SAgA-NeRF: Subject-Agnostic and Animatable Neural Radiance Fields for Human Avatar

by

Jamal Ahmed Rahim

BEng, The University of Hong Kong, 2019

Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

in the
School of Computing Science
Faculty of Applied Sciences

SIMON FRASER UNIVERSITY
Summer 2022

Declaration of Committee

Name:	**Jamal Ahmed Rahim**
Degree:	**Master of Science**
Thesis title:	**SAgA-NeRF: Subject-Agnostic and Animatable Neural Radiance Fields for Human Avatar**
Committee:	**Chair:** Jiangchuan Liu Professor, Computing Science
	Ping Tan Supervisor Adjunct Professor, Computing Science
	Yasutaka Furukawa Committee Member Associate Professor, Computing Science
	Ali Mahdavi-Amiri Examiner Assistant Professor, Computing Science

Abstract

The advancement of Neural Radiance Fields (NeRF) has largely boosted the visual quality of human avatar constructed from RGB inputs. However, existing works either need per-subject training, and thus, cannot generalize to novel subjects (i.e. subject-specific), or can only reproduce the seen human poses contained within the inputs and cannot render novel poses (i.e. non-animatable). To this end, we propose the Subject-Agnostic and Animatable Neural Radiance Fields (SAgA-NeRF) for human avatar modeling from sparse-view videos, which can generalize to novel subjects and poses at the same time. To handle challenges posed by the task, we propose two main techniques, namely pose-based input frame selection, and a novel feature fusion on a parametric human body model. We compare SAgA-NeRF with existing subject-agnostic or animatable works, and show comparable results for both seen and novel poses. We also justify our design choices by showing that our proposed components outperform naive baselines.

Keywords: 3D human body modeling; 3D feature learning; Differentiable rendering; Neural Radiance Fields

Dedication

To my parents, grandparents, and siblings.

Acknowledgements

In the name of Allah, most Merciful and Compassionate. I would like to begin by praising Allah, to Whom all praise belongs. Only through His will, was I given the strength, knowledge, ability, and opportunity to undertake and complete this graduate program and thesis. These are, but a fraction, of His innumerable blessings.

I would also like to express my gratitude and sincere thanks to my senior supervisor, Dr. Ping Tan, for guiding me through my time at SFU, and making sure I prioritize my well-being over all else during these challenging times of the pandemic.

I am also extremely grateful to my co-author, Ziqian Bai, without whom this research would not have made such great progress; and to other co-authors, Zhaopeng Cui, Boming Zhao, Yinda Zhang, and Boxin Xi.

Most importantly, I am indebted to my family - my parents, grandparents, siblings, and uncles, for their unconditional love and support, without which, this endeavour would not have been possible.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Neural Rendering

Tewari et al. [39] define Neural Rendering as, "Deep image or video generation approaches that enable explicit or implicit control of scene properties such as illumination, camera parameters, pose, geometry, appearance, and semantic structure." Recently, Neural Rendering has made huge leaps in the field of novel view synthesis, which is the problem of rendering a scene from a novel camera view given some images of the scene. Classically, novel view synthesis has been performed using Image-based rendering (IBR) methods [9, 5]. These are optimization-based methods that perform 3D reconstruction of the scene and warp the observations onto the novel view image plane. Recently, Neural Rendering methods have been proposed for novel view synthesis and have shown better results compared to classical methods. Neural Rendering methods combine classic computer graphics with learnable components in the form of neural networks. Recently, Neural Radiance Fields (NeRF) [26], a Neural Rendering method, has achieved impressive results on photo-realistic novel view synthesis, and has sparked a lot of interest in Neural Rendering, with researchers proposing adaptations for solving various related problems. In this section, we will look into NeRF in detail, as well as some general trends of NeRF-related research.

As a direct prelude to NeRF, Neural Rendering methods had been proposed to define an implicit surface representation. Occupancy Networks [25] was proposed to predict binary occupancy given a 3D point co-ordinate and object encoding. IM-Net [7] was similarly proposed to predict binary occupancy, and also showed that the model could also be used to generate new shapes. DeepSDF [27] was proposed to regress a signed distance function instead of binary occupancy, and took in as input, similar to IM-Net and Occupancy Networks, a latent code and 3D point to predict the signed distance. The signed distance would be positive if the point is outside the shape, and negative if inside, and therefore, the shape's surface would be the zero-level-set of the learned function. Pixel-Aligned Implicit Function for High-Resolution Clothed Human Digitization (PIFu) [34] showed that Neural Rendering methods could learn highly detailed implicit models. This was achieved by gener-

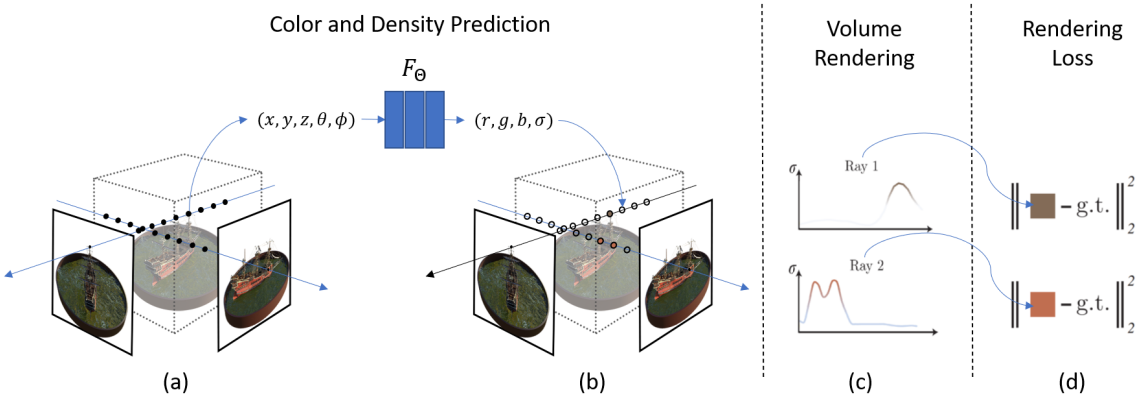

Figure 1.1: **Overview of NeRF [26] method.** (a) 3D points are sampled along camera rays from the target camera view. The coordinates and view direction are used as 5D inputs to the MLP, which (b) predicts color and density. (c) Differentiable volumetric rendering is used to calculate pixel color for each ray, and (d) the pixel color compared with corresponding pixel from the ground truth image to calculate loss and optimize.

ating a 3D-encoding for the input image (such that each pixel has a feature representation), and then using the pixel-aligned feature along with different depth values to predict binary occupancy for the human model. Another encoder, conditioned with the output from the previous geometry feature encoder, is used to generate another set of pixel-aligned feature representations to then also predict the RGB values at different depths, so that the model can be reconstructed with texture. All these methods have a common major disadvantage, that they need ground truth meshes in order to be trained. This is because rendering an image from the 3D representations learned by these methods requires a ray marching algorithm, and the algorithm used by these methods is the Marching Cubes [23] method. Since the Marching Cubes method is not differentiable, the gradient of the rendered value with respect to the input image can not be calculated through back-propagation. Hence, these methods can not be trained directly on the images, and the ground truth meshes are required so that the loss can be back-propagated after the occupancy/SDF has been predicted.

NeRF [26] was then proposed by Mildenhall et al. which uses principles from classical volume rendering [16] to render the color of any ray passing through the scene. The formulation of volume rendering used by NeRF is differentiable, and therefore, can be trained directly on images, no longer requiring ground truth meshes. This work sparked huge interest in Neural Rendering methods as it showed excellent results and potential, despite it's simplicity. Figure 1.1 shows a simplified overview of the NeRF method, and we discuss some details below.

NeRF proposed using a 5D-vector valued function to represent a continuous scene. Given a 3D point $\mathbf{x} = (x, y, z)$ and the 2D camera-viewing direction $\mathbf{v} = (\theta, \phi)$, NeRF outputs an RGB color $\mathbf{c} = (r, g, b)$ and volume density σ at that 3D co-ordinate. A Multilayer

Perceptron (MLP) F_Θ is used to learn and approximate the 5D-vector valued function such that $F_\Theta(\mathbf{x}, \mathbf{v}) = (\mathbf{c}, \sigma)$. NeRF uses some simple yet effective tricks. The density is predicted only as a function of the input co-ordinate $\mathbf{x}$, whereas the color $\mathbf{c}$ is predicted using both, the co-ordinate $\mathbf{x}$ and viewing direction $\mathbf{v}$. This helps keep the learned representation multi-view consistent while allowing for view-dependent specularities. Furthermore, positional encoding is used to map the inputs $xyz\theta\phi$ to a higher frequency so as to avoid bias towards learning lower frequency functions. Finally, once color and density have been predicted, volume rendering for each camera ray is used to calculate color of the pixel using the predicted color and density of sampled points along the ray. The output pixel color can be compared with the pixel color from the ground truth image to calculate the loss and optimize the learned representation.

NeRF has since been adapted to solve various related problems, as well as to solve some of the disadvantages of vanilla-NeRF. Some disadvantages are that it is slow in training as well as rendering, it can only represent static scenes, and it has to be trained per-scene (not generalizable to other scenes).

To speed up NeRF, some of the following works were proposed:

- KiloNeRF [33], which uses thousands of tiny MLPs to represent parts of scene so that the model does not have to query a deep MLP millions of times.
- Automatic Integration [19], which introduces learning volume integral to speed up rendering.
- Learned Initializations for Optimizing Coordinate-Based Neural Representations [38], which speeds up training by using learned initializations.

Some works were proposed to model dynamic scenes. Nerfies [28] and D-NeRF [31] use a second MLP to apply a deformation.

Generalizable NeRF methods were also proposed. IBRNet [41] is one of the most important of such methods. For a 3D query point, it aggregates features from nearby camera views from the given image set. This is done by first projecting the point onto the selected views and extracting pixel-aligned features, image view direction, and pixel color, and then using an MLP to aggregate the information and predict a color and density feature. The density features of all points along a ray are then fed into a ray transformer module to predict the densities for each sample.

We use a NeRF-based approach for our goal of rendering, from a target camera view, an arbitrary human in an arbitrary human pose. We first discuss works done in the field of NeRF-based human avatar rendering, their formulations, and their advantages and disadvantages in Section 1.3. We then discuss the motivation for our work and define the problem further in Section 1.4.

1.2 Parametric Human Body Model: SMPL

In order to make this thesis self contained, before discussing NeRF-based human avatar rendering, it is important to discuss parametric human body models, as they greatly help with such methods. This is because rendering human body is a very specific task, and not as general as rendering any scene, and therefore a model with fixed topology can help create a more tractable formulation of the problem as well as boost learning, since it would help introduce strong priors. Furthermore, most human avatar rendering methods focus on making the model animatable. This extra goal would mean introducing an input to the formulation that explains the desired target human pose, and therefore, using a parametric human body model that takes into account human pose as a parameter can greatly simplify the problem formulation.

