision*, pages 696–712. Springer, 2020.

[2] Thiemo Alldieck, Marcus Magnor, Bharat Lal Bhatnagar, Christian Theobalt, and Gerard Pons-Moll. Learning to reconstruct people in clothing from a single rgb camera. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1175–1186, 2019.

[3] Thiemo Alldieck, Marcus Magnor, Weipeng Xu, Christian Theobalt, and Gerard Pons-Moll. Detailed human avatars from monocular video. In *2018 International Conference on 3D Vision (3DV)*, pages 98–109. IEEE, 2018.

[4] Thiemo Alldieck, Marcus Magnor, Weipeng Xu, Christian Theobalt, and Gerard Pons-Moll. Video based reconstruction of 3d people models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8387–8397, 2018.

[5] Thiemo Alldieck, Gerard Pons-Moll, Christian Theobalt, and Marcus Magnor. Tex2shape: Detailed full human body geometry from a single image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2293–2303, 2019.

[6] Thiemo Alldieck, Mihai Zanfir, and Cristian Sminchisescu. Photorealistic monocular 3d reconstruction of humans wearing clothing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1506–1515, 2022.

[7] Dragomir Anguelov, Praveen Srinivasan, Daphne Koller, Sebastian Thrun, Jim Rodgers, and James Davis. Scape: shape completion and animation of people. In *ACM SIGGRAPH 2005 Papers*, pages 408–416. 2005.

[8] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

[9] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5855–5864, 2021.

[10] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5470–5479, 2022.

[11] Miguel Angel Bautista, Pengsheng Guo, Samira Abnar, Walter Talbott, Alexander Toshev, Zhuoyuan Chen, Laurent Dinh, Shuangfei Zhai, Hanlin Goh, Daniel Ulbricht, et al. Gaudi: A neural architect for immersive 3d scene generation. *arXiv preprint arXiv:2207.13751*, 2022.

[12] Marc G Bellemare, Ivo Danihelka, Will Dabney, Shakir Mohamed, Balaji Lakshminarayanan, Stephan Hoyer, and Rémi Munos. The cramer distance as a solution to biased wasserstein gradients. *arXiv preprint arXiv:1705.10743*, 2017.

[13] Alexander W Bergman, Petr Kellnhofer, Yifan Wang, Eric R Chan, David B Lindell, and Gordon Wetzstein. Generative neural articulated radiance fields. *arXiv preprint arXiv:2206.14314*, 2022.

[14] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3d faces. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 187–194, 1999.

[15] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

[16] Shengqu Cai, Anton Obukhov, Dengxin Dai, and Luc Van Gool. Pix2nerf: Unsupervised conditional p-gan for single image to neural radiance fields translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3981–3990, 2022.

[17] Eric R Chan, Connor Z Lin, Matthew A Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas J Guibas, Jonathan Tremblay, Sameh Khamis, et al. Efficient geometry-aware 3d generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16123–16133, 2022.

[18] Eric R Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5799–5809, 2021.

[19] Wei Cheng, Su Xu, Jingtan Piao, Chen Qian, Wayne Wu, Kwan-Yee Lin, and Hongsheng Li. Generalizable neural performer: Learning robust radiance fields for human novel view synthesis. *arXiv preprint arXiv:2204.11798*, 2022.

[20] Peng Dai, Yinda Zhang, Zhuwen Li, Shuaicheng Liu, and Bing Zeng. Neural point cloud rendering via multi-plane projection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7830–7839, 2020.

[21] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised nerf: Fewer views and faster training for free. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12882–12891, 2022.

[22] AXYZ Design. AXYZ Design website. `https://secure.axyz-design.com`. Last accessed 13 September 2022.

[23] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021.

[24] Yao Feng, Vasileios Choutas, Timo Bolkart, Dimitrios Tzionas, and Michael J Black. Collaborative regression of expressive bodies using moderation. In *2021 International Conference on 3D Vision (3DV)*, pages 792–804. IEEE, 2021.

[25] Jianglin Fu, Shikai Li, Yuming Jiang, Kwan-Yee Lin, Chen Qian, Chen Change Loy, Wayne Wu, and Ziwei Liu. Stylegan-human: A data-centric odyssey of human generation. *arXiv preprint arXiv:2204.11823*, 2022.

[26] Matheus Gadelha, Subhransu Maji, and Rui Wang. 3d shape induction from 2d views of multiple objects. In *2017 International Conference on 3D Vision (3DV)*, pages 402–411. IEEE, 2017.

