

PointAvatar: Deformable Point-based Head Avatars from Videos

Yufeng Zheng^{1,2} Wang Yifan³ Gordon Wetzstein³ Michael J. Black² Otmar Hilliges¹
¹ETH Zurich, ²Max Planck Institute for Intelligent Systems, ³Stanford University



Figure 1. **PointAvatar** learns lighting-disentangled point-based head avatars from a monocular RGB video captured by a smartphone.

Abstract

The ability to create realistic, animatable and relightable head avatars from casual video sequences would open up wide ranging applications in communication and entertainment. Current methods either build on explicit 3D morphable meshes (3DMM) or exploit neural implicit representations. The former are limited by fixed topology, while the latter are non-trivial to deform and inefficient to render. Furthermore, existing approaches entangle lighting in the color estimation, thus they are limited in re-rendering the avatar in new environments. In contrast, we propose PointAvatar, a deformable point-based representation that disentangles the source color into intrinsic albedo and normal-dependent shading. We demonstrate that PointAvatar bridges the gap between existing mesh- and implicit representations, combining high-quality geometry and appearance with topological flexibility, ease of deformation and rendering efficiency. We show that our method is able to generate animatable 3D avatars using monocular videos from multiple sources including hand-held smartphones, laptop webcams and internet videos, achieving state-of-the-art quality in challenging cases where previous methods fail, e.g., thin hair strands, while being significantly more efficient in training than competing methods.

1. Introduction

Personalized 3D avatars are crucial building blocks of the metaverse and beyond. Successful tools for creating avatars should enable easy data capture, efficient computation, and create a photo-realistic, animatable, and relightable 3D representation of the user. Unfortunately, existing approaches fall short of meeting these requirements.

Recent methods that create 3D avatars from videos either build on 3D morphable models (3DMMs) [25, 34] or leverage neural implicit representations [31–33]. The former allow for efficient rasterization and inherently generalize to unseen deformations, yet they cannot easily model individuals with eyeglasses or complex hairstyles, as template meshes suffer from a-priori fixed topologies and are limited to surface-like geometries. Recently, neural implicit representations have also been used to model 3D heads [4, 11, 15, 51]. While they outperform 3DMM-based methods in capturing hair strands and eyeglasses, they are significantly less efficient to train and render, since rendering a single pixel requires querying many points along the camera ray. Moreover, deforming implicit representations in a generalizable manner is non-trivial and existing approaches have to revert to an inefficient root-finding loop, which impacts training and testing time negatively [9, 24, 44, 51].

To address these issues, we propose PointAvatar, a novel avatar representation that uses point clouds to represent the canonical geometry and learns a continuous deformation field for animation. Specifically, we optimize an oriented

contact: yufeng.zheng@inf.ethz.ch
project page: <https://zhengyuf.github.io/pointavatar/>

	Efficient Rendering	Easy Animation	Flexible Topology	Thin Strands	Surface Geometry
Meshes	✓	✓	✗	✗	✓
Implicit Surfaces	✗	✗	✓	✗	✓
Volumetric NeRF	✗	✗	✓	✓	✗
Points (ours)	✓	✓	✓	✓	✓

Table 1. **PointAvatar** is efficient to render and deform which enables straightforward rendering of full images during training. It can also handle flexible topologies and thin structures and can reconstruct good surface normals in surface-like regions, *e.g.*, skin.

point cloud to represent the geometry of a subject in a canonical space. For animation, the learned deformation field maps the canonical points to the deformed space with learned blendshapes and skinning weights, given expression and pose parameters of a pretrained 3DMM. Compared to implicit representations, our point-based representation can be rendered efficiently with a standard differentiable rasterizer. Moreover, they can be deformed effectively using established techniques, *e.g.*, skinning. Compared to meshes, points are considerably more flexible and versatile. Besides the ability to conform the topology to model accessories such as eyeglasses, they can also represent complex volume-like structures such as fluffy hair. We summarize the advantages of our point-based representation in Tab. 1.

One strength of our method is the disentanglement of lighting effects. Given a monocular video captured in unconstrained lighting, we disentangle the apparent color into the intrinsic albedo and the normal-dependent shading; see Fig. 1. However, due to the discrete nature of points, accurately computing normals from point clouds is a challenging and costly task [5, 16, 28, 35], where the quality can deteriorate rapidly with noise, and insufficient or irregular sampling. Hence we propose two techniques to (a) robustly and accurately obtain normals from learned canonical points, and (b) consistently transform the point normals with the non-rigid surface deformation while preserving geometric details. For the former, we exploit the low-frequency bias of MLPs [36] and estimate the normals by fitting a smooth signed distance function (SDF) to the points; for the latter, we leverage the continuity of the deformation mapping and transform the normals analytically using the deformation’s Jacobian. The two techniques lead to high-quality normal estimation, which in turn propagates the rich geometric cues contained in shading to further improve the point geometry. With disentangled albedo and detailed normal directions, PointAvatar can be relit and rendered in novel scenes.

As demonstrated using various videos captured with DSLR, smartphone, laptop cameras, or obtained from the internet, the proposed representation combines the advantages of popular mesh and implicit representations, and surpasses both in many challenging scenarios. In summary, our contributions include:

1. We propose a novel representation for 3D animatable avatars based on an explicit canonical point cloud and continuous deformation, which shows state-of-the-art photo-realism while being considerably more efficient than existing implicit 3D avatar methods;
2. We disentangle the RGB color into a pose-agnostic albedo and a pose-dependent shading component, enabling relighting in novel scenes;
3. We demonstrate the advantage of our methods on a variety of subjects captured through various commodity cameras, showing superior results in challenging cases, *e.g.*, for voluminous curly hair and novel poses with large deformation.

Code will be made available for research purposes.

2. Related Work

Point clouds for neural rendering. Point-based neural rendering is gaining attention thanks to the flexibility and scalability of point representations [42]. Aliev *et al.* [1] introduce one of the first approaches to point-based neural rendering, in which each point is augmented with a neural feature. These features are projected to a set of multi-scale 2D feature maps using a standard point rasterizer, then converted to an RGB image. Recently, Point-NeRF [46] combines points with volumetric rendering, achieving perceptually high-quality results while being efficient to train. Both of the above methods obtain point geometry through off-the-shelf multi-view stereo methods and keep them fixed during optimization, other approaches [38, 49] propose to jointly optimize the point geometry using differentiable point rasterization [23, 45, 48]. Our method not only jointly optimizes both the point geometry and colors, but also the deformation of the point cloud through a forward deformation field proposed in [9, 51].

Head avatars from 2D. Learning photo-realistic head avatars from 2D observations is an emerging research topic in the computer vision community. Based on 3DMMs [25, 34], recent works [12, 21] leverage differentiable neural rendering to learn the detailed facial appearance and complete head geometry for a single subject. The idea is extended in [7, 20] to enable generative or one-shot head avatars. Another line of work leverages neural implicit representations. NerFace [11] extends photo-realistic and flexible neural radiance fields (NeRF) [32] to model the dynamic head geometry and view-dependent appearance. IMavatator [51] employs neural implicit surfaces [19, 31, 33, 47] and learns an implicit deformation field for generalizable animation. Implicit head avatars have been extended to a multi-subject scenario by [4, 15]. To the best of our knowledge, our work is the first to learn deformable point-based head avatars.