We will specifically discuss the work on SMPL: A skinned multi-person linear model [22], which is the moset widely used parametric human body model in the computer vision community. The idea of SMPL is to have a template human body model, that can deform to form a model that represents some human in some pose. SMPL uses a mesh representation with fixed topology. The motivation behind SMPL is to have human body models that are fast to render, easy to deploy, and compatible with existing rendering engines. SMPL is a learned body model, and learns from data: the rest pose template $\mathbf{T}$, blend weights $\mathcal{W}$, pose-dependent blend shapes $\mathcal{P}$, identity-dependent blend shapes $\mathcal{S}$, and a regressor $\mathcal{J}$ from vertices to joint locations. SMPL is a statistical parametric human body model that uses two parameters to represent and encode a body model:

1. Shape parameter β, which is a 10-D scalar valued vector, that encodes the shape of the person.

2. Pose parameter, which is a $23 \times 3 + 3 = 72$-D vector. It denotes in axis-angle representation, the relative rotation of each of the 23 joints with respect to it's parent joint in the kinematic tree (23×3) as well as the root orientation ($+3$).

A model of some person with shape parameters β in some pose with pose parameters θ is constructed in three stages. For our discussion and purposes, an overview of these stages without calculation details is more than sufficient. The stages are:

1. Deformation of template model $\mathbf{T}$ in rest pose to model of person in rest pose T_i using the shape parameters of the person β, and learned identity-dependent blend shapes $\mathcal{S}$ done by a linear function B_S. Regression of joints location J of the person in rest pose is also performed at this stage using learned regressor $\mathcal{J}$:

$$T_i = \mathbf{T} + B_S(\beta; \mathcal{S}) \tag{1.1}$$

$$J = \mathcal{J}(T_i) \tag{1.2}$$

2. Further deformation of T_i to explain for the pose-dependent deformations (such as soft-tissue motion). This is done using linear function B_P given learned pose-dependent blend shapes $\mathcal{P}$, and gives us the model of the person T_p in rest pose with pose-dependent deformations:

$$T_p = T_i + B_p(\theta; \mathcal{P}) \tag{1.3}$$

3. Lastly, the joints are transformed according to the pose θ, and all the vertices of the model are transformed accordingly. This is done by using learned blend weights $\mathcal{W}$, which explain how much each vertex is affected by the movement of each joint, so that a weighted sum of the transformation of each joint can be calculated for each vertex. This is done by function W to give the final model M:

$$M = W(T_p, J, \theta; \mathcal{W}) \tag{1.4}$$

The SMPL model has shown to be very useful for NeRF-based human avatar rendering as shown by works discussed in Section 1.3.

1.3 NeRF-Based Human Rendering

Novel view synthesis of human avatars has mostly been done using traditional techniques that rely on using complicated equipment such as depth sensors [8, 10] or a dense array of cameras [36, 13] to perform reconstruction of the human body, making the process not widely applicable. We narrow the scope of the problem to image-based rendering, where the goal is to perform novel view synthesis given a fixed set of multi-view images to help widen the scope of application.

Recently, great progress has been made on using deep neural networks (DNNs) to render photo-realistic images by works such as NeRF [26] and IBR-Net [41] among many others, as discussed in Section 1.1. However, these methods are designed for generic scenes, and therefore don't leverage strong human priors. Pixel-Aligned Implicit Function for High-Resolution Clothed Human Digitization (PIFu) [34] is a Neural Rendering method (discussed in Section 1.1) that shows detailed rendering of human subjects, but this work also does not leverage human priors and only performs novel view synthesis for a static human pose, which has little practical application. A lot of methods build on top of NeRF but specialize to human subjects so as to render human subjects, and most of them use SMPL [22] as a strong human prior to aid in their learning, as well as to make animation of the human model tractable. Below, we will discuss two categories of work done in this field: animatable rendering, and subject agnostic rendering.

1.3.1 Animatable Rendering

The goal of animatable rendering is, given:

1. a fixed set of N_t multi-view images of a human, $\{I_t^c | c = 1, 2, ..., N_c, t = 1, 2, ..., N_t\}$ where N_c is the number of views, of a human,

2. a target human pose P_{tar}, and

3. a novel target camera view c_{tar},

the method needs to render the human in the target pose from the target camera view.

Animatable NeRF-based human rendering methods propose conditioning NeRF on an extra human pose parameter to control the rendering [29, 21, 42, 44, 30]. Below we discuss three important works in this category: Animatable NeRF [29], Neural Body [30], and Structured Local Radiance Fields for Human Avatar Modeling [44].

Animatable NeRF [29] renders a novel view of a human model in a target pose, by leveraging SMPL as human body prior and using deformations to canonical pose (which in this case is a T-pose). For a sampled query 3D point, blend weights are initialized using the SMPL model of the target pose, and a neural blend weight field (which is a learnable MLP) is used to predict blend weights offset. With the final blend weights, the query point is transformed to canonical space. The transformed co-ordinates along with the viewing direction, are fed into a NeRF module to predict the color and density. In essence, the model learns the canonical space for the human subject and predicts the color and density in canonical space, making the problem tractable. However, the approach raises a major issue; pose-dependent deformations (such as soft-tissue motion, wrinkles in clothing and shading) are not taken into account. This is because the method only learns the canonical space, and uses essentially, the same color and densities for the target pose. This can make the animation look very unrealistic.

Neural Body [30] also uses SMPL as the human body prior, but utilizes a very different approach overall. Neural Body attaches a learnable latent code to each vertex of the SMPL model for a subject. Given the target pose P_{tar}, the SMPL model is first transformed according to the pose. The 3D bounding box is then divided into small voxels with voxel size of $5mm \times 5mm \times 5mm$. The latent code of each voxel is then set as the mean of the latent codes of vertices in the voxel. A Sparse Convolutional Network [12] is then used to transform these latent codes to obtain a new latent code volume. For any sampled query point, the latent code is obtained from the latent code volume using trilinear interpolation. The interpolated latent code, along with the SMPL pose parameter 3D co-ordinates, and view direction are passed into MLPs to predict color and density. This method improves greatly upon Animatable-NeRF, as it allows for pose-dependent deformations implicitly through the interpolation of a latent code from the latent code volume, as well as through using the pose parameters as input to the final MLPs. However, Neural body is still limited in it's animation capacity due to the fact spatial changes of the latent code are still not being modelled (The 3D convolution can not model spatial changes).

Structured Local Radiance Fields for Human Avatar Modeling (SLRF) [44], to our best knowledge, is the current state-of-the-art method in this category. SLRF also uses SMPL as the human body prior, and proposes a new formulation for solving the problem. It first defines N nodes on the SMPL model using farthest point sampling. Each node also has an associated blend skinning weight vector as it has been directly sampled from the SMPL model, and therefore can be transformed to any pose given SMPL pose parameter θ. SLRF also assigns a residual translation $\Delta n_i^{(t)}$ to the node that varies according to to time. This helps model loose clothing. A conditional variational auto-encoder (cVAE) [37] attached to each of these nodes, takes in as input the timestamp and SMPL pose parameters to predict the residual translation, which is then added to the node's original co-ordinates. Given a sampled query point for which color and density are required, the local coordinate of the point with respect to each node is calculated and a tiny MLP attached to each node takes in the corresponding local coordinate to predict a feature vector. Feature vectors from all nodes are blended using to the node blend weights from the SMPL model. The aggregated feature vector is then finally passed through an MLP to predict the color and density of the sampled point. In essence, tiny MLPs are used to learn local radiance fields, while loose clothing has also been implicitly taken into account for by predicting a residual translation for the nodes. Although SLRF produces highly detailed images, one major drawback is that the performance depends on pose variance in the data due to the learned cVAE on timestamps. This also means that if animation poses and variation are very different from training poses and variations, the results would not be plausible as shown in the paper itself [44].

However, similar to original NeRF, these methods are usually subject-specific. They require training the model for each subject separately, thus need expensive retraining when new subjects need to be rendered. Unlike existing works, our animatable method is subject-agnostic. Once trained, our model can be apply to arbitrary subjects on-the-fly without extra training steps.

1.3.2 Subject-Agnostic Rendering

To our knowledge, two works have been proposed to adapt NeRF to render human images in a subject-agnostic way: Neural Human Performer [18] and GP-NeRf [6]. These methods train a single model on large scale datasets and apply it to arbitrary novel subjects without retraining. To enable the subject-agnostic property, these methods explicitly take in observation images as inputs, based on which, output images are synthesized.

The problem is formulated as follows; given:

1. a multi-view sequence of N_t timesteps of a human, $\{I_t^c | c = 1, 2, ..., N_c, t = 1, 2, ..., N_t\}$ where N_c is the number of views, of an arbitrary human, and

2. a novel target camera view c_{tar},

the goal is to synthesize $\{I_t^{ctar} | t = 1, 2, ..., N_t\}$, a novel view video from the target view. This can thus, be exploited to transform a video into a full 360 degree free-viewpoint video.

Neural Human Performer (NHP) [18] tackles the problem by using a sparse set (N_c =3 or 4) of multi-view cameras as the given views for the sequence $\{I_t^c | c = 1, 2, ..., N_c, t = 1, 2, ..., N_t\}$. Frames from the sequence are explicitly used as inputs to the pipeline so as to enable the subject-agnostic property, for the pipeline to "know" which human subject to render. The goal is to learn to use observations across input timesteps and views, to perform the novel view-synthesis. To make the problem tractable, only three timesteps $\{t - 20, t, t + 20\}$ are used as inputs to render timestep t. NHP has two main components; construction of time-augmented features $\{s^c | c = 1, 2, ..., N_c\}$, and exploiting these multi-view features to predict color and density of a query point $\mathbf{x}$. A skeletal feature bank is first calculated by projecting the SMPL model M_i at each timestep i to the image plane, and extracting pixel-aligned features $\{P_t^c | c = 1, 2, ..., N_c, t = 1, 2, ..., N_t\}$. To synthesise novel view at timestep t, timesteps $\{t - 20, t, t + 20\}$ are used as inputs to the pipeline, and a transformer module [40] is used to aggregate pixel-aligned features across input timesteps $\{P_t^c | c = 1, 2, ..., N_c, t = t - 20, t, t + 20\}$ to construct time-augmented features $\{s^c | c = 1, 2, ..., N_c\}$. Given a query point $\mathbf{x}$, pixel aligned features $\{P_{\mathbf{x}}^c | c = 1, 2, ..., N_c\}$ are sampled by projecting the point to the multi-view images at required timestep t. For the query point, features from the time-augmented features are also sampled to form the featureset $\{P'^c_{\mathbf{x}} | c = 1, 2, ..., N_c\}$. Another transformer module takes in these two feature sets ($\{P_{\mathbf{x}}^c\}$ and $\{P'^c_{\mathbf{x}}\}$), and aggregates across views to form a single feature vector per-vertex, which is then used as input to a NeRF module to predict color and density. Results produced by NHP are very smoothed out since it lacks any usage of geometry information.