[27] Xiangjun Gao, Jiaolong Yang, Jongyoo Kim, Sida Peng, Zicheng Liu, and Xin Tong. Mps-nerf: Generalizable 3d human rendering from multiview images. *arXiv preprint arXiv:2203.16875*, 2022.

[28] Stephan J Garbin, Marek Kowalski, Matthew Johnson, Jamie Shotton, and Julien Valentin. Fastnerf: High-fidelity neural rendering at 200fps. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14346–14355, 2021.

[29] Partha Ghosh, Pravir Singh Gupta, Roy Uziel, Anurag Ranjan, Michael J Black, and Timo Bolkart. Gif: Generative interpretable faces. In *2020 International Conference on 3D Vision (3DV)*, pages 868–878. IEEE, 2020.

[30] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 2672–2680, Cambridge, MA, USA, 2014. MIT Press.

[31] Artur Grigorev, Karim Iskakov, Anastasia Ianina, Renat Bashirov, Ilya Zakharkin, Alexander Vakhitov, and Victor Lempitsky. Stylepeople: A generative model of fullbody human avatars. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5151–5160, 2021.

[32] Markus Gross and Hanspeter Pfister. *Point-based graphics*. Elsevier, 2011.

[33] Jiatao Gu, Lingjie Liu, Peng Wang, and Christian Theobalt. Stylenerf: A style-based 3d-aware generator for high-resolution image synthesis. *arXiv preprint arXiv:2110.08985*, 2021.

[34] Jie Gui, Zhenan Sun, Yonggang Wen, Dacheng Tao, and Jieping Ye. A review on generative adversarial networks: Algorithms, theory, and applications. *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[35] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. *Advances in neural information processing systems*, 30, 2017.

[36] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[37] Tong He, Yuanlu Xu, Shunsuke Saito, Stefano Soatto, and Tony Tung. Arch++: Animation-ready clothed human reconstruction revisited. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11046–11056, 2021.

[38] Philipp Henzler, Niloy J Mitra, and Tobias Ritschel. Escaping plato's cave: 3d shape from adversarial rendering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9984–9993, 2019.

[39] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

[40] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[41] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE international conference on computer vision*, pages 1501–1510, 2017.

[42] Zeng Huang, Yuanlu Xu, Christoph Lassner, Hao Li, and Tony Tung. Arch: Animatable reconstruction of clothed humans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3093–3102, 2020.

[43] Wei Jiang, Kwang Moo Yi, Golnoosh Samei, Oncel Tuzel, and Anurag Ranjan. Neuman: Neural human radiance field from a single video. *arXiv preprint arXiv:2203.12575*, 2022.

[44] Hanbyul Joo, Tomas Simon, and Yaser Sheikh. Total capture: A 3d deformation model for tracking faces, hands, and bodies. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8320–8329, 2018.

[45] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

[46] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. *Advances in Neural Information Processing Systems*, 34:852–863, 2021.

[47] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019.

[48] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8110–8119, 2020.

[49] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[50] Naveen Kodali, Jacob Abernethy, James Hays, and Zsolt Kira. On convergence and stability of gans. *arXiv preprint arXiv:1705.07215*, 2017.

[51] Maria Kolos, Artem Sevastopolsky, and Victor Lempitsky. Transpr: Transparency ray-accumulating neural 3d scene point renderer. In *2020 International Conference on 3D Vision (3DV)*, pages 1167–1175. IEEE, 2020.

[52] Georgios Kopanas, Julien Philip, Thomas Leimkühler, and George Drettakis. Point-based neural rendering with per-view optimization. In *Computer Graphics Forum*, volume 40, pages 29–43. Wiley Online Library, 2021.

[53] Youngjoong Kwon, Dahun Kim, Duygu Ceylan, and Henry Fuchs. Neural human performer: Learning generalizable radiance fields for human performance rendering. *Advances in Neural Information Processing Systems*, 34:24741–24752, 2021.

[54] Christoph Lassner and Michael Zollhofer. Pulsar: Efficient sphere-based neural rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1440–1449, 2021.

[55] Verica Lazova, Eldar Insafutdinov, and Gerard Pons-Moll. 360-degree textures of people in clothing from a single image. In *2019 International Conference on 3D Vision (3DV)*, pages 643–653. IEEE, 2019.

[56] Jae Hyun Lim and Jong Chul Ye. Geometric gan. *arXiv preprint arXiv:1705.02894*, 2017.

[57] Bingchen Liu, Yizhe Zhu, Kunpeng Song, and Ahmed Elgammal. Towards faster and stabilized gan training for high-fidelity few-shot image synthesis. In *International Conference on Learning Representations*, 2020.