Point-based human avatars. PoP [30] learns to model pose-dependent clothing geometry by mapping points from

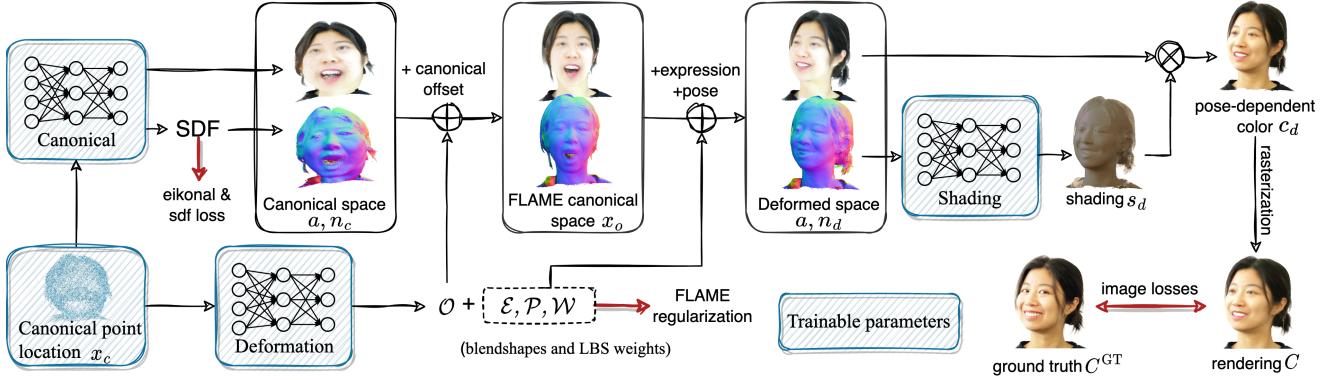


Figure 2. **Method pipeline.** We model the human head as a learned deformable point cloud consisting of the point locations \mathbf{x}_c , normals \mathbf{n}_c and albedo \mathbf{a} , which describe the subject’s geometry and intrinsic appearance in the canonical space. To deform points given a target expression and pose in a controllable and interpretable way, we warp \mathbf{x}_c into the FLAME canonical space via a learned offset \mathcal{O} , and then deform further to the target deformed space by applying the blendshape and skinning using learned personalized blendshape bases (\mathcal{E}, \mathcal{P}) and skinning weights (\mathcal{W}). After obtaining the deformed geometry, a small shading MLP is used to obtain shadings s_d from the deformed normals \mathbf{n}_d . These are multiplied with the albedo colors \mathbf{a} to produce the shaded colors c_d , which are rendered to images via differentiable rasterization. PointAvatar leverages a combination of per-pixel and image-based losses to improve photo-realism, while the FLAME regularization term encourages controllable and generalizable animations.

the minimally-clothed SMPL [27] surface to the clothing surface and demonstrates impressive geometric quality on various clothing types. Point-based clothing representation is extended by [26, 29] to mitigate point sparsity issues for long skirts. Our work is principally different in two ways: 1) We learn the deformation, geometry, and appearance jointly from scratch, without explicitly relying on a 3DMM template. 2) We learn from monocular videos, whereas previous works [26, 29, 30] require 3D scans.

3. Method

Given a monocular RGB video of a subject performing various expressions and poses, our model jointly learns (1) a point cloud representing the pose-agnostic geometry and appearance of the subject in a canonical space; (2) a deformation network that transforms the point cloud into new poses using FLAME expression and pose parameters extracted from the RGB frames; (3) a shading network that outputs a per-point shading vector based on the point normals in the deformed space. The three components can be jointly optimized by comparing the rendering of the shaded points in the deformed space with the input frames. Once trained, we can synthesize new sequences of the same subject under novel poses, expressions and lighting conditions. Fig. 2 represent an overview of our method.

3.1. Point-based Canonical Representation

Our canonical point representation consists of a set of learnable points $\mathcal{P}_c = \{\mathbf{x}_c^i\}$ with $i = \{1, 2, \dots, N\}$, where $\mathbf{x}_c \in \mathbb{R}^3$ denotes the optimizable point locations. We em-

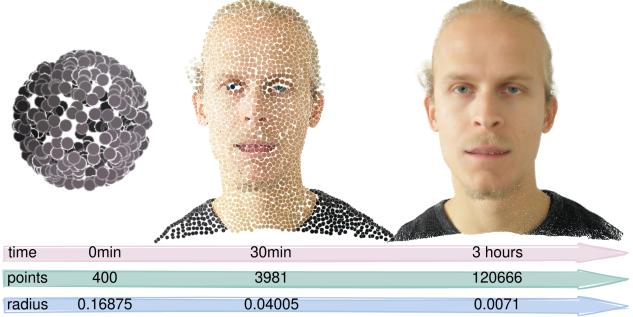


Figure 3. The **Coarse-to-fine** optimization strategy upsamples points and reduces point radii periodically during training, enabling fast convergence and detailed final reconstruction.

pirically choose to learn points in an unconstrained canonical space without enforcing them to correspond to a pre-defined pose, and found that this leads to better geometry quality (implementation explained in Sec. 3.2).

For optimization, we initialize with a sparse point cloud randomly sampled on a sphere and periodically upsample the point cloud while reducing the rendering radii. This coarse-to-fine optimization scheme enables fast convergence of training because the initial sparse points are efficient to deform and render, and can quickly approximate the coarse shape, whereas denser points in the late training stage lead to good reproduction of details. A schematic of this procedure is provided in Fig. 3. Furthermore, at the end of every epoch, we prune away invisible points that have

not been projected onto any pixels with visibility above a predefined threshold, which further accelerates training.

Our method disentangles the point colors into a pose-agnostic albedo component and a pose-dependent shading component (explained later in Sec. 3.3), where the shading component is inferred from the point normals in the deformed space. We first discuss how to compute the normals and the albedo in canonical space, followed by the learned transformation of points and normals into the deformed space (see Sec. 3.2).

Canonical normals. Point normals are local geometric descriptors that can be estimated from neighboring point locations. However, estimating normals in this way yields poor results, especially when the points are sparse, noisy and irregularly sampled, as is the case during early training. To alleviate this issue, we propose to estimate the point normals from an SDF defined by the canonical points. Normals are then defined as the spatial gradient of the SDF:

$$\mathbf{n}_c = \nabla_{\mathbf{x}_c} \text{SDF}(\mathbf{x}_c). \quad (1)$$

Specifically, we represent the SDF with an MLP and fit the zero-level set to the canonical point locations using a data term and an Eikonal regularizer, defined as:

$$\mathcal{L}_{\text{sdf}} = \|\text{SDF}(\mathbf{x}_c)\|^2 \text{ and } \mathcal{L}_{\text{eik}} = (\|\nabla_{\mathbf{x}_e} \text{SDF}(\mathbf{x}_e)\| - 1)^2, \quad (2)$$

where the Eikonal samples \mathbf{x}_e consist of the original and perturbed point locations [13]. We update the SDF using current point locations at each training step.