Geometry-Guided Progressive NeRF (GP-NeRF) [6] is a very recent method, released during the time of writing this thesis. The code and results have not yet been made public and therefore, comparison with GP-NeRF is currently not possible. For the sake of completeness, we will discuss the formulation proposed by GP-NeRF. GP-NeRF, unlike NHP [18], only takes in one multi-view timestep t as input to the pipeline to render a novel-view of timestep t. A learnable latent code is attached to each smpl vertex v, that would help encode geometry. Each vertex from the SMPL model M_t in the pose at timestep t, is projected to each camera view and pixel-aligned features are extracted at the projected location. The pixel-aligned features $\{P_v^c | c = 1, 2, ..., N_c\}$ for a vertex v are fed into an attention module [40] as key and value with the vertex's learned latent code Q_v as the query to aggregate the features across the multiple input views. A denser geometry feature volume $\tilde{F}$ is obtained by using a Sparse Convolution Network [12] similar to Neural Body [30]. Given a query point $\mathbf{x}$, a feature vector is sampled from the feature volume $\tilde{F}(\mathbf{x})$, as well as pixel-aligned features $\{P_{\mathbf{x}}^c | c = 1, 2, ..., N_c\}$ from input multi-view images by projecting the query point to each image plane. The mean vector and variance vector of $\{P_{\mathbf{x}}^c\}$ along with $\tilde{F}(\mathbf{x})$ are passed into an MLP that predicts density at the query point. To predict

color, the pixel-aligned features $\{P^c_{\mathbf{x}}\}$ along with its mean and variance vectors are passed into an MLP that predicts the color at the query point. GP-NeRF shows improvement over NHP [18], and validates and justifies all its design choices.

As seen from the works, the typical way of leveraging input images is using *pixel-aligned features* [34], which is associating image features to an arbitrary 3D query point by projecting the query to the image plane and sampling features with bilinear interpolation. However, pixel-aligned features can only be obtained when an image of the rendered human pose is given as part of the input images. Thus, these methods cannot render novel human poses, of which an observation image is not available, making animation infeasible. By contrast, our method has specific designs to inference images of unobserved target poses from observations of limited poses, leading to an animatable human avatar.

1.4 Motivation for our work

Rendering realistic human avatars continues to be of great interest to researchers. It enables numerous downstream applications such as immersive telepresence in AR/VR, virtual try-on, game modeling, movie production, etc.

Recently, Neural Radiance Fields (NeRF) [26] has achieved impressive results on photo-realistic novel view synthesis. Efforts have been made in adapting NeRF to rendering human subjects [30, 21, 18, 29, 42, 44, 6]. However, two problems are raised by naively adapting NeRF to human subjects:

1. The model would have to be trained for one subject specifically (i.e, not **subject-agnostic**). Every novel subject would require retraining a new model, which is expensive in both time, and storage.

2. The model will not be able to support novel human poses. The model can only generate images of human poses observed in input images, we thus cannot animate the human subject with arbitrary poses (i.e, not **animatable**), as it is infeasible to include all possible poses inside inputs.

As discussed in Section 1.3, prior works have been proposed to achieve either **subject-agnostic** [18, 6] or **animatable** [21, 29, 42, 44] human avatar NeRF-based rendering.

However, no works have been proposed to address both issues and maintain the properties of animatable and subject-agnostic simultaneously, which would greatly reduce the additional steps required to render a new human body. Moreover, it is nontrivial to design such a method. Firstly, in order to make the method animatable, implicit features need be attached and driven by the body poses, which further requires an accurate shape estimation. This is easy for subject-specific methods that learn the specific body shape during training, but not for subject-agnostic methods that need to estimate body shape during inference. Moreover, in order to make the method subject-agnostic, the method should also learn to

aggregate the features from input images in a generalizable way. This is relatively easy for the task of novel view synthesis for seen poses as pixel-aligned features can be calculated, but hard for unseen poses.

In this work, we propose a novel **subject-agnostic** and **animatable** neural radiance fields (SAgA-NeRF) pipeline for human avatar from sparse-view videos by taking SMPL [22] models as priors. Given sparse-view video sequences of an arbitrary human body, our method can render the human body in novel views and arbitrary poses. In order to get an accurate body shape, we adopt the SMPL+D representation, which are optimized according to the input body masks. Moreover, in order to fuse the features from input images better, we design a novel feature fusion module. Specifically, we first fuse the multi-view features, and then attention-based fusion across timesteps according to input and target poses. Lastly, a new input frames selection method is proposed to select appropriate input frames from the video dynamically with respect to the target human pose.

The contributions are summarized as follows. At first, to the best of our knowledge, we propose the first method that is able to learn **subject-agnostic** and **animatable** neural radiance fields of human bodies from sparse view input. Moreover, we solve the challenges to fulfill this goal by adopting several novel techniques. We compare our method with subject-agnostic as well as animatable methods. The experiments show that our method performs at a similar level for the same task, as well as when both capabilities of our model together (subject-agnostic **and** animatable) are compared against the one capability of the comparison method (subject-agnostic **or** animatable). We also verify and justify our proposed components by performing ablation studies, in which our method outperforms naive baselines.

Chapter 2

Formulation of the Problem

For our formulation, we take inspiration from Neural Rendering methods discussed in Section 1.1 and NeRF-based human rendering methods discussed in Section 1.3. NeRF [26] introduces an image-based novel-view synthesis Nueral Rendering method that we adopt for our rendering, and IBRNet [41] and PIFu [34] show the efficacy of using pixel-aligned features and multi-view input images to achieve scene generalizability / subject-agnosticism. The existing subject-agnostic methods [18, 6] discussed show efficacy of using attention modules [40] to aggregate features. Animation methods discussed in Section 1.3 have less of an impact on our formulation, as their formulation are based on subject-specific learning, such as learning the canonical pose [29] or subject-specific latent code [30], or subject-specific embeddings and node offsets [44].

In this chapter, we will formulate a high-level overview of the problem, and required components, and also compare how our formulation has to be non-trivially different than the discussed subject-agnostic methods [18, 6].

Problem definition: We aim to design a method to learn subject agnostic and animatable neural radiance fields for human avatar rendering. Given:

1. a fixed set of N_t multi-view images of a human, $\{C_t^c | c = 1, 2, ..., N_c, t = 1, 2, ..., N_t\}$ where N_c is the number of views, of an arbitrary human,
2. a target human pose P_{tar}, and
3. a novel target camera view c_{tar},

the method needs to render the arbitrary human in the target pose from the target camera view, without having the need to retrain the model on the subject.

We assume that SMPL poses and body shape for all frames are available so that the SMPL model for each frame can be obtained. The parameters can be easily obtained using the OpenPose tool [4], and can be further optimized using EasyMoCap [1]. Furthermore, our method also proposes using an improved optimization, namely SMPL+D, which we discuss in detail in Chapter 3. The formulation in this chapter focuses on the rendering pipeline, after obtaining and optimizing the parameters.

We break down the formulation into three components: Inputs, Using visual cues, and radiance field decoding and rendering. We provide a high-level overview of the formulation (detail of the designs are discussed in Chapter 3: Method). We also compare our formulation with existing subject-agnostic methods: NHP [18], and GP-NeRF [6], to show that our formulation is non-trivially different

2.1 Inputs

Our formulation: Since our method is subject-agnostic, multi-view frames are required as input $\{I_t^c | c = 1, 2, ..., N_c, \quad t = 1, 2, ..., N_t'\}$ where N_c is the number of views of the input frames, and N_t' is the number of frames/timesteps being used as input. This subset is selected from the given sequence/image set $\{I\}$. This selection is for the pipeline to "know" which subject to render and to extract appropriate visual cues. Since our method is also animatable and the target human pose is not part of the input frames, the input frames need to be selected appropriately based on the target pose P_{tar}, so as to make sure that appropriate visual cues can be extracted. A selection module $\mathcal{I}$ can be design to select appropriate frames:

$$\{I_t^c\} = \mathcal{I}(P_{tar}, \{I\}) \tag{2.1}$$

Such a selection method is non-trivial to design (details in Chapter 3).

In contrast, it is very trivial for the non-animatable subject-agnostic methods to select input frames as seen from their formulations below.

NHP [18] formulation: To render a novel view of time-step i, NHP selects the set $\{i - 20, i, i + 20\}$ as inputs:

$$\{I_t^c; i\} = \{I_{i-20}^c, I_i^c, I_{i+20}^c | c = 1, 2, ..., N_c\} \tag{2.2}$$

GP-NeRF [6] formulation: To render a novel view of time-step i, GP-NeRF select just the time-step i as input:

$$\{I_t^c; i\} = \{I_i^c | c = 1, 2, ..., N_c\} \tag{2.3}$$

2.2 Using Visual Cues

Our formulation: Visual cues from the input images have to be extracted and used. The cues can be represented by pixel-aligned features $\{F_{t,v}^c\}$ for each vertex v by projecting each vertex to each input frame t and each camera-view c. These can be extracted by using a feature extractor $\mathcal{F}$ to calculate 3D-feature maps for each image, then projecting each vertex v to each image plane and sampling feature vectors from the feature maps for each of the projected locations $\{\pi_{t,v}^c\}$:

$$\{F_{t,v}^c\} = \mathcal{S}(\mathcal{F}(\{I_t^c\}), \{\pi_{t,v}^c\}) \tag{2.4}$$

where $\mathcal{S}$ represents the sampling. The visual cues are now across multiple frames and views, and hence need to be generalized. Each input frame t is a different pose P_t, and each view is from an input camera c. Therefore the generalization $\mathcal{G}$ on the features needs to be performed based on the collection of input cameras $\{c\}$, input poses $\{P_t\}$, and target pose P_{tar}, so as to construct per-vertex features $\{F_{tar,v}\}$ to infer body shape and appearance under the target pose:

$$\{F_{tar,v}\} = \mathcal{G}(\{F^c_{t,v}\}, \{c\}, \{P_t\}, P_{tar}) \tag{2.5}$$

In contrast, NHP [18] and GP-NeRF [6] do not have to deal with the extra target pose input, which simplifies their formulation. Both methods further simplify their formulation of the generalization by not using any camera view information, whereas our formulation recognizes the importance of this information to help boost the generalizability of our model to various inputs, and uses it while constructing the final per-vertex features $\{F_{tar,v}\}$ by calculating and using a vertex visibility mask (details in Chapter 3).

NHP [18] formulation: NHP similarly samples per-vertex image features $\{F^c_{t,v}\}$, but uses a different and very coarse generalization. It takes into account the query point $\mathbf{x}$ during the generalization, so as to obtain a fused vector $F_{\mathbf{x}}$ that encodes shape and appearance at the query point. The query point is used to get pixel-aligned features for the query point at time-step i (the time-step being rendered). This is possible for NHP as the target time-step is part of the input frames. The generalization therefore, only utilizes the per-vertex features, and query point $\mathbf{x}$:

$$F_{\mathbf{x}} = \mathcal{G}(\{F^c_{t,v}\}, \mathbf{x}) \tag{2.6}$$

NHP's formulation of this problem is a very simplified one, and does not utilize any geometry, camera, and human pose information.