[58] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *Advances in Neural Information Processing Systems*, 33:15651–15663, 2020.

[59] Lingjie Liu, Marc Habermann, Viktor Rudnev, Kripasindhu Sarkar, Jiatao Gu, and Christian Theobalt. Neural actor: Neural free-view synthesis of human actors with pose control. *ACM Transactions on Graphics (TOG)*, 40(6):1–16, 2021.

[60] Shichen Liu, Tianye Li, Weikai Chen, and Hao Li. Soft rasterizer: A differentiable renderer for image-based 3d reasoning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7708–7717, 2019.

[61] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *arXiv preprint arXiv:1906.07751*, 2019.

[62] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 34(6):248:1–248:16, October 2015.

[63] Phil Wang lucidrains. Implementation of 'lightweight' GAN, proposed in ICLR 2021, in pytorch. `https://github.com/lucidrains/lightweight-gan`. Last accessed 22 September 2022.

[64] Qianli Ma, Shunsuke Saito, Jinlong Yang, Siyu Tang, and Michael J Black. Scale: Modeling clothed humans with a surface codec of articulated local elements. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16082–16093, 2021.

[65] Qianli Ma, Jinlong Yang, Anurag Ranjan, Sergi Pujades, Gerard Pons-Moll, Siyu Tang, and Michael J Black. Learning to dress 3d people in generative clothing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6469–6478, 2020.

[66] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2794–2802, 2017.

[67] Ben Mildenhall, Peter Hedman, Ricardo Martin-Brualla, Pratul P Srinivasan, and Jonathan T Barron. Nerf in the dark: High dynamic range view synthesis from noisy raw images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16190–16199, 2022.

[68] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

[69] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *arXiv preprint arXiv:2201.05989*, 2022.

[70] Thomas Neff, Pascal Stadlbauer, Mathias Parger, Andreas Kurz, Joerg H Mueller, Chakravarty R Alla Chaitanya, Anton Kaplanyan, and Markus Steinberger. Donerf: Towards real-time rendering of compact neural radiance fields using depth oracle networks. In *Computer Graphics Forum*, volume 40, pages 45–59. Wiley Online Library, 2021.

[71] Thu Nguyen-Phuoc, Chuan Li, Lucas Theis, Christian Richardt, and Yong-Liang Yang. Hologan: Unsupervised learning of 3d representations from natural images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7588–7597, 2019.

[72] Thu H Nguyen-Phuoc, Christian Richardt, Long Mai, Yongliang Yang, and Niloy Mitra. Blockgan: Learning 3d object-aware scene representations from unlabelled images. *Advances in Neural Information Processing Systems*, 33:6767–6778, 2020.

[73] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021.

[74] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11453–11464, 2021.

[75] Atsuhiro Noguchi, Xiao Sun, Stephen Lin, and Tatsuya Harada. Neural articulated radiance field. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5762–5772, 2021.

[76] Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. f-gan: Training generative neural samplers using variational divergence minimization. *Advances in neural information processing systems*, 29, 2016.

[77] Roy Or-El, Xuan Luo, Mengyi Shan, Eli Shechtman, Jeong Joon Park, and Ira Kemelmacher-Shlizerman. Stylesdf: High-resolution 3d-consistent image and geometry generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13503–13513, 2022.

[78] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed A. A. Osman, Dimitrios Tzionas, and Michael J. Black. Expressive body capture: 3D hands, face, and body from a

single image. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 10975–10985, 2019.

[79] Sida Peng, Junting Dong, Qianqian Wang, Shangzhan Zhang, Qing Shuai, Xiaowei Zhou, and Hujun Bao. Animatable neural radiance fields for modeling dynamic human bodies. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14314–14323, 2021.

[80] Sida Peng, Yuanqing Zhang, Yinghao Xu, Qianqian Wang, Qing Shuai, Hujun Bao, and Xiaowei Zhou. Neural body: Implicit neural representations with structured latent codes for novel view synthesis of dynamic humans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9054–9063, 2021.

[81] Matt Pharr, Wenzel Jakob, and Greg Humphreys. *Physically based rendering: From theory to implementation*. Morgan Kaufmann, 2016.

[82] PyTorch3D. Render a textured mesh. `https://pytorch3d.org/tutorials/render_textured_meshes`. Last accessed 24 September 2022.

[83] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[84] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv:2007.08501*, 2020.

[85] RenderPeople. RenderPeople website. `https://renderpeople.com`. Last accessed 13 September 2022.