Canonical albedo. We use an MLP to map the point locations \mathbf{x}_c to the albedo colors $\mathbf{a} \in \mathbb{R}^3$, similar to [8]. Compared with directly modeling the albedo of each point as an individual per-point feature, the inductive bias of MLPs automatically enforces a local smoothness prior on albedo colors [2, 3, 14, 41]. For efficiency, we use a shared MLP to compute the canonical normals and albedo in practice, *i.e.*,

$$[\text{SDF}(\mathbf{x}_c); \mathbf{a}] = \text{MLP}_c(\mathbf{x}_c). \quad (3)$$

3.2. Point Deformation

In order to achieve controllable animations, PointAvatar is deformed using the same expression and pose parameters as FLAME [25], a parametric head model learned from 4D scans. Our deformation takes a two-step approach shown in Fig. 2. In the first stage, we warp the canonical points \mathbf{x}_c to its location in an intermediate pose \mathbf{x}_o , which corresponds to a predefined mouth-opened canonical pose of the FLAME template. In the second stage, we use the target FLAME expression and pose parameters to transform \mathbf{x}_o to the deformed space based on learned blendshapes and

skinning weights. Our experiments (see Sec. 4.4) empirically show that the proposed two-staged deformation helps to avoid bad local minima during optimization and yields more accurate geometries.

To improve expression fidelity and to account for accessories such as eyeglasses, we learn personalized deformation blendshapes and skinning weights, similar to IMavtar [51]. Specifically, a coordinate-based MLP is used to map each canonical point \mathbf{x}_c to (1) an offset $\mathcal{O} \in \mathbb{R}^3$, which translates \mathbf{x}_c to its corresponding location in the flame canonical space \mathbf{x}_o ; (2) n_e expression blendshapes and n_p pose blendshapes, denoted as $\mathcal{E} \in \mathbb{R}^{n_e \times 3}$ and $\mathcal{P} \in \mathbb{R}^{n_p \times 9 \times 3}$; (3) LBS weights $\mathcal{W} \in \mathbb{R}^{n_j}$ associated to the n_j bones. The point location in deformed space \mathbf{x}_d is then computed as:

$$\mathbf{x}_o = \mathbf{x}_c + \mathcal{O} \quad (4)$$

$$\mathbf{x}_d = \text{LBS}(\mathbf{x}_o + \mathbf{B}_P(\theta; \mathcal{P}) + \mathbf{B}_E(\psi; \mathcal{E}), \mathbf{J}(\psi), \theta, \mathcal{W}), \quad (5)$$

where LBS and \mathbf{J} define the standard skinning function and the joint regressor defined in FLAME, and \mathbf{B}_P and \mathbf{B}_E denote the linear combination of blendshapes, which outputs the additive pose and expression offsets from the animation coefficients θ and ψ and the blendshape bases \mathcal{P} and \mathcal{E} . Despite the similarity of this formulation to IMavtar [51], our forward deformation mapping only needs to be applied once in order to map canonical point locations x_c to the deformed space, and therefore does not need the computationally costly correspondence search of [51].

Normal deformation. The deformation mapping defined in Eq. (5) is differentiable w.r.t. the input point locations. This allows us to transform canonical normals analytically with the inverse of the deformation Jacobian

$$\mathbf{n}_d = l \mathbf{n}_c \left(\frac{\partial \mathbf{x}_d}{\partial \mathbf{x}_c} \right)^{-1}, \quad (6)$$

where l is a normalizing scalar to ensure the normals are of unit length. The formulation can be obtained via Taylor's theorem. Please check Supp. Mat. for the detailed proof.

3.3. Point Color

The color of each point in the deformed space \mathbf{c}_d is computed using a simple yet flexible shading model, which allows us to disentangle colors into a scene-specific and pose-dependent lighting component $\mathbf{s}_d \in \mathbb{R}^3$, and a scene- and pose-agnostic albedo component $\mathbf{a} \in \mathbb{R}^3$:

$$\mathbf{c}_d = \mathbf{s}_d \circ \mathbf{a}, \quad (7)$$

where \circ denotes the Hadamard product. This disentanglement allows us to render the points in new environments by changing the lighting component (see Sec. 4.2).

As we assume our input video is captured under a fixed lighting condition and camera position, we model the shading as a function of the deformed point normals \mathbf{n}_d . In practice, we approximate their relation using a shallow MLP:

$$\mathbf{s}_d = \text{MLP}_s(\mathbf{n}_d) \quad (8)$$

This parameterization of shading equates to directional light sources at infinite distance, while more complex lighting conditions can be incorporated, we find this simple model sufficient for our data.

3.4. Differentiable Point Rendering

One advantage of our point-based representation is that it can be rendered efficiently using rasterization. We adopt PyTorch3D’s [37] differentiable point renderer. It splats each point as a 2D circle with a uniform radius, arranges them in z-buffers, and finally composites the splats using alpha compositing. The alpha values are calculated as $\alpha = 1 - d^2/r^2$, where d is the distance from the point center to the pixel center and r represents the splatting radius. Then, the transmittance values are calculated as $T_i = \prod_{k=1}^{i-1} (1 - \alpha_k)$, where i denotes the index of the sorted points in the z-buffer. The point color values are integrated to produce the final pixel color:

$$\mathbf{c}_{\text{pix}} = \sum_i \alpha_i T_i \mathbf{c}_{d,i}. \quad (9)$$

3.5. Training Objectives

The efficiency of point-based rendering allows us to render the whole image at each training step. This makes it possible to apply perceptual losses on the whole image, whereas implicit-based avatar methods are often limited to per-pixel objectives. Specifically, we adopt VGG feature loss [17], defined as

$$\mathcal{L}_{\text{vgg}}(\mathbf{C}) = \|\mathbf{F}_{\text{vgg}}(\mathbf{C}) - \mathbf{F}_{\text{vgg}}(\mathbf{C}^{\text{GT}})\|, \quad (10)$$

where \mathbf{C} and \mathbf{C}^{GT} denote the predicted and ground truth image, and $\mathbf{F}_{\text{vgg}}(\cdot)$ calculates the features from the first four layers of a pre-trained VGG [40] network.