GP-NeRF [6] formulation: GP-NeRF similarly samples per-vertex image features $\{F^c_{t,v}\}$, but uses a different generalization approach more in-line with it's goal of non-animatable novel-view synthesis. It uses a learned per-vertex v latent code $\{Q_v\}$ that encodes geometry to perform generalization, but like NHP, GP-NeRF still does not utilize camera view information. Like our formulation, GP-NeRF does not utilize the query point at this stage, but at the next stage. Their generalization therefore, only utilizes the per-vertex image features $\{F^c_{t,v}\}$ and latent code $\{Q_v\}$:

$$\{F_{tar,v}\} = \mathcal{G}(\{F^c_{t,v}\}, \{Q_v\}) \tag{2.7}$$

GP-NeRF, like NHP, does not have to deal with any pose information as it is non-animatable. It does not also utilize the camera view information and is only geometry guided through the latent codes.

2.3 Radiance Field Decoding and Rendering

This part is trivial, based on the generalization output from previous component, the generalized feature has to be decoded to color and density and rendering is done using the differentiable renderer proposed by NeRF [26].

Our formulation: Finally, given a 3D query point $\mathbf{x}$, target novel camera view c_{tar} and per-SMPL-vertex v features $\{F_{tar,v}\}$ that explain body shape and appearance under the target pose, appropriate features needs to be collected from nearby features (collection process represented by $\mathcal{A}$). An MLP (represented by $\mathcal{N}$) can be used to predict color $\mathbf{c}$ and density σ based on the sampled features.

$$(\mathbf{c}, \sigma) = \mathcal{N}(\mathcal{A}(\{F_{tar,v}\}, \mathbf{x})) \tag{2.8}$$

For each ray, the pixel color for an image from camera view c_{tar} can be rendered using the differentiable volume rendering module from NeRF once color and density for all sampled points along the ray have been predicted.

NHP [18] formulation: Since NHP has already generalized multi-view multi-timestep features using query point $\mathbf{x}$, and has obtained a single query vector, a NeRF [26] module is used to predict the color $\mathbf{c}$ and density σ:

$$(\mathbf{c}, \sigma) = \mathcal{N}(F_{\mathbf{x}}, c_{tar}) \tag{2.9}$$

GP-NeRF [6] formulation: Similar to our formulation, GP-NeRF aggregates from near the query point $\mathbf{x}$ from features obtained by the generalization. It however, splits the density prediction and appearance prediction by separate components ($\mathcal{D}$ and $\mathcal{C}$ respectively). The density prediction module utilizes the aggregated feature (using module $\mathbf{A}$), and the appearance prediction module uses only query point $\mathbf{x}$. The query point is used to extract the pixel-aligned features from the multi-view input frame at time-step i (the same time-step being rendered).

$$\sigma = \mathcal{D}(\mathcal{A}(\{F_{tar,v}\}, \mathbf{x})) \tag{2.10}$$

$$\mathbf{c} = \mathcal{C}(\mathbf{x}) \tag{2.11}$$

2.4 Summary of Our Formulation

Our formulation, as discussed has three components. We first select input frames for the pipeline from the given image set:

$$\{I_t^c\} = \mathcal{I}(P_{tar}, \{I\}). \tag{2.12}$$

We then extract per-SMPL-vertex features and generalize across input views and frames/human poses:

$$\{F^c_{t,v}\} = \mathcal{S}(\mathcal{F}(\{I^c_t\}), \{\pi^c_{t,v}\}), \tag{2.13}$$

$$\{F_{tar,v}\} = \mathcal{G}(\{F^c_{t,v}\}, \{c\}, \{P_t\}, P_{tar}). \tag{2.14}$$

We finally predict appearance and density for each given query point:

$$(\mathbf{c}, \sigma) = \mathcal{N}(\mathcal{A}(\{F_{tar,v}\}, \mathbf{x}), \mathbf{x}, c_{tar}). \tag{2.15}$$

Chapter 3

Method

As discussed, our main goal is to create the first NeRF-based method for human avatar rendering, that is subject-agnostic, as well as animatable. Given the following inputs:

1. multi-view video of an arbitrary human performance,
2. a target human pose, and
3. a target camera view,

our goal is to render, from the given camera view, the human in the target pose. As a subject-agnostic method, our pipeline would need to take in input frames $\{I_t^c | c = 1, 2, ..., N_c, \quad t = 1, 2, ..., N_t'\}$ where N_c is the number of views of the input frames, and N_t' is the number of frames/timesteps being used as input. These input frames serve as a reference, to "know" which human to render, such as in Neural Human Performer [18]. The visual cues from these input images need to be appropriately "fused", to serve as a starting point for rendering the required avatar, in the required target human pose. A tractable way of using visual cues is having features anchored to a human body prior, which is also very useful to establish correspondence between images and perform the fusion. For the prior, we use SMPL [22], which we further optimize for better geometry.

We propose a two-stage pipeline which achieves the target goal. The first stage is the **SMPL+D optimization** to optimize the SMPL geometry for a given subject (Section 3.1). The second stage is the rendering pipeline which has three main components:

1. **Posed-based input frame selection** to select appropriate input frames that would help cover the visual cues needed to render the avatar in the target pose (Section 3.2),
2. **Per-vertex feature extraction and fusion** from the input images into a single feature for each SMPL vertex, based on localized pose and angle difference between each SMPL vertex's normal and the frame's camera view direction (Section 3.3), and
3. **Decoding features and volumetric rendering**, where a feature for each query point in the radiance fields is interpolated using the nearest SMPL vertices in the

target pose, and then decoded to a color and density, followed by volumetric rendering to get the output image. (Section 3.4).

Figure 3.1 shows an overview of our rendering pipeline (Stage 2).

In this chapter, we will discuss each of the four components (SMPL+D optimization and the three components of the Rendering Pipeline) in detail. To make the pipeline easy to follow, each of the three components of the rendering pipeline will have a short summary discussing the input(s) and outputs(s), and linking to the formulation of the solution from Chapter 2.

3.1 SMPL+D Optimization

Inspired by previous Inspired by previous works [18, 30], we adopt the SMPL model [22] as a body prior to establish coarse correspondences across frames. However, the fitted SMPL shapes are usually inaccurate and not well aligned with images due to the limited model capacity. We alleviate this issue by adding per-vertex 3D offsets on the template shape (i.e, SMPL+D [2] formulation), which are optimized according to masks of frames. The implementation of our SMPL+D Optimization is inspired by EasyMoCap [1].

Parameterization: The SMPL+D model has the following parameters. For parameters shared by all frames, we have SMPL shape coefficients β, and per-vertex 3-D offset δ added on the template. For per-frame parameters, we have SMPL pose P_i, global rotation $\mathbf{R}_i \in SO(3)$, and translation $\mathbf{t}_i$ for frame i. Thus, the world coordinates of mesh vertices v and joints J can be computed as the following

$$v = \mathbf{R}_i \mathcal{V}(\beta, \delta, P_i) + \mathbf{t}_i \tag{3.1}$$

$$J = \mathbf{R}_i \mathcal{J}\left(\mathcal{V}(\beta, \delta, P_i)\right) + \mathbf{t}_i, \tag{3.2}$$

where $\mathcal{V}$ is the articulated SMPL+D model and $\mathcal{J}$ is the linear joint regressor. Given calibrated camera c, We can project the vertices and the joints of frame i to the image plane by the mapping $\Pi_c : \mathbb{R}^3 \to \mathbb{R}^2$

$$\Pi_c(x) = \mathbf{K}_c \mathbf{R}_c x + \mathbf{t}_c, \tag{3.3}$$

where $\mathbf{K}_c$ is the camera intrinsic and ($\mathbf{R}_c$, $\mathbf{t}_c$) is the camera extrinsic.

Optimization Stages and Objectives: Following EasyMoCap [1], the parameters are optimized in a multi-stage manner.

1. The SMPL shape coefficients β is determined by bone length of triangulated 3D keypoints.

2. Only the global rotations $\mathbf{R}_i \in SO(3)$ and translations $\mathbf{t}_i$ are optimized in this stage using 3D keypoint loss (L2 loss between triangulated 3D keypoints and predicted

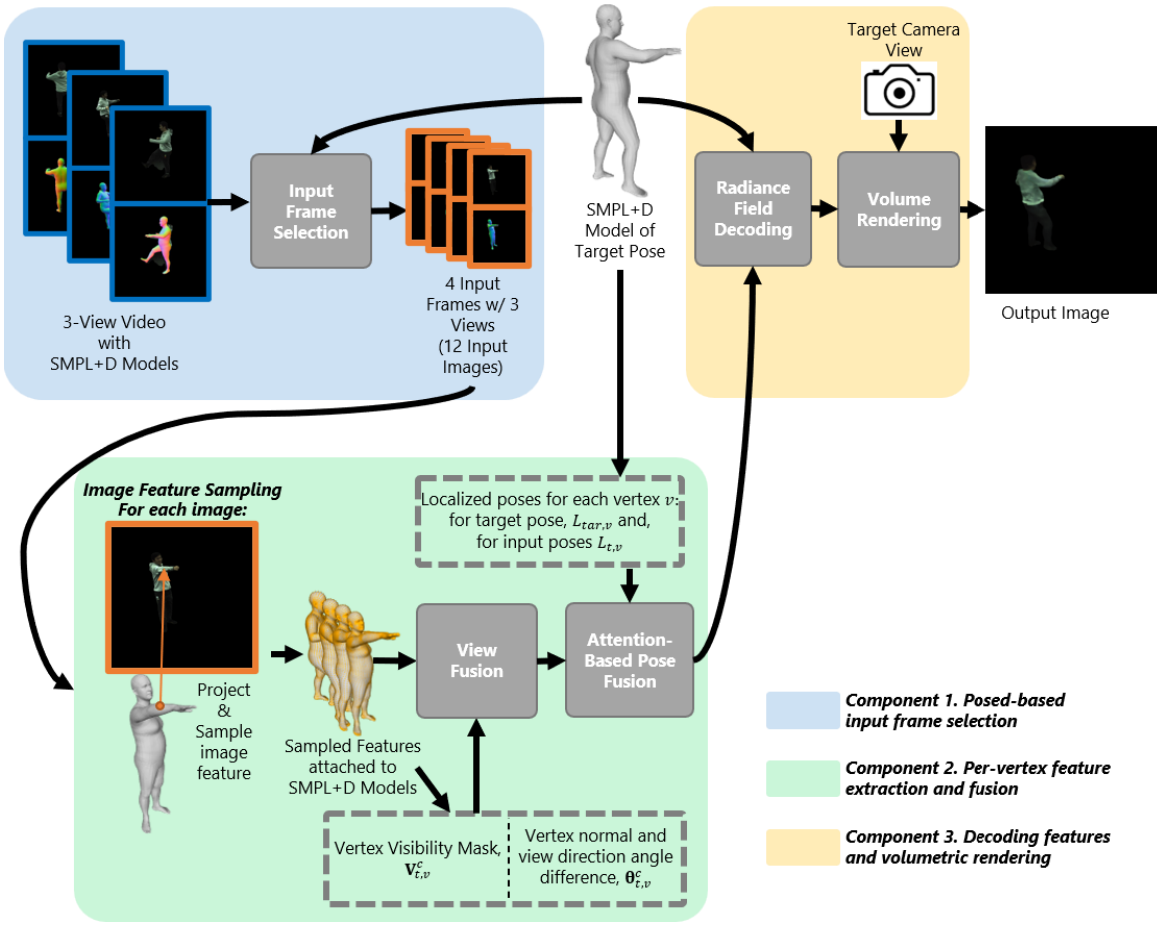

Figure 3.1: **Overview of the rendering pipeline (Stage 2).** This stage is after the SMPL+D optimization of the SMPL human shape (Section 3.1). The rendering pipeline, given inputs: (a) a sparse-view video of an arbitrary human with SMPL+D models (i.e. human poses), (b) a target human pose with SMPL+D model, and (c) a target camera view, generates an image, from the target camera view, of the arbitrary human in the target pose.

joints), temporal smoothness on joints, temporal smoothness on poses, and L2 regularization on poses.