[86] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.

[87] Javier Romero, Dimitrios Tzionas, and Michael J Black. Embodied hands: Modeling and capturing hands and bodies together. *arXiv preprint arXiv:2201.02610*, 2022.

[88] Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li. Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2304–2314, 2019.

[89] Shunsuke Saito, Tomas Simon, Jason Saragih, and Hanbyul Joo. Pifuhd: Multi-level pixel-aligned implicit function for high-resolution 3d human digitization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 84–93, 2020.

[90] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. *Advances in Neural Information Processing Systems*, 33:20154–20166, 2020.

[91] Yawar Siddiqui, Justus Thies, Fangchang Ma, Qi Shan, Matthias Nießner, and Angela Dai. Texturify: Generating textures on 3d shape surfaces. *arXiv preprint arXiv:2204.02411*, 2022.

[92] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015.

[93] Casper Kaae Sønderby, Jose Caballero, Lucas Theis, Wenzhe Shi, and Ferenc Huszár. Amortised map inference for image super-resolution. *arXiv preprint arXiv:1610.04490*, 2016.

[94] Justus Thies, Michael Zollhöfer, and Matthias Nießner. Deferred neural rendering: Image synthesis using neural textures. *ACM Transactions on Graphics (TOG)*, 38(4):1–12, 2019.

[95] Dustin Tran, Rajesh Ranganath, and David Blei. Hierarchical implicit models and likelihood-free variational inference. *Advances in Neural Information Processing Systems*, 30, 2017.

[96] Treedy's. Treedy's website. `https://www.treedys.com`. Last accessed 13 September 2022.

[97] Twindom. Twindom website. `https://web.twindom.com`. Last accessed 13 September 2022.

[98] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T Barron, and Pratul P Srinivasan. Ref-nerf: Structured view-dependent appearance for neural radiance fields. *arXiv preprint arXiv:2112.03907*, 2021.

[99] Chung-Yi Weng, Brian Curless, Pratul P Srinivasan, Jonathan T Barron, and Ira Kemelmacher-Shlizerman. Humannerf: Free-viewpoint rendering of moving people from monocular video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16210–16220, 2022.

[100] Olivia Wiles, Georgia Gkioxari, Richard Szeliski, and Justin Johnson. Synsin: End-to-end view synthesis from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7467–7477, 2020.

[101] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. *Advances in neural information processing systems*, 29, 2016.

[102] Jiqing Wu, Zhiwu Huang, Janine Thoma, Dinesh Acharya, and Luc Van Gool. Wasserstein divergence for gans. In *Proceedings of the European conference on computer vision (ECCV)*, pages 653–668, 2018.

[103] Zongze Wu, Dani Lischinski, and Eli Shechtman. Stylespace analysis: Disentangled controls for stylegan image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12863–12872, 2021.

[104] Hongyi Xu, Thiemo Alldieck, and Cristian Sminchisescu. H-nerf: Neural radiance fields for rendering and temporal reconstruction of humans in motion. *Advances in Neural Information Processing Systems*, 34:14955–14966, 2021.

[105] Qiangeng Xu, Zexiang Xu, Julien Philip, Sai Bi, Zhixin Shu, Kalyan Sunkavalli, and Ulrich Neumann. Point-nerf: Point-based neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5438–5448, 2022.

[106] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. Plenoctrees for real-time rendering of neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5752–5761, 2021.

[107] Jianfeng Zhang, Zihang Jiang, Dingdong Yang, Hongyi Xu, Yichun Shi, Guoxian Song, Zhongcong Xu, Xinchao Wang, and Jiashi Feng. Avatargen: a 3d generative model for animatable human avatars. *arXiv preprint arXiv:2208.00561*, 2022.

[108] Zerong Zheng, Tao Yu, Yebin Liu, and Qionghai Dai. Pamir: Parametric model-conditioned implicit representation for image-based human reconstruction. *IEEE transactions on pattern analysis and machine intelligence*, 44(6):3170–3184, 2021.

[109] Peng Zhou, Lingxi Xie, Bingbing Ni, and Qi Tian. Cips-3d: A 3d-aware generator of gans based on conditionally-independent pixel synthesis. *arXiv preprint arXiv:2110.09788*, 2021.

[110] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

[111] Jun-Yan Zhu, Zhoutong Zhang, Chengkai Zhang, Jiajun Wu, Antonio Torralba, Josh Tenenbaum, and Bill Freeman. Visual object networks: Image generation with disentangled 3d representations. *Advances in neural information processing systems*, 31, 2018.