In addition to \mathcal{L}_{vgg} , we follow prior work and adopt the losses of [51]:

$$\begin{aligned} \mathcal{L}_{\text{RGB}} &= \|\mathbf{C} - \mathbf{C}^{\text{GT}}\|, \\ \mathcal{L}_{\text{flame}} &= \frac{1}{N} \sum_{i=1}^N (\lambda_e \|\mathcal{E}_i - \hat{\mathcal{E}}_i\|_2 + \lambda_p \|\mathcal{P}_i - \hat{\mathcal{P}}_i\|_2 + \lambda_w \|\mathcal{W}_i - \hat{\mathcal{W}}_i\|_2), \\ \mathcal{L}_{\text{mask}} &= \|\mathbf{M} - \mathbf{M}^{\text{GT}}\|. \end{aligned}$$

Here, \mathbf{M} and \mathbf{M}^{GT} denote the predicted and ground truth head mask, $\hat{\mathcal{E}}$, $\hat{\mathcal{P}}$ and $\hat{\mathcal{W}}$ are pseudo ground truth defined by the nearest FLAME vertex. The mask prediction at

	L1 ↓	LPIPS ↓	SSIM ↑	PSNR ↑
IMavatar [51]	0.033; 0.050	0.178; 0.261	0.874; 0.770	22.4; 18.7
NerFace [11]	0.030; 0.045	0.126; 0.187	0.877; 0.782	22.7; 19.6
Ours	0.028; 0.045	0.103; 0.165	0.884; 0.776	24.5; 21.0
NHA [12]	0.022; 0.029	0.086; 0.123	0.890; 0.837	25.7; 21.6
Ours (no cloth)	0.021; 0.024	0.084; 0.108	0.902; 0.847	27.0; 24.9

Table 2. **Quantitative comparison.** The first and second number in each cell represent scores for lab-capture sequences (IMavatar and NerFace datasets) and casual videos (smartphone, webcam and internet videos), respectively. PointAvatar outperforms implicit-based IMavatar and NerFace by a large margin in perceptual quality reflected by LPIPS [50]. Compared to 3DMM-based NHA, our method not only generates complete avatars with shoulders and clothing, but also performs better for the head region.

each pixel is obtained by $\mathbf{m}_{\text{pix}} = \sum_i \alpha_i \mathbf{T}_i$, and the ground truth mask obtained using an off-the-shelf foreground estimator [18],

Our total loss is

$$\mathcal{L} = \lambda_{\text{rgb}} \mathcal{L}_{\text{RGB}} + \lambda_{\text{mask}} \mathcal{L}_{\text{mask}} + \lambda_{\text{flame}} \mathcal{L}_{\text{flame}} + \lambda_{\text{vgg}} \mathcal{L}_{\text{vgg}}. \quad (11)$$

The loss weights can be found in Supp. Mat. together with other implementation details.

4. Experiments

Datasets. We compare our approach with state-of-the-art (SOTA) methods on 2 subjects from IMavatar [51] and 2 subjects from NerFace [11]. Additionally, we evaluate different methods on 1 subject collected from the internet, 4 subjects captured with hand-held smartphones, and 1 subject from a laptop webcam. These settings pose new challenges to avatar methods due to limited head pose variation, automatic exposure adjustment or low image resolution. In Supp. Mat., we evaluate reconstructed point geometry on a synthetic dataset rendered from the MakeHuman project [6]. For all subjects, we use the same face-tracking results for all comparing methods.

Baselines. We compare our method with three SOTA baselines, including (1) NerFace [11], which leverages dynamic neural radiance fields [32], (2) neural head avatar (NHA) [12] based on 3D morphable mesh models [25] (3) and IMavatar [51], which builds on neural implicit surfaces [47] and learnable blendshapes. Together with our method, which represents the head geometry with deformable point clouds, our experiments reveal the strength and weaknesses of each geometry representation in the scenario of head avatar reconstruction. Through our experiments, we demonstrate the efficiency, flexibility and photorealism achieved by point-based avatars.

4.1. Comparison with SOTA Methods

In Tab. 2, we quantitatively compare our point-based avatar method with SOTA baselines using conventional

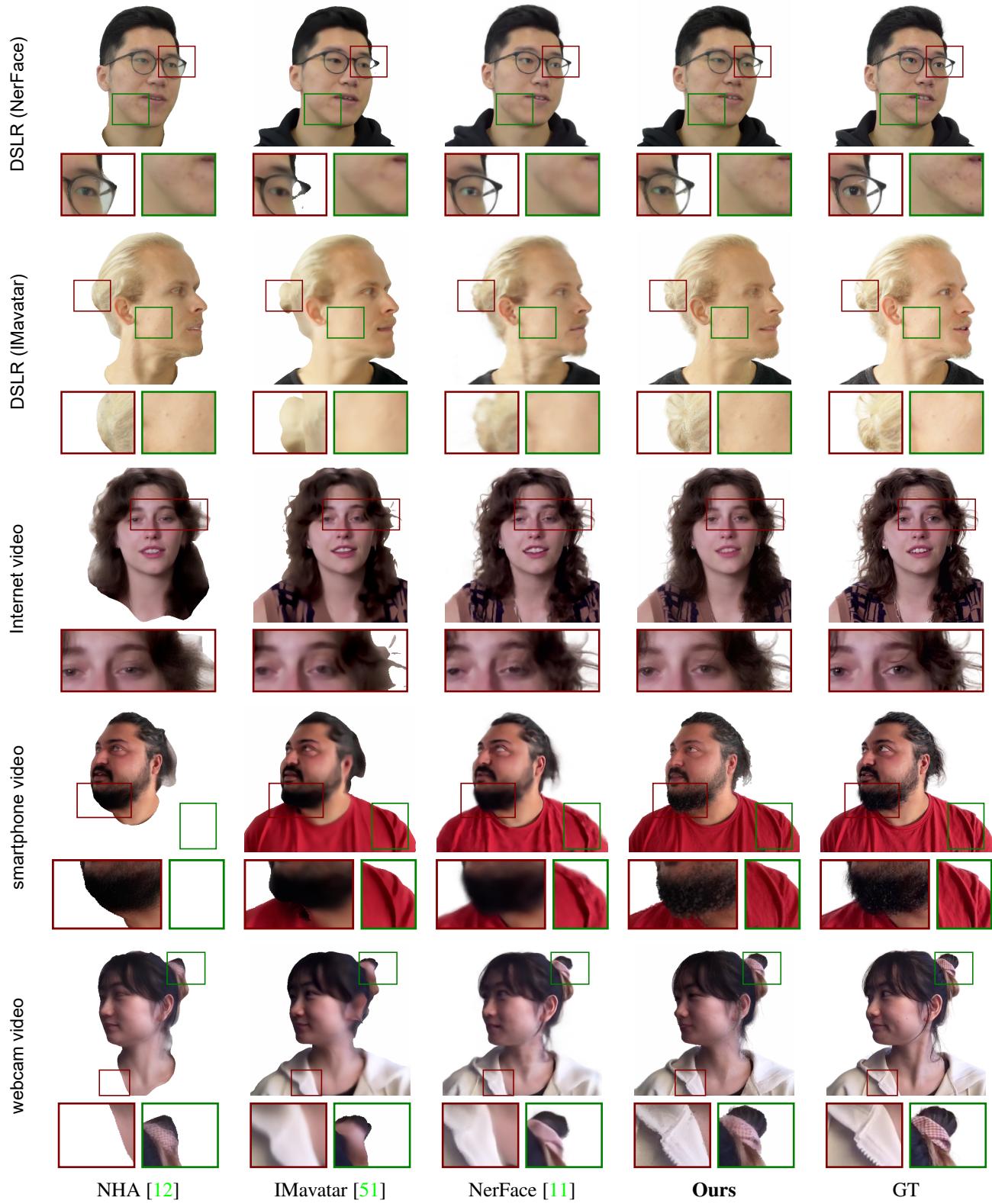


Figure 4. **Qualitative comparison.** PointAvatar produces photo-realistic and detailed appearance compared to SOTA methods, especially apparent in skin details and hair textures. Our point-based method is also flexible enough to capture challenging geometries such as eyeglasses and thin hair strands, which cannot be handled by mesh-based methods.

metrics including L1, LPIPS [50], SSIM and PSNR. Since NHA [12] only models the head region, we compare with NHA without considering the clothing region. PointAvatar achieves the best metrics among all methods on both lab-captured DSLR sequences and videos from more casual capture settings, *e.g.*, with smartphones.