3. The SMPL poses P_i are added to the parameters being optimized ($\mathbf{R}_i$ and $\mathbf{t}_i$), and the same losses as previous stage are used to optimize the parameters.

4. The SMPL shape coefficients β and per-vertex 3-D offset δ are added to the parameters being optimized. The parameters are optimized in this stage using 2D keypoint loss (multi-view reprojection loss between 2D keypoints and projected joints), temporal smoothness on joints, temporal smoothness on poses, L2 regularization on poses, and Mask loss (inspired by [17]).

The mask loss is the additional loss, and the per-vertex offset is the additional parameter we have added to the EasyMoCap optimization. The mask loss is inspired by Silhouette Consistency and Silhouette Coverage energy terms from [17]. The mask loss penalizes for points projected to the image plane that lie outside the foreground mask (Silhouette Consistency), and pulls points near the mask boundary towards the boundary (Silhouette Coverage). Furthermore, mask loss also includes an energy term to promote laplacian smoothing for the mesh, and also adds an As-Rigid-As Possible Regularization Loss (ARAP loss) [15].
Optimization Solver: For the optimization, similar to EasyMoCap [1], we use the LBFGS optimizer [20].

Despite the simplicity of this modification, our experiments demonstrate good improvements from it, indicating the necessity of incorporating mask-derived SMPL+D shapes for subject-agnostic methods.

3.2 Pose-based Input Frame Selection

To make our method subject-agnostic, we need to make the subject video an explicit input of our method. However, the input video usually has hundreds or thousands of frames, making it intractable to feed all frames into modern deep neural networks. To address this issue, a common practice is to select a subset of frames as network inputs to reduce computational cost (we denote this process as Input Frame Selection). Existing non-animatable method [18] uses the frame corresponding to target pose and its neighbor frames as network inputs to cover enough visual cues of target pose. Unlike NHP [18], our method also focuses on novel pose synthesis, and the frame corresponding to target pose would not exist under our problem setting. Therefore, we need to carefully select a limited number of input frames $\{I_t^c | c = 1, 2, ..., N_c, \;\; t = 1, 2, ..., N_t'\}$ while covering enough visual cues at the same time. More specifically, we select $N_c = 3$ cameras and $N_t' = 4$ frames as inputs. To solve this problem of selecting appropriate input frames, we propose our novel pose-based input frames selection, which uses the target pose P_{tar} to select the four frames from the given video.

Since the goal is to cover enough visual cues to render the human in the target pose, a tractable way to manage that is to select input frames, such that each input timestep t has at least some part of it's human pose P_t be similar to the same part of the target pose. To this end, we divide the body into four parts: left hand plus torso, right hand plus torso, left leg plus torso and right leg plus torso. For each body part p, given the SMPL pose P_t for each timestep t in the given sequence, and the target SMPL pose P_{tar}, we calculate the mean joint angle difference d_p of the set of joints in that part $J_p = \{j \forall j \in p\}$ as such:

$$d_{p,t} = \frac{1}{|J_p|} \sum_{j \in J_p} (||\vartheta(P_{t,j} \times (P_{tar,j})^{-1})||), \tag{3.4}$$

where ϑ denotes conversion of a rotation matrix to axis angle.

One timestep is selected for each body part p with minimum d_p, without replacement, and therefore, a total of four unique frames are selected, each with three uniformly-distributed views for a total of 12 input images.

Summary: This module of the rendering pipeline follows from the formulation of selecting inputs discussed in Section 2.1:

$$\{I_t^c\} = \mathcal{I}(P_{tar}, \{I\}), \tag{3.5}$$

where $\mathcal{I}$ denotes the module. The **inputs** to this module are:

1. a multi-view image/sequence set $\{I\}$ of an arbitrary human, and
2. a target human pose P_{tar}.

The **output** is the set of selected frames $\{I_t^c\}$ to be used as explicit inputs to the next module in order to provide enough visual cues to render the human in the target pose. The frames are selected by finding frames closes in pose to the target pose, by dividing the body into four parts and finding one frame closest to each body part for a total of four 3-view frames (12 images).

3.3 Feature Extraction and Fusion

Given the selected input frames $\{I_t^c\}$ and the fitted SMPL+D shapes that are aligned to input images, the next task is to collect visual cues from these inputs and infer the body shape and appearance under the target pose. To this end, we split the task into 2 sub-tasks: view fusion, and attention-based pose fusion. Figure 3.2 shows a detailed overview of the component.

In view fusion, the high-level idea is to perform a "psuedo" sparse-view 3D reconstruction on each frame by fusing image features from different camera views. For each frame t, we first extract image features with a backbone ResNet [14], then project SMPL+D vertices

onto image planes and sample pixel aligned features [18, 43, 34] with bilinear interpolation. The feature is denoted as $F^c_{t,v}$, which is anchored on SMPL+D vertex v, and sampled from camera c. Then, we develop a module to fuse the multi-view features $\{F^c_{t,v}\}^{N_c}_{c=1}$ into one feature vector $F''_{t,v}$ per-vertex. To better gather information from different views, we consider the following 2 intuitions:

View direction relative to body surface: Intuitively, if a camera is directly facing the body surface (i.e. view direction is parallel to surface normal), then its image likely contains more visual cues of the surface than another image whose camera is looking from the side (i.e. view direction is orthogonal to surface normal). To take this information into account, for each SMPL+D vertex v, we compute $\theta^c_{t,v}$ the camera view direction relative to vertex normal, for which we adopt the relative view direction formulation in IBRNet [41].

Body occlusions: The articulated structure leads to complex body occlusions. If the body surface is occluded from one camera, then we should not take information from the image of this camera. To make our model occlusion aware, we compute the visibility $\mathbf{V}^c_{t,v}$ for each vertex v under every camera view c by z-buffer testing.

Given the view direction relative to normal $\theta^c_{t,v}$, the visibility $\mathbf{V}^c_{t,v}$, and the multi-view feature $F^c_{t,v}$ of each SMPL+D vertex v, we concatenate them and use a MLP ϕ_1 followed with weighted sum to fuse across views, where the weighted sum is based on learned weights produced by a fully connected layer ϕ_2. More specifically, we have

$$F'^c_{t,v} = \phi_1\left(concat\left(F^c_{t,v}, \theta^c_{t,v}, \mathbf{V}^c_{t,v}\right)\right) \tag{3.6}$$

$$F''_{t,v} = \sum_{c=1}^{N_c}\left(F'^c_{t,v} \circ \sigma\left(\phi_2\left(F'^c_{t,v}\right)\right)\right), \tag{3.7}$$

where σ is softmax across the view dimension, and $\circ$ is element-wise multiplication.

After getting view-fused features $F''_{t,v}$, we use the attention-based pose fusion to infer the body shape and appearance under the target pose. Since our target pose is not directly corresponded to any frame t, we have to extract target pose information from view-fused features $F''_{t,v}$. Intuitively, if one frame has a more similar pose to the target one, then this frame likely contains more target pose information. Thus, we could use body poses as a good reference to determine how much contributions each frame should give to the target pose rendering. To implement this idea, we adopt multi-head attention [40] $F = \Omega\left(Q, K, V\right)$ with localized poses [35] as query $Q = \mathcal{Q}(L_{tar,v})$ and key $K = \mathcal{K}(L_{t,v})$, where $\mathcal{Q}$ and $\mathcal{K}$ are learnable embedding functions, $L_{t,v}$ is localized pose for frame t, and $L_{tar,v}$ is localized pose for target. To get the value V, we apply another learnable embedding function $\mathcal{V}$ on the view-fused features $F''_{t,v}$ as $V = \mathcal{V}(F''_{t,v})$. Finally, the attention output $F_{tar,v} = \Omega\left(Q, K, V\right)$ is a fused feature anchored on SMPL+D vertex v, which contains the body shape and appearance information of the target pose.

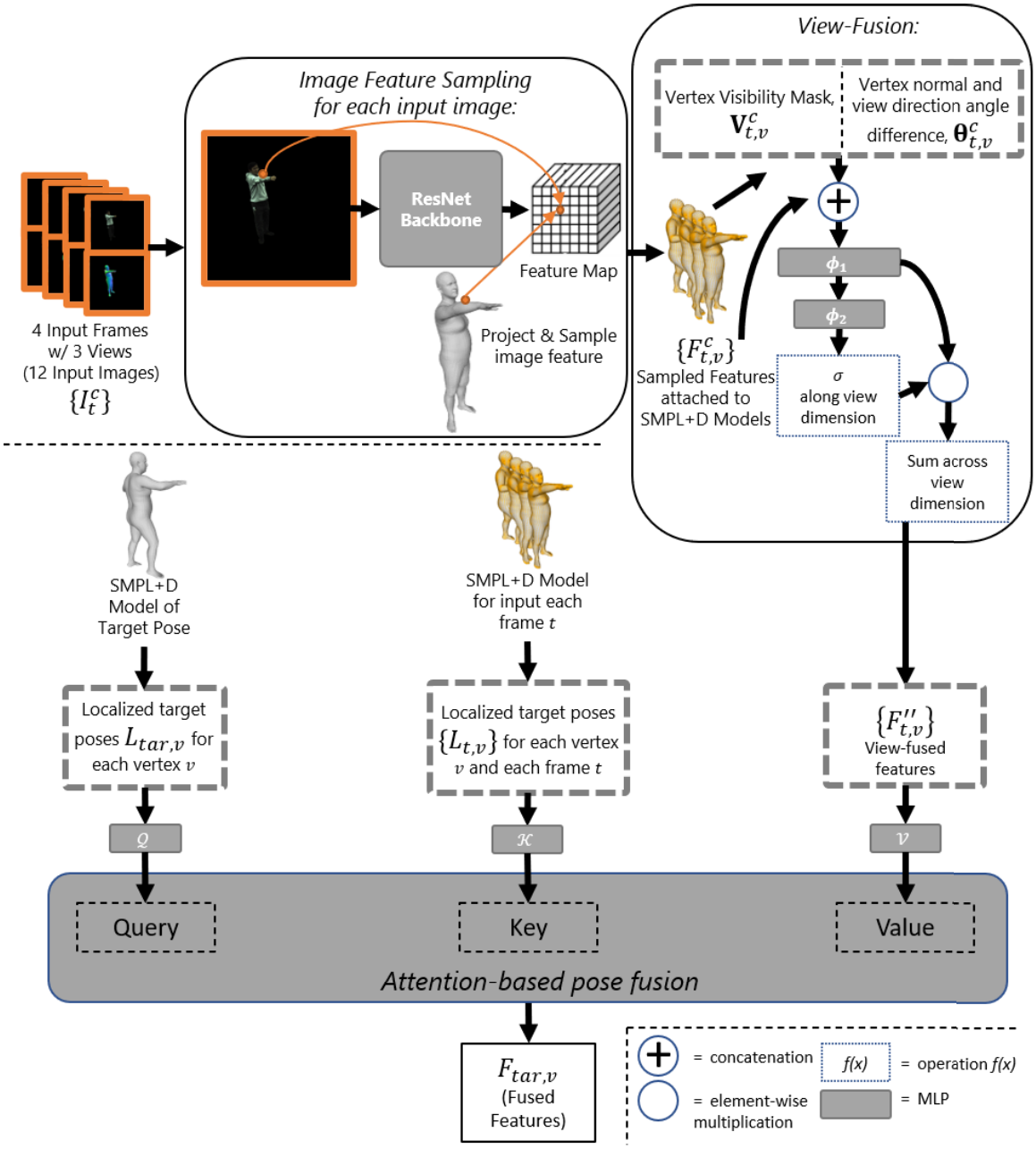

Figure 3.2: **Overview of the feature extraction and fusion.** Pixel-aligned features are extracted for each SMPL vertex for each frame. These features, along with a vertex visibility mask, and vertex-normal and view-direction angle difference, to perform a "pseudo 3D reconstruction" by fusing the features across input views. The view-fused features, along with per-vertex localized target and input poses are used to fuse the features across input frames/poses through an attention module [40]. The outputs are per-vertex features which contains the body shape and appearance information of the target pose.