Eye glasses and detailed structures. In Fig. 4, we show qualitatively that our method can successfully handle the non-face topology introduced by eye glasses, which pose severe challenges for 3DMM-based methods. NHA cannot model the empty space between the frame of the eye glasses because of its fixed topology and, instead, learns a convex hull painted with dynamic textures. In the second example of a man with a hair bun, even though the bun is topologically consistent with the template mesh, NHA still produces a worse hair silhouette than other methods. IMavatar, based on implicit surfaces, is theoretically capable of modeling the geometry of eyeglasses, but it also fails to learn part of the thin frame of glasses. Furthermore, for IMavatar to learn such thin structures, accurate foreground segmentation masks are required, which are hard to obtain in-the-wild. In Supp. Mat., we show that our method can be trained without mask supervision or known background.

Surface-like skin and volumetric hair. Both NHA and IMavatar enforce a surface constraint by design. While such constraint helps with surface geometry reconstruction and multi-view consistency, it also causes problems when modeling volumetric structures such as the curly hair in the third example. Our point-based method is free to model volumetric structures where needed, but also encourages surface-like point geometry in the skin and clothing regions via point pruning, which removes unseen inside points. We show that our method renders sharp and realistic images even for extreme head poses. In contrast, NerFace is a flexible volumetric NeRF [32]-based avatar method. While it is capable of modeling thin structures of any topology, its under-constrained nature leads to poor geometry and compromised rendering quality in uncommon poses. Even for near frontal poses shown in the first and third examples, our method still produces sharper skin details than NerFace.

4.2. Lighting Disentanglement

Our method disentangles rendered colors into intrinsic albedo colors and normal-dependent shading values, and can be faithfully relit as shown in Fig. 5. In the following, we demonstrate that lighting disentanglement, along with the proposed normal estimation techniques, improves the geometry details of the reconstruction.

In column 2 of Fig. 6, we show that, disentangling shading and albedo itself improves facial geometry (see the cheek area). We further ablate two of our design choices for obtaining better point normals. First, we compare our

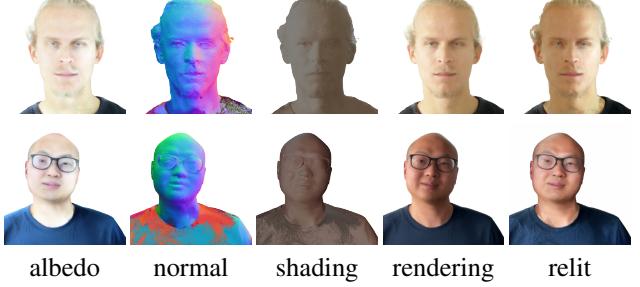


Figure 5. **Self-supervised lighting disentanglement.** PointAvatar disentangles albedo and normal-dependent shading from a single video captured with a fixed lighting condition. After training, PointAvatar can be faithfully relit.

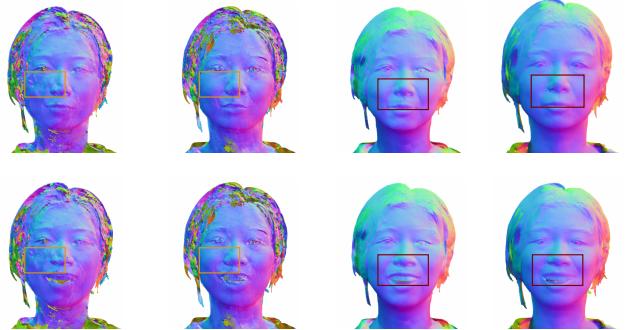


Figure 6. **Geometry ablation.** Disentangling shading and albedo improves facial normals compared to an entangled rendering model. Moreover, obtaining canonical point normals from SDF further improved smoothness. Transforming normals with the spatial Jacobian of the deformation field enables the capture of blendshape-related normal changes, *e.g.*, around the nasal line.

SDF-based normal estimation with direct normal estimation, which calculates normals by approximating the local plane of neighboring points. Normals obtained through SDF are less noisy, especially in highly detailed regions such as the hair, eyes and mouth, where direct normal estimation often fails. Second, we compare our proposed normal transformation using the deformation Jacobian vs. transforming the normals simply with the rotation matrix of the point deformation. The latter ignores the spatial changes of blendshapes and LBS weights. As column 4 shows, our normal transformation method takes into account the effects of additive blendshapes, and is able to produce correct normals around the nasal line when smiling.

4.3. Training Efficiency

In Tab. 3 and Fig. 7, we show that PointAvatar is considerably faster to train and render than implicit head avatar methods. Thanks to the coarse-to-fine learning strategy and point pruning, our method only needs to render a small

Method	Training time (hour)	Rendering time per image (train)
IMavatar	48h	100s
NerFace	54h	4s
Ours	11h	0.1s - 1.5s (varies with point numbers)

Table 3. **Training and rendering time.** Compared to implicit-based methods, our method is significantly faster in training and is able to render full-images much more efficiently.

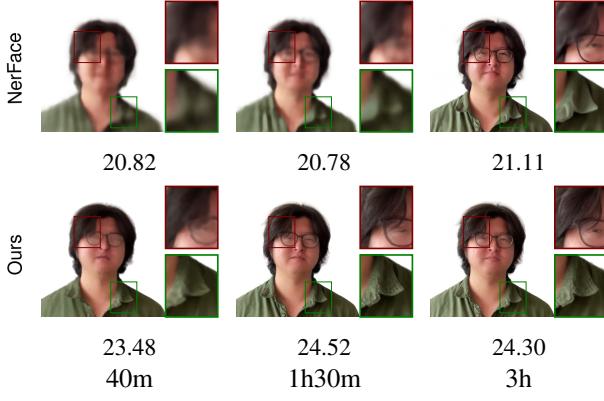


Figure 7. **Training efficiency.** PointAvatar converges much faster than implicit-based methods. Here we show NerFace [11] as an example, and indicate the average PSNR of the test set.