Summary: This module of the rendering pipeline follows from the formulation of using visual cues discussed in Section 2.2. We first extract pixel-aligned per-vertex v features $\{F^c_{t,v}\}$ for the SMPL+D models for all input images $\{I^c_t\}$:

$$\{F^c_{t,v}\} = \mathcal{S}(\mathcal{F}(\{I^c_t\}), \{\pi^c_{t,v}\}), \tag{3.8}$$

where $\mathcal{F}$ denotes a feature map extractor (we use ResNet [14]) from images, and $\mathcal{S}$ denotes sampling of features for all projected points $\{\pi^c_{t,v}\}$ of SMPL+D vertices on to respective image planes.

The multi-view multi-pose features are then fused into single feature vectors that encode the appearance and geometry of each vertex under the target pose:

$$\{F_{tar,v}\} = \mathcal{G}(\{F^c_{t,v}\}, \{c\}, \{P_t\}, P_{tar}), \tag{3.9}$$

where $\mathcal{G}$ denotes the whole fusion process. The fusion process utilizes camera view information for the input images ($\{c\}$) to perform multi-view fusion, and uses the poses for the input frames $\{P_t\}$, and the target pose P_{tar} to perform pose fusion. The camera-view information is utilized by calculating a vertex visibility mask and view direction relative to body surface before performing the multi-view fusion. The multi-pose fusion is done using an attention-based module with target pose as query, input poses as key, and pixel-aligned features as value.

The **inputs** to this module are:

1. input images $\{I^c_t\}$ selected using pose-based input frames selection (Section 3.2),
2. camera-view information $\{c\}$) for all input images,
3. human poses $\{P_t\}$ for all input frames t, and
4. target human pose P_{tar}.

The **output** is per-vertex features $\{F_{tar,v}\}$ on the SMPL+D target model, that encode the appearance and geometry of the vertex under the target pose.

3.4 Decoding Features and Volumetric Rendering

Given features $F_{tar,v}$ on each target pose SMPL+D vertex v, we now decode it to a radiance field. Given a query point $\mathbf{x} \in \mathbb{R}^3$, we first find its 20-nearest-neighbor SMPL+D vertices $\{N\}$, gather the features $\{F_{tar,v} \forall v \in N\}$, and compute the vertex-to-query vector $\mathbf{y} = \mathbf{x} - v$. Inspired by [3], we compute the distance $d = ||\mathbf{y}||_2$ and cosine $c = cos(\mathbf{y}, \mathbf{n})$ as the rotation-invariant spatial features, where $\mathbf{n}$ is the vertex normal. These 3 per-vertex quantities ($F_{tar,v}$, d, c) are then passed through a PointNet [32] style network to predict the color and density of $\mathbf{x}$. The colors and densities of all query points are then used by volumetric rendering as

in NeRF [24] to obtain the final output image. L2-norm loss is used as supervision during training. Figure 3.3 provides a detailed overview of the component.
Summary: This module of the rendering pipeline follows from the formulation of radiance field decoding and rendering, discussed in Section 2.3:

$$(\mathbf{c}, \sigma) = \mathcal{N}(\mathcal{A}(\{F_{tar,v}\}, \mathbf{x})). \tag{3.10}$$

$\mathcal{A}$ denotes the collection of features given a 3D query point $\mathbf{x}$. We collect features by gathering the fused features from the previous module as well as some rotation invariant features for 20 vertices nearest to the query point. $\mathcal{N}$ denotes the decoding module. We use a PointNet [32] to decode the 20 unordered feature sets to a color and density for the query point.

For each sampled query point, the **inputs** to this module are:

1. the fused features from the feature extraction and fusion module (Section **??**),
2. the query point $\mathbf{x}$.

The **outputs** then, are the color $\mathbf{c} = (r, g, b)$ and density σ at that query point.

Finally, for each ray, the pixel color for an image from camera view c_{tar} can be rendered using the differentiable volume rendering module from NeRF once color and density for all sampled points along the ray have been predicted. This forms the final rendered image of the given human under target pose and target camera view.

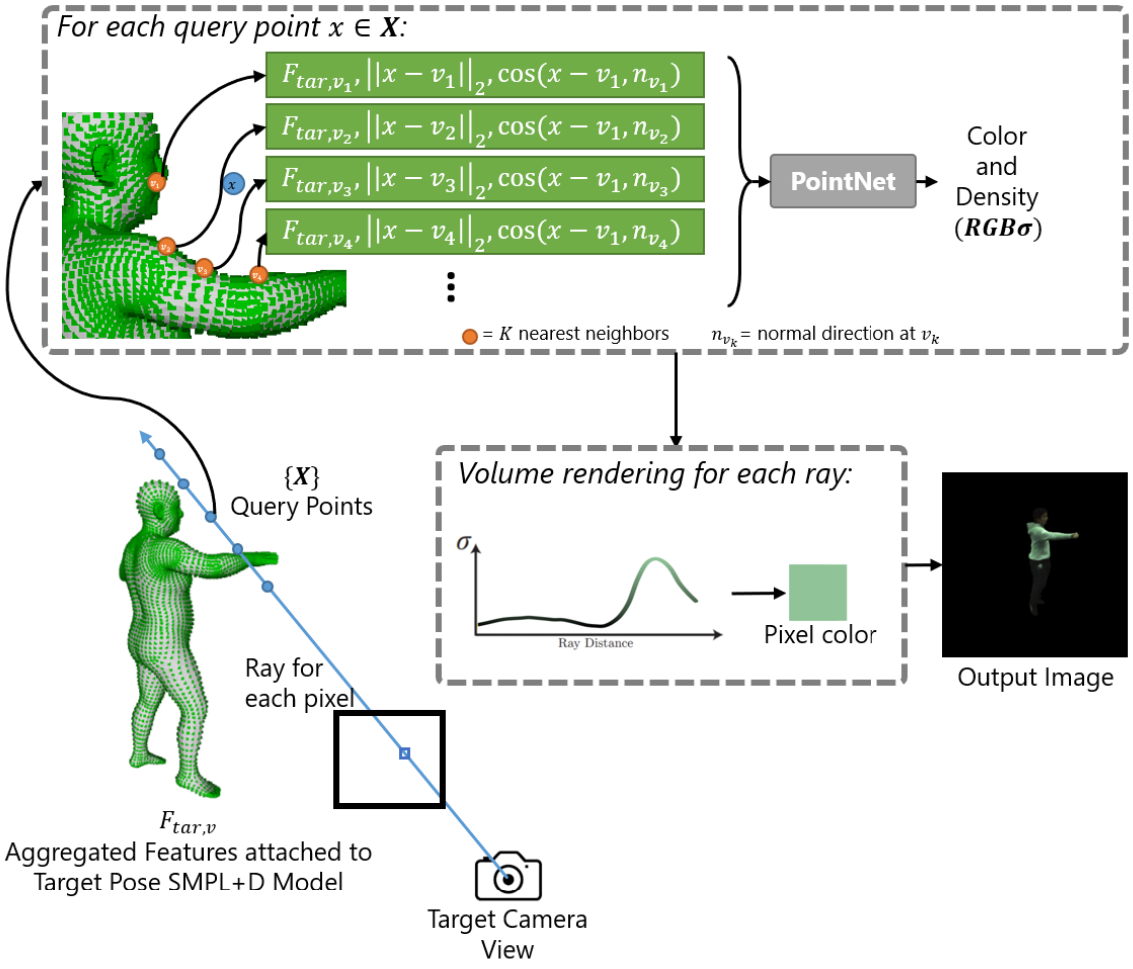

Figure 3.3: **Overview of the decoding and volumetric rendering component.** Given aggregated features from the feature fusion component attached to target pose on SMPL+D model, and the target view, we: (1) shoot a ray for each pixel in the image, then (2) sample query points along the ray, then (3) for each query point we find the nearest neighbours, gather information, and feed through PointNet [32] to get color and density, then finally (4) for each ray, integrate color and density for all query points along the ray using volumetric rendering to obtain the pixel color.

Chapter 4

Experimental Results

Our method is the first NeRF-based subject-agnostic and animatable human rendering method, and so utilizes different settings that help us show proof as concept as well as compare against existing methods that are either subject-agnostic or animatable. We perform our training and testing on the ZJUMoCap dataset [11, 30].

We first compare with a recent state-of-the-art subject-agnostic novel view synthesis method, Neural Human Performer (NHP) [18]. The comparison with NHP is the most relevant comparison that we make. For completeness, we also chose to evaluate against animatable methods, although at this stage we do still expect subject-specific models to create more detailed renderings.

Furthermore, we perform and report ablation studies to study the effect of and validate our design choices. The results show that our design choices are effective and justified.

For our quantitative analysis, we calculate and report the peak-signal-to-noise-ratio (PSNR) and structural similarity (SSIM).

4.1 Comparison Against Neural Human Performer

Neural Human Performer (NHP) [18] is a very recent method for subject-agnostic human avatar rendering. NHP is close behind [6] - the current state-of-the-art (SOTA) subject-agnostic method - in terms of performance. Comparison with [6] is not currently possible, as it was announced while this thesis was being wrapped up, and the results and code have also not yet been publicly made available.

The goal is for our pipeline to perform at a similar or higher level to NHP, while also being animatable. We use the pre-trained model provided by [18] to obtain results on their method. NHP is trained on 7 subjects, leaving out subjects **387**, **393**, and **394** for testing. For comparison, we use the same split of subjects.