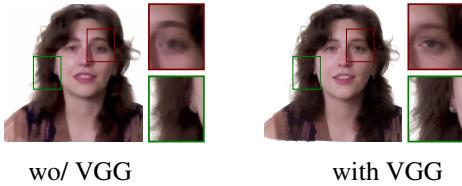


Figure 8. **Ablation:** VGG loss improves photo-realism compared to only using a per-pixel L1 loss. PointAvatar is able to render full images efficiently during training, enabling the usage of various image- and patch-based losses.

number of points with large radii in early training stages, which significantly speeds up training. With the ability to render full images efficiently during training, PointAvatar can be trivially combined with various image- and patch-based training objectives, which target certain properties, *e.g.*, photo-realism, much more effectively than pixel-based losses. In Fig. 8, we ablate our VGG feature loss, revealing its significant effect in boosting photo-realism.

4.4. Ablation: Free Model Canonical Space

As depicted in Fig. 9, having an additional freely-learned canonical space, compared to forcing the model to directly learn in a predefined FLAME space, improves the canonical point geometry and generalization to novel poses significantly. A possible reason is that the deformations, modeled by an MLP, are easier to optimize compared to point loca-



Figure 9. **Ablation: canonical offset** improves the canonical 3D geometry and therefore boosts generalization to novel head poses. For both cases, we show the canonical point representation (transformed to the FLAME canonical space if using canonical offset) and a deformed representation in a novel pose.

tions. Without the canonical offset \mathcal{O} , the model overfits and learns wrong canonical geometries.

5. Discussion

We propose PointAvatar, a deformable point-based avatar representation that features high flexibility, efficient rendering and straightforward deformation. We show that our method is able to handle various challenging cases in the task of head avatars modeling, including eye glasses, voluminous hair, skin details and extreme head poses. Furthermore, our method renders efficiently and can be trained significantly faster than previous methods. Despite trained only on a video with fixed lighting conditions, PointAvatar disentangles lighting effects from the intrinsic albedo, enabling realistic re-rendering in novel lighting conditions.

There are several exciting directions for future work. (1) Our shading MLP maps normal directions to shadings, assuming uniform reflectance properties for all points. Learning surface roughness could improve regional specular effects, *e.g.*, for eyes and teeth. (2) We render all points with uniform radius but some regions require denser and finer points to be modeled accurately, *e.g.*, eyes and hair strands. Rendering with different point sizes could potentially achieve detailed reconstructions with less points, and boost training and rendering efficiency even further. (3) We focus on the explainability of our method, but future works could combine PointAvatar with neural rendering to boost photo-realism. (4) Our method cannot faithfully model the reflection of eyeglass lenses. Future work could model transparencies and reflections [43] to improve this.

Acknowledgements Yufeng Zheng is supported by the Max Planck ETH Center for Learning Systems. Wang Yifan is partially funded by the SNF postdoc mobility fellowship. MJB has received research gift funds from Adobe, Intel, Nvidia, Meta/Facebook, and Amazon. MJB has financial interests in Amazon, Datagen Technologies, and Meshcapade GmbH. While MJB is a part-time employee of Meshcapade, his research was performed solely at, and funded solely by, the Max Planck Society.

Supplemental Materials

In this supplemental document, we provide the additional experiments mentioned in the main paper in Sec. 1, implementation details for our method in Sec. 2, the proof for Jacobian-based normal transformation in Sec. 3, and discussions about data capture ethics in Sec. 4. Additionally, we recommend checking out our supplemental video.

1. Additional Results

1.1. Results on the MakeHuman Dataset

We evaluate the reconstructed surface geometry on the MakeHuman synthetic dataset rendered from [6]. We show that PointAvatar is not only capable of capturing volumetric structures as shown in the main paper, but also performs on par with state-of-the-art methods on surface geometry reconstruction, as shown in Fig. 1 and Tab. 1.

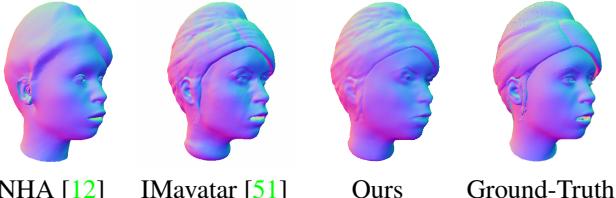


Figure 1. **Qualitative comparison of surface geometry.** Our point-based avatar representation reconstructs comparable head geometry to surface-only representations such as meshes (NHA) and implicit surfaces (IMavatator).

Method	female 1	female 2	male 1	male 2
NHA [12]	0.94	0.95	0.94	0.94
IMavatator [51]	0.961	0.966	0.954	0.955
Ours	0.954	0.954	0.944	0.958

Table 1. **Quantitative comparison on the MakeHuman dataset [6].** We report the normal consistency metric for evaluating reconstructed surface geometry. The scores for NHA [12] and IMavatator [51] are obtained from their papers.

We thank the authors of NHA [12] for kindly sharing this evaluation dataset with us.

1.2. Learning without Foreground Masks



Figure 2. **Self-supervised foreground-background disentanglement.** PointAvatar can be learned without mask supervision or known background.

It is beneficial to learn avatars without relying on foreground segmentation masks, because off-the-shelf segmentation networks often fail in in-the-wild scenarios. In Fig. 2, we show that PointAvatar is able to roughly disentangle foreground and background contents in a self-supervised manner. Note that NerFace [11] cannot be learned without known backgrounds.

2. Implementation Details

In this section, we provide implementation details regarding network architectures, training and evaluation procedures, and the preprocessing of videos. In addition, we will release our code for research purposes.

2.1. Network Architecture

We show the architecture of the canonical, deformation and shading MLPs in Fig. 3.

2.2. Training Details

Loss weights. We choose $\lambda_{\text{rgb}} = 1$, $\lambda_{\text{mask}} = 1$, $\lambda_{\text{flame}} = 1$, and $\lambda_{\text{vgg}} = 0.1$ for most of our experiments, except for the experiment shown in Fig. 6 (main paper), where we used $\lambda_{\text{rgb}} = 5$, and $\lambda_{\text{vgg}} = 0.05$, and the experiment on synthetic data (Sec. 1.1 in Supp. Mat.), where we used $\lambda_{\text{rgb}} = 10$, and $\lambda_{\text{vgg}} = 0$. For the flame loss, we choose $\lambda_e = 1000$, $\lambda_p = 1000$, and $\lambda_w = 1$. For optimizing the canonical SDF, we use $\lambda_{\text{SDF}} = 1$, and $\lambda_{\text{eik}} = 0.1$. We leverage an Adam optimizer [22] for training, with a learning rate of $\eta = 1e^{-4}$, and $\beta = (0.9, 0.999)$.