We compare two settings of model against NHP, namely `ours-seen` and `ours-unseen`. `ours-seen` includes the frame with target human poses as part of the selected input frames, and therefore performs the same task as NHP, which is novel-view synthesis. `ours-unseen`

	NHP-Seen	Ours-Seen	NHP-Seen	Ours-Unseen
PSNR	25.93	**26.12**	**25.93**	25.31
SSIM	0.9041	**0.9147**	**0.9041**	0.9008

Table 4.1: Quantitative Comparison of our two settings against the one setting of NHP.

reserves the first 300 frames of each sequence as candidate input frames, while poses from the remaining frames are rendered. Comparison with `ours-unseen` provides a direct comparison of our method's goal task (subject-agnostic and animatable) with NHP's task of subject-agnostic novel-view synthesis.

Table 4.1 shows the average quantitative results obtained for NHP, `ours-seen`, and `ours-unseen` on the test subjects.

Quantitatively, our method outperforms NHP on the same task as NHP, and delivers very similar performance when the target human pose is not part of the selected input frames (animatable).

Figure 4.1 shows qualitative comparison against NHP. Our method clearly outperforms NHP on the same task as can be seen by comparing against `ours-seen`, and also shows strengths against NHP in the comparison with `ours-unseen`, as it renders better shape and details overall, and mostly better color as well.

The comparison against NHP is the most relevant comparison for our task. Our model has not only outperformed NHP on the same task but also shown that our subject-agnostic method can produce at least equally good results on unseen human poses, a task which NHP can not even handle. This shows that the boundaries of what subject-agnostic methods are capable of, can be pushed further.

4.2 Comparison with Animatable Methods

For the sake of completeness, we also perform comparison with methods that are animatable but subject-specific, i.e, they need to be trained on a per-subject basis. At this stage, we do expect subject-specific methods to create more detailed renderings simply due to their nature of being trained on one specific subject; this is especially true if our model has not seen the testing subject at all during training. We compare against three methods; Structured Local Radiance Fields for Human Avatar Modeling (SLRF) [44], Animatable Nerf (AN) [29], Neural Body (NB) [30]. The numbers for all three methods are obtained from [44]. Similar to [44], we report numbers for two subjects, subject **387** and subject **392**. The three comparison methods obtain numbers by first training the model on the first 300 frames of the video sequence of each subject and then tested by rendering the seen and unseen poses. We train our model on the remaining 8 subjects of the dataset, and test on the same two testing subjects mentioned, such that the testing subjects are completely new subject for our model during testing. For comparison against the "seen" poses setting of

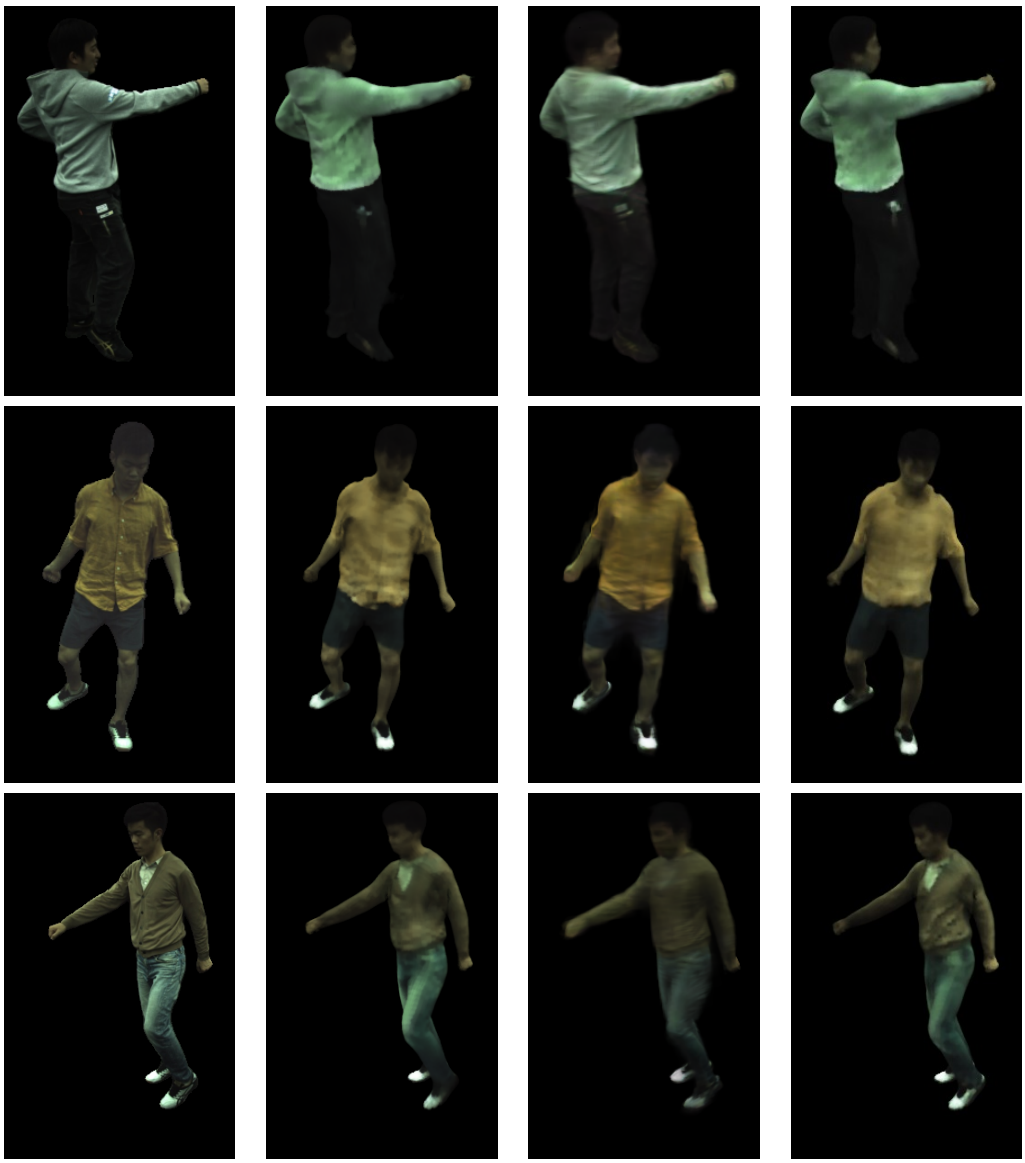

Figure 4.1: Qualitative comparison against NHP [18].

ID	Pose Type	PSNR				SSIM			
		NB	AN	SLRF	Ours	NB	AN	SLRF	Ours
387	Seen	25.79	24.38	**28.32**	26.28	0.928	0.903	**0.953**	0.925
	Unseen	21.60	21.29	23.61	**24.14**	0.870	0.860	**0.905**	0.901
392	Seen	29.44	27.43	**30.79**	29.16	0.946	0.919	**0.958**	0.940
	Unseen	25.76	24.59	26.74	**27.34**	0.909	0.889	0.927	**0.928**

Table 4.2: Quantitative comparison against animatable methods [30], [29], and [44].

animatable methods, we use the same setting as `ours-seen` (Section 4.1) so the pose is "seen". For comparison against the "unseen" poses setting of animatable methods, we use the same setting as `ours-unseen` (Section 4.1). This means that the 4 selected input frames will likely contain poses very different to the target pose. Furthermore, not training our method on the test subjects shows proof-of-concept of our animatable method working in a subject-agnostic manner.

Table 4.2 shows our quantitative results against the animatable subject-specific methods. The PSNR and SSIM measures show that quantitatively, our method performs at a similar level to these methods, while not having been trained on either of the two subjects.

Figure 4.2 shows our qualitative results against [44] for the unseen poses setting. The qualitative results shed some more light on the comparison, and as expected, show that SLRF produces more detailed images (the PSNR and SSIM are not too sensitive to smoothing in images). However, our method produces results close behind in terms of quality while not having seen the subjects at all during training. This is a strong proof-of-concept of our model, and shows potential of subject-agnostic methods being able to properly handle animation as well.

4.3 Ablation Experiments

We also investigate the effectiveness of our design choices. We claim that our main components and contributions that enable us to achieve our goals are the SMPL+D optimization, pose-based input frames selection, and pose-driven feature fusion across input frames using multi-head attention. We verify all of these design choices by training and testing variants of our model. The variants we test are:

1. our pipeline without SMPL+D optimization (**NoSD**),
2. our pipeline without pose-based input frame selection, in which we select 4 input frames (2 of them close to the frame being rendered, and the other 2 random) from the candidate range during training, and 20 random input frames from the candidate range during testing (**NoFS**),
3. our pipeline with multi-head attention replaced with mean pooling (**AvgP**), and

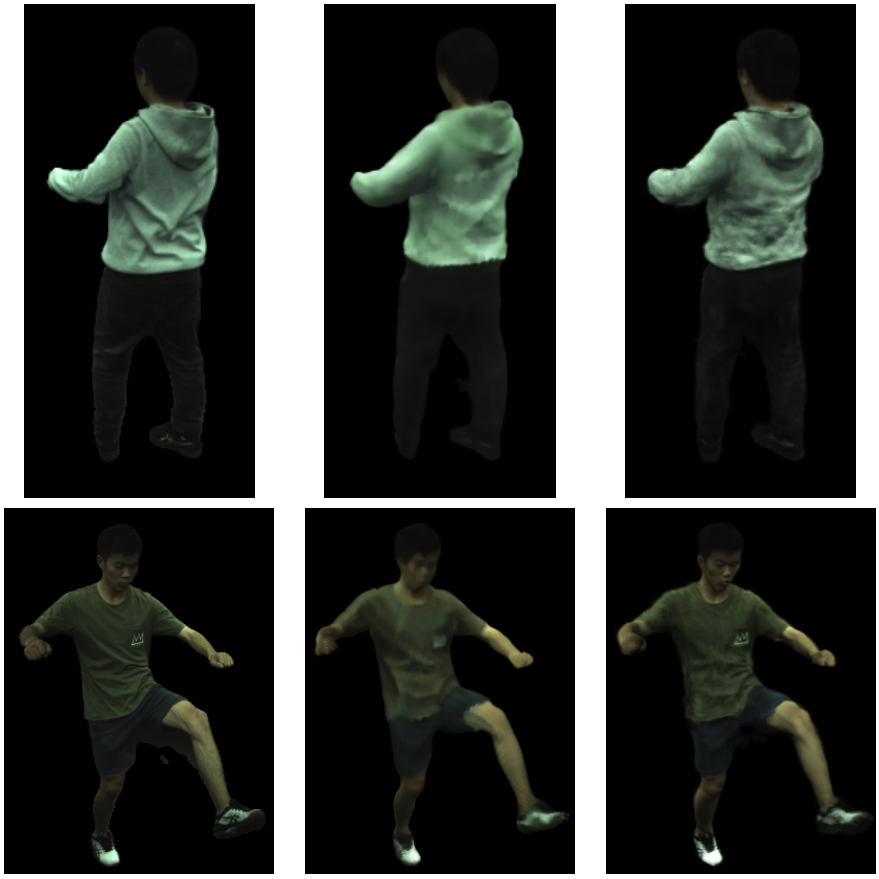

Figure 4.2: Qualitative comparison against animatable method [44].

ID	PSNR					SSIM				
	NoFS	MaxP	AvgP	NoSD	Ours	NoFS	MaxP	AvgP	NoSD	Ours
387	23.72	**24.24**	23.84	23.47	24.14	0.898	0.900	0.899	0.894	**0.901**
392	26.92	27.23	27.16	26.80	**27.34**	0.909	**0.928**	0.927	0.922	**0.928**

Table 4.3: Ablation: Quantitative Comparison

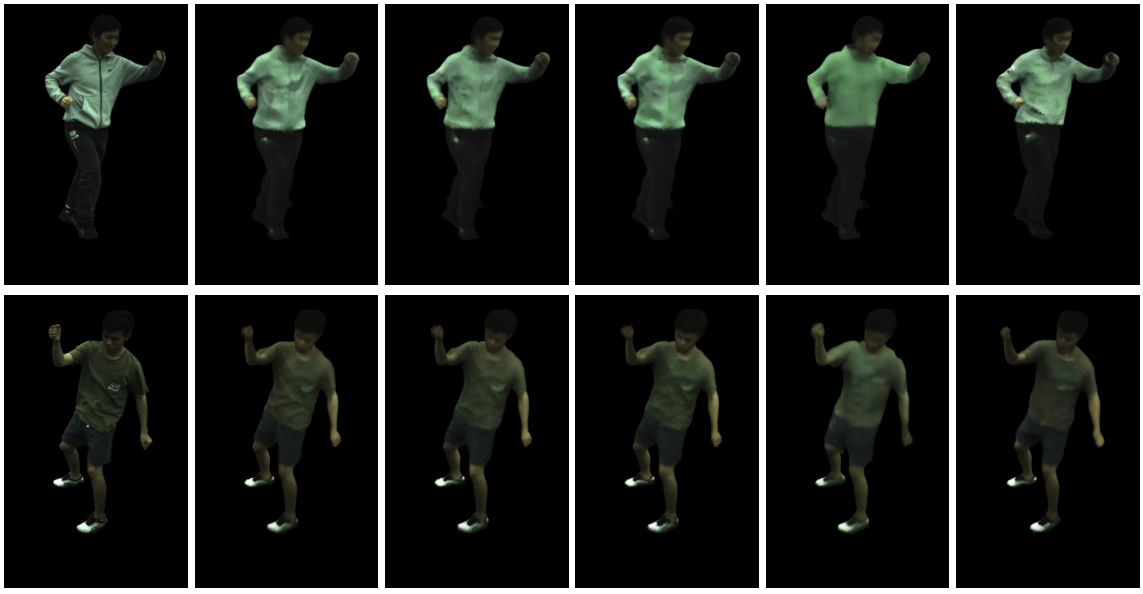

Figure 4.3: Ablation: Qualitative Comparison

4. our pipeline with multi-head attention replaced with max pooling (**MaxP**).

All tests are done on unseen subjects and unseen human poses.

Table 4.3 shows quantitative comparison of SAgA-NeRF (Ours) against the variants. SAgA-NeRF outperforms all variants but shows similar level of performance to **MaxP** and **AvgP**. Further qualitative analysis in Figure 4.3 shows that our model's fusion is justified over **MaxP** and **AvgP** as it creates more detailed wrinkles, such that the wrinkles don't appear smoothed out, and have better shading around them making them look more realistic. The qualitative comparison also clearly shows that our pose-based input frame selection and SMPL+D optimization are justified design choices.

Chapter 5

Conclusion

We present SAgA-NeRF, which is, to the best of our knowledge, the first method that is able to learn subject-agnostic and animatable neural radiance fields of human bodies from sparse view input. We propose and verify techniques to solve the challenges posed by the task of achieving our goal, with our main contributions being the pose-based input frame selection, and view and pose-based feature fusion anchored on a parametric body model. Our comparison against subject-agnostic method NHP [18] shows that SAgA-NeRF is capable of pushing the boundaries of subject-agnostic methods by not only rendering better images, but by also being pose-driven, hence increasing the range of its application. Comparison against subject-specific animatable methods also show great potential of having animation being performed on-the-fly without having to retrain the model on a new subject. In our testing on the ZJU-MoCap setting, our primary comparison against NHP shows that we achieve greatly improved performance in subject-agnostic/generalizable human avatar rendering, and provide a starting benchmark for subject-agnostic **and** animatable methods.

Bibliography

[1] Easymocap - make human motion capture easier. Github, 2021.

[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 *Proc. of Computer Vision and Pattern Recognition (CVPR)*, pages 1175–1186, 2019.

[3] Ziqian Bai, Timur Bagautdinov, Javier Romero, Michael Zollhöfer, Ping Tan, and Shunsuke Saito. Autoavatar: Autoregressive neural fields for dynamic avatar modeling. *arXiv preprint arXiv:2203.13817*, 2022.

[4] Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[5] Gaurav Chaurasia, Sylvain Duchene, Olga Sorkine-Hornung, and George Drettakis. Depth synthesis and local warps for plausible image-based navigation. In *ACM Trans. Graph.*, July 2013.

[6] Mingfei Chen, Jianfeng Zhang, Xiangyu Xu, Lijuan Liu, Yujun Cai, Jiashi Feng, and Shuicheng Yan. Geometry-guided progressive nerf for generalizable and efficient neural human rendering. In *Proc. of European Conference on Computer Vision (ECCV)*, 2022.

[7] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*.

[8] Alvaro Collet, Ming Chuang, Pat Sweeney, Don Gillett, Dennis Evseev, David Calabrese, Hugues Hoppe, Adam Kirk, and Steve Sullivan. High-quality streamable free-viewpoint video. In *ACM TOG*, 2015.

[9] Paul Debevec, Yizhou Yu, and George Borshukov. Efficient view-dependent image-based rendering with projective texture-mapping. In *EG Rendering Workshop*, June 1998.

[10] Mingsong Dou, Sameh Khamis, Yury Degtyarev, Philip Davidson, Sean Ryan Fanello, Adarsh Kowdle, Sergio Orts Escolano, Christoph Rhemann, David Kim, Jonathan Taylor, Pushmeet Kohli, Vladimir Tankovich, and Shahram Izadi. Fusion4d: Real-time performance capture of challenging scenes. In *ACM TOG*, 2016.

[11] Qi Fang, Qing Shuai, Junting Dong, Hujun Bao, and Xiaowei Zhou. Reconstructing 3d human pose by watching humans in the mirror. In *CVPR*, 2021.

[12] Benjamin Graham, Martin Engelcke, and Laurens van der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, 2018.

[13] Kaiwen Guo, Peter Lincoln, Philip Davidson, Jay Busch, Xueming Yu, Matt Whalen, Geoff Harvey, Sergio Orts-Escolano, Rohit Pandey, Jason Dourgarian, Danhang Tang, Anastasia Tkach, Adarsh Kowdle, Emily Cooper, Mingsong Dou, Sean Fanello, Graham Fyffe, Christoph Rhemann, Jonathan Taylor, Paul Debevec, and Shahram Izadi. The relightables: Volumetric performance capture of humans with realistic relighting. *ACM Transactions on Graphics*, 38(6):1–19, dec 2019.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[15] Qixing Huang, Xiangru Huang, Bo Sun, Zaiwei Zhang, and Junfeng Jiang. Category-specific object reconstruction from a single image. In *Proc. of International Conference on Computer Vision (ICCV)*, 2021.

[16] James T. Kajiya and Brian P Von Herzen. Ray tracing volume densities. In *Proc. of ACM SIGGRAPH*, 1984.

[17] Abhishek Kar, Shubham Tulsiani, João Carreira, and Jitendra Malik. Category-specific object reconstruction from a single image. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, 2015.

[18] Youngjoong Kwon, Dahun Kim, Duygu Ceylan, and Henry Fuchs. Neural human performer: Learning generalizable radiance fields for human performance rendering. *Proc. of Advances in Neural Information Processing Systems (NeurIPS)*, 34, 2021.

[19] David B. Lindell, Julien N.P. Martel, and Gordon Wetzstein. Autoint: Automatic integration for fast neural volume rendering. In *Proceedings of the conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[20] D.C. Liu and J. Nocedal. On the limited memory bfgs method for large scale optimization. In *Mathematical Programming 45*, pages 503–528. 1989.

[21] 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 Trans. on Graphics (TOG)*, 40(6):1–16, 2021.

[22] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. *ACM Trans. on Graphics (TOG)*, 34(6):1–16, 2015.

[23] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Proc. of ACM SIGGRAPH*, volume 21, pages 163–169, 1987.

[24] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, pages 7210–7219, 2021.

[25] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[26] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 405–421. Springer, 2020.

[27] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[28] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *Proc. of International Conference on Computer Vision (ICCV)*, pages 5865–5874, 2021.

[29] 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 *Proc. of International Conference on Computer Vision (ICCV)*, pages 14314–14323, 2021.

[30] 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 *Proc. of Computer Vision and Pattern Recognition (CVPR)*, pages 9054–9063, 2021.

[31] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-NeRF: Neural Radiance Fields for Dynamic Scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[32] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, pages 652–660, 2017.

[33] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *Proc. of International Conference on Computer Vision (ICCV)*, 2021.

[34] 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 *Proc. of International Conference on Computer Vision (ICCV)*, pages 2304–2314, 2019.

[35] Shunsuke Saito, Jinlong Yang, Qianli Ma, and Michael J Black. Scanimate: Weakly supervised learning of skinned clothed avatar networks. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, pages 2886–2897, 2021.

[36] Johannes L Schonberger and Jan-Michael Frahm. Structure from-motion revisited. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, 2016.

[37] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. structured output representation using deep conditional generative models. In *Proc. of Advances in Neural Information Processing Systems (NeurIPS)*, 2015.

[38] Matthew Tancik, Ben Mildenhall, Terrance Wang, Divi Schmidt, Pratul P. Srinivasan, Jonathan T. Barron, and Ren Ng. Learned initializations for optimizing coordinate-based neural representations. In *CVPR*, 2021.

[39] Ayush Tewari, Justus Thies, Ben Mildenhall, Pratul Srinivasan, Edgar Tretschk, Yifan Wang, Christoph Lassner, Vincent Sitzmann, Ricardo Martin-Brualla, Stephen Lombardi, et al. Advances in neural rendering. *arXiv preprint arXiv:2111.05849*, 2021.

[40] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Proc. of Advances in Neural Information Processing Systems (NeurIPS)*, 30, 2017.

[41] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul P Srinivasan, Howard Zhou, Jonathan T Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. Ibrnet: Learning multi-view image-based rendering. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, pages 4690–4699, 2021.

[42] Hongyi Xu, Thiemo Alldieck, and Cristian Sminchisescu. H-nerf: Neural radiance fields for rendering and temporal reconstruction of humans in motion. *Proc. of Advances in Neural Information Processing Systems (NeurIPS)*, 34, 2021.

[43] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, pages 4578–4587, 2021.

[44] Zerong Zheng, Han Huang, Tao Yu, Hongwen Zhang, Yandong Guo, and Yebin Liu. Structured local radiance fields for human avatar modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022.