Point upsampling and pruning. Every 5 epochs, we double the number of points and reduce the point rasterization radii, which allows our method to capture finer details. The new point locations are obtained by perturbing the current points with random noises. The factor for radius reduction is carefully chosen: Assuming that points roughly form a surface, the radius should be decreased by a factor of $1/\sqrt{2}$. We choose to reduce the point radii by a factor of 0.75 in practice. Additionally, after each epoch, we prune

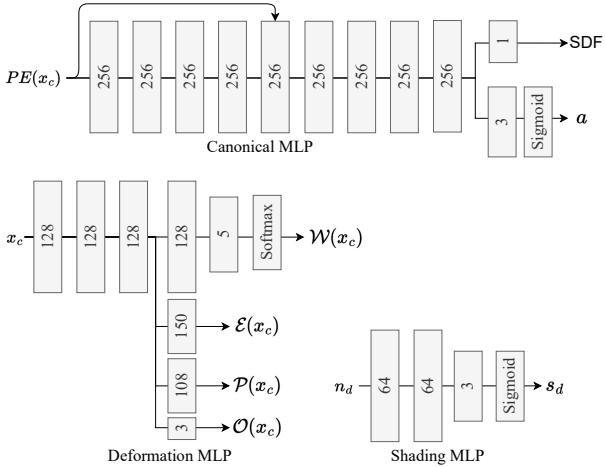


Figure 3. Network architecture. We show the network architectures of the canonical, deformation and shading MLPs. Except from the last layer, each linear layer is followed by weight normalization [39] and non-linear activation. We use the Softplus [10] function for the canonical and deformation MLP, and the ReLU activation for the shading MLP.

away points that are invisible, *i.e.*, not projected onto any pixels with compositing weights larger than a threshold. We choose 0.5 as the pruning threshold.

2.3. Evaluation Details

FLAME [25] parameter optimization. Similar to NHA [12], our method also fine-tunes pre-tracked FLAME expression and pose parameters during training and evaluation. For training, the loss weights are elaborated in the main paper and in Sec. 2.2 of Supp. Mat. For evaluation, we only use the RGB loss.

Hole filling. Our point-based avatar can suffer from sparsity issues under large deformations. While more advanced hole-filling techniques can be applied, we leverage a simple trick by applying erosion and dilation to the rendered images. We use a 3×3 kernel in our experiments.

Relighting. After training, there are two options to perform relighting. The first option manipulates the normal directions (input of the shading network) via transformations, *e.g.*, flipping or rotation. See Fig. 5 in the main paper for examples. However, since our method is only trained on one single video, the shading network cannot represent arbitrary lighting conditions. To this end, the second option discards the shading network and instead renders avatars with standard shading models using the learned albedo and normals. This option allows more flexibility and can be used to re-light avatars in novel environments. In the supplemental

video, we show relighting results obtained with the second option, using a simple diffused-only shading model.

2.4. Data Preprocessing

We leverage the preprocessing pipeline of IMAvatar [51] and use the same camera and FLAME parameters for all methods. This allows us to compare head avatar methods without considering the influence of different face-tracking schemes used for preprocessing.

3. Proof of Eq. 6 (deformed normal formula)

First, we recap Eq. 6 from the main paper, which states that deformed normals can be obtained via the constraint:

$$\mathbf{n}_d \cdot (\mathbf{x}_{d,i} - \mathbf{x}_d) = 0,$$

which needs to be satisfied for every neighboring point $\mathbf{x}_{d,i}$. Note that $\mathbf{x}_{d,i}$ can be linearly approximated as

$$\mathbf{x}_{d,i} = \mathbf{x}_d + \frac{\partial \mathbf{x}_d}{\partial \mathbf{x}_c} (\mathbf{x}_{c,i} - \mathbf{x}_c),$$

which allows us to rewrite the constraint as

$$\mathbf{n}_d \cdot \frac{\partial \mathbf{x}_d}{\partial \mathbf{x}_c} (\mathbf{x}_{c,i} - \mathbf{x}_c) = 0.$$

Note that the canonical normal \mathbf{n}_c satisfies

$$\mathbf{n}_c \cdot (\mathbf{x}_{c,i} - \mathbf{x}_c) = 0.$$

Therefore, in order to satisfy the constraint for all neighboring points $\mathbf{x}_{d,i}$, \mathbf{n}_d must satisfy

$$\mathbf{n}_d \cdot \frac{\partial \mathbf{x}_d}{\partial \mathbf{x}_c} = l \mathbf{n}_c,$$

where l is a normalizing scalar to ensure unit normal length. This leads to the formulation in Eq. 6.

4. Ethics

We captured 5 human subjects with smartphones or laptop cameras for our experiments. All subjects have given written consents for using the captured images in this project. We will make the data publicly available for research purposes, where permission for data publishing is given by the subjects.

Our method could be extended to generate media content of real people performing synthetic poses and expressions. We do not condone using our work to generate fake images or videos of any person with the intent of spreading misinformation or tarnishing their reputation.

References

- [1] Kara-Ali Aliiev, 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](#)
- [2] Jonathan T Barron and Jitendra Malik. High-frequency shape and albedo from shading using natural image statistics. In *CVPR 2011*, pages 2521–2528. IEEE, 2011. [4](#)
- [3] Jonathan T Barron and Jitendra Malik. Shape, albedo, and illumination from a single image of an unknown object. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 334–341. IEEE, 2012. [4](#)
- [4] Alexander W. Bergman, Petr Kellnhofer, Wang Yifan, Eric R. Chan, David B. Lindell, and Gordon Wetzstein. Generative neural articulated radiance fields. In *arXiv:2206.14314*, 2022. [1, 2](#)
- [5] Alexandre Boulch and Renaud Marlet. Fast and robust normal estimation for point clouds with sharp features. In *Computer graphics forum*, volume 31, pages 1765–1774. Wiley Online Library, 2012. [2](#)
- [6] Leyde Briceno and Gunther Paul. *MakeHuman: A Review of the Modelling Framework: Volume V: Human Simulation and Virtual Environments, Work With Computing Systems (WWCS), Process Control*. 2019. [5, 1](#)
- [7] Marcel C. Buehler, Abhimitra Meka, Gengyan Li, Thabo Beeler, and Otmar Hilliges. Varitex: Variational neural face textures. In *International Conference on Computer Vision (ICCV)*, 2021. [2](#)
- [8] Pol Caselles, Eduard Ramon, Jaime Garcia, Xavier Giro-i Nieto, Francesc Moreno-Noguer, and Gil Triginer. Sira: Relightable avatars from a single image. *arXiv preprint arXiv:2209.03027*, 2022. [4](#)
- [9] Xu Chen, Yufeng Zheng, Michael J Black, Otmar Hilliges, and Andreas Geiger. Snarf: Differentiable forward skinning for animating non-rigid neural implicit shapes. In *International Conference on Computer Vision (ICCV)*, 2021. [1, 2](#)
- [10] Charles Dugas, Yoshua Bengio, François Bélisle, Claude Nadeau, and René Garcia. Incorporating second-order functional knowledge for better option pricing. In T. Leen, T. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems*, volume 13. MIT Press, 2001. [2](#)
- [11] Guy Gafni, Justus Thies, Michael Zollhöfer, and Matthias Nießner. Dynamic neural radiance fields for monocular 4d facial avatar reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8649–8658, June 2021. [1, 2, 5, 6, 8](#)
- [12] Philip-William Grassal, Malte Prinzler, Titus Leistner, Carsten Rother, Matthias Nießner, and Justus Thies. Neural head avatars from monocular rgb videos. *arXiv preprint arXiv:2112.01554*, 2021. [2, 5, 6, 7, 1](#)
- [13] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit geometric regularization for learning shapes. In *Proceedings of Machine Learning and Systems 2020*, pages 3569–3579. 2020. [4](#)
- [14] Roger Grosse, Micah K Johnson, Edward H Adelson, and William T Freeman. Ground truth dataset and baseline evaluations for intrinsic image algorithms. In *2009 IEEE 12th International Conference on Computer Vision*, pages 2335–2342. IEEE, 2009. [4](#)
- [15] Yang Hong, Bo Peng, Haiyao Xiao, Ligang Liu, and Juyong Zhang. Headnerf: A real-time nerf-based parametric head model. 2022. [1, 2](#)
- [16] Hui Huang, Dan Li, Hao Zhang, Uri Ascher, and Daniel Cohen-Or. Consolidation of unorganized point clouds for surface reconstruction. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia 2009)*, 28:176:1–176:7, 2009. [2](#)
- [17] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, 2016. [5](#)
- [18] Zhanghan Ke, Jiayu Sun, Kaican Li, Qiong Yan, and Rynson W.H. Lau. Modnet: Real-time trimap-free portrait matting via objective decomposition. In *AAAI*, 2022. [5](#)
- [19] Petr Kellnhofer, Lars Jebe, Andrew Jones, Ryan Spicer, Kari Pulli, and Gordon Wetzstein. Neural lumigraph rendering. In *CVPR*, 2021. [2](#)
- [20] Taras Khakhulin, Vanessa Sklyarova, Victor Lempitsky, and Egor Zakharov. Realistic one-shot mesh-based head avatars. In *European Conference of Computer vision (ECCV)*, 2022. [2](#)
- [21] Hyeongwoo Kim, Pablo Garrido, Ayush Tewari, Weipeng Xu, Justus Thies, Matthias Nießner, Patrick Pérez, Christian Richardt, Michael Zollöfer, and Christian Theobalt. Deep video portraits. *ACM Transactions on Graphics (TOG)*, 37(4):163, 2018. [2](#)
- [22] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. [1](#)
- [23] 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. [2](#)
- [24] Ruilong Li, Julian Tanke, Minh Vo, Michael Zollhofer, Jürgen Gall, Angjoo Kanazawa, and Christoph Lassner. Tava: Template-free animatable volumetric actors. 2022. [1](#)
- [25] Tianye Li, Timo Bolkart, Michael. J. Black, Hao Li, and Javier Romero. Learning a model of facial shape and expression from 4D scans. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*, 36(6):194:1–194:17, 2017. [1, 2, 4, 5](#)
- [26] Siyou Lin, Hongwen Zhang, Zerong Zheng, Ruizhi Shao, and Yebin Liu. Learning implicit templates for point-based clothed human modeling. In *ECCV*, 2022. [3](#)
- [27] 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, Oct. 2015. [3](#)
- [28] Dening Lu, Xuequan Lu, Yangxing Sun, and Jun Wang. Deep feature-preserving normal estimation for point cloud filtering. *Computer-Aided Design*, 125:102860, 2020. [2](#)
- [29] Qianli Ma, Jinlong Yang, Michael J. Black, and Siyu Tang. Neural point-based shape modeling of humans in challenging clothing. In *2022 International Conference on 3D Vision (3DV)*, 2022. [3](#)

- [30] Qianli Ma, Jinlong Yang, Siyu Tang, and Michael J. Black. The power of points for modeling humans in clothing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2021. 2, 3
- [31] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4460–4470, 2019. 1, 2
- [32] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 1, 2, 5, 7
- [33] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 165–174, 2019. 1, 2
- [34] Pascal Paysan, Reinhard Knothe, Brian Amberg, Sami Romdhani, and Thomas Vetter. A 3d face model for pose and illumination invariant face recognition. In *2009 sixth IEEE international conference on advanced video and signal based surveillance*, pages 296–301. Ieee, 2009. 1, 2
- [35] Xiaojuan Qi, Renjie Liao, Zhengze Liu, Raquel Urtasun, and Jiaya Jia. Geonet: Geometric neural network for joint depth and surface normal estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 283–291, 2018. 2
- [36] Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred A. Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. 2018. 2
- [37] 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. 5
- [38] Darius Rückert, Linus Franke, and Marc Stamminger. Adop: Approximate differentiable one-pixel point rendering. *ACM Transactions on Graphics (TOG)*, 41(4):1–14, 2022. 2
- [39] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen, and Xi Chen. Improved techniques for training gans. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. 2
- [40] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 5
- [41] Marshall Tappen, William Freeman, and Edward Adelson. Recovering intrinsic images from a single image. *Advances in neural information processing systems*, 15, 2002. 4
- [42] A. Tewari, J. Thies, B. Mildenhall, P. Srinivasan, E. Tretschk, W. Yifan, C. Lassner, V. Sitzmann, R. Martin-Brualla, S. Lombardi, T. Simon, C. Theobalt, M. Nießner, J. T. Barron, G. Wetzstein, M. Zollhöfer, and V. Golyanik. Advances in neural rendering. *Computer Graphics Forum*, 41(2):703–735, 2022. 2
- [43] 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. *CVPR*, 2022. 8
- [44] Shaofei Wang, Katja Schwarz, Andreas Geiger, and Siyu Tang. Arah: Animatable volume rendering of articulated human sdbs. In *European Conference on Computer Vision*, 2022. 1
- [45] 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. 2
- [46] Qiangeng Xu, Zexiang Xu, Julien Philip, Sai Bi, Zhixin Shu, Kalyan Sunkavalli, and Ulrich Neumann. Pointnerf: Point-based neural radiance fields. *arXiv preprint arXiv:2201.08845*, 2022. 2
- [47] Lior Yariv, Yoni Kasten, Dror Moran, Meirav Galun, Matan Atzmon, Basri Ronen, and Yaron Lipman. Multiview neural surface reconstruction by disentangling geometry and appearance. *Advances in Neural Information Processing Systems*, 33, 2020. 2, 5
- [48] Wang Yifan, Felice Serena, Shihao Wu, Cengiz Öztïreli, and Olga Sorkine-Hornung. Differentiable surface splatting for point-based geometry processing. *ACM Transactions on Graphics (TOG)*, 38(6):1–14, 2019. 2
- [49] Qiang Zhang, Seung-Hwan Baek, Szymon Rusinkiewicz, and Felix Heide. Differentiable point-based radiance fields for efficient view synthesis. *arXiv preprint arXiv:2205.14330*, 2022. 2
- [50] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 5, 7
- [51] Yufeng Zheng, Victoria Fernández Abrevaya, Marcel C. Bühler, Xu Chen, Michael J. Black, and Otmar Hilliges. I M Avatar: Implicit morphable head avatars from videos. In *Computer Vision and Pattern Recognition (CVPR)*, 2022. 1, 2, 4, 5, 6