AVATAR SCANNING AND PERSONALIZATION FROM CONSUMER DEVICES

DISSERTATION

ZUR ERLANGUNG DES GRADES EINES

DOKTORS DER NATURWISSENSCHAFTEN

DER TECHNISCHEN UNIVERSITÄT DORTMUND
AN DER FAKULTÄT INFORMATIK

VON

TIMO MENZEL

DORTMUND
2026

Tag der mündlichen Prüfung:	18. August 2026
Dekan:	Prof. Dr. Jens Teubner
Gutachter:	Prof. Dr. Mario Botsch
	Prof. Dr.-Ing. Justus Thies

ACKNOWLEDGMENTS

After six years, my PhD journey finally came to an end. During those years, I was lucky to have many wonderful people support me, whom I would like to thank here:

I am deeply grateful to my supervisor, Prof. Dr. Mario Botsch, for giving me the opportunity to pursue my PhD. His guidance and support always encouraged me to improve my research. His feedback and his thorough attention to detail greatly improved the quality of the avatars, making them much more convincing.

I want to thank Prof. Dr. Marc Erich Latoschik and Prof. Dr. Ulrich Schwanecke for the valuable insights provided during our meetings for our joint publications. I would also like to thank Prof. Dr. Carolin Wienrich for her involvement.

Sincere thanks go to all my co-authors for the very enjoyable and productive collaboration. Stephan Wenninger, as my predecessor in avatar research, was always there to help with difficult questions. The trip to Tsukuba stands out as a highlight of my PhD. Reinventing the avatar reconstruction with Friedemann Runte was a very educational time, leading to our exciting trip to Montreal. Together, we experienced a range of activities, from engaging presentations to exploring the city and tasting the local cuisine (like Poutine). Everything was very memorable to me. I want to thank the student research assistant Niklas Spinczyk, who was very helpful in exploring new ideas for avatar reconstruction during the Avatars for the Masses project. I appreciate the work with Erik Wolf, Peter Kullmann, and Lena Holderrieth from the University of Würzburg, who were very valuable in evaluating the personalized avatars.

Many thanks to all my colleagues in the research group, for all the insightful and interesting discussions during meetings, in the hallway, or in the Mensa. I really appreciate the support we gave each other, and I will remember our shared time of suffering while correcting endless exams. In particular, I am very grateful to Astrid Pontzen and Melanie Derksen for always being there to listen during difficult times.

This work was partially funded by the German Federal Ministry of Education and Research (BMBF) through the project *VIA-VR: Technology Platform for VR Adventures in Medical Therapy* and by the state of North Rhine-Westphalia as part of the Lamarr Institute for Machine Learning and Artificial Intelligence.

Finally, I want to thank my family and my girlfriend, Sina. Your constant support helped me navigate times of doubt and uncertainty. You encouraged me to keep pushing forward and helped me get through all the challenges along the way. In that sense, reaching this milestone was very much a team effort. I am beyond happy that we got to celebrate this milestone together. Thank you!

ABSTRACT

This thesis presents methods for creating and personalizing realistic mesh-based avatars from smartphone input data. In the first part, we present an avatar reconstruction method that enables non-expert users to generate high-quality avatars using only consumer hardware, i.e., smartphones. To the best of our knowledge, this is the first fully automated end-to-end system that integrates guided smartphone capture with automated reconstruction to generate personalized, realistic, and VR-ready full-body avatars without requiring additional user input or preprocessing. Our reconstruction system consists of an iOS application that visually guides users through the scanning process and a server that automatically computes realistic avatars using photogrammetry, landmark detection, and template fitting. While this setup allows accessible avatar creation, the system is prone to motion-induced errors during scanning. We address this by reducing the effects of micro-motion at the photogrammetry stage and adapting the reconstruction method accordingly. Specifically, we compute partial point clouds from sub-scans using photogrammetry, incorporate silhouette constraints to improve fitting robustness, and apply inverse rendering to refine the texture on a per-image basis.

The resulting avatars accurately match the scanned person's shape. However, animating the person's face is only possible with generic blendshapes. While this is sufficient to create new facial expressions, the resulting facial expression may lead to incongruent social cues. To address this, the second part of this thesis focuses on the personalization and animation of facial expressions. Building on the example-based facial rigging (EBFR) approach by Li et al. [LWP10], we introduce a method for personalizing blendshapes of realistic 3D avatars using smartphone input data. We developed an application to automatically capture example facial expressions. From these, we compute 52 ARKit-compatible blendshapes using a position-based EBFR formulation and seamless partial deformation transfer. The resulting personalized blendshapes enable users to create faithful face animations. Finally, we evaluate the plausibility and naturalness of these facial animations by comparing different blendshape animation methods, including synthesized, tracked, and static (non-animated) facial expressions.

CONTENTS

1 Introduction **1**

I Animatable Avatar Reconstruction from Smartphone Input **7**

2 Motivation **9**

3 Fundamentals **11**
3.1 Statistical Human Templates . . . 11
3.1.1 Pose Adjustments . . . 11
3.1.2 Shape Adjustments . . . 14
3.2 Template Fitting . . . 15
3.3 Multi-Camera Rig-Based Pipeline . . . 21
3.4 Smartphone-Based Pipeline . . . 23

4 Related Work **27**
4.1 Avatars for Self-Representation in Virtual Reality . . . 27
4.2 Realistic Avatar Reconstruction . . . 28
4.2.1 Representation of Avatars . . . 29
4.2.2 Input Devices . . . 30

5 Avatars for the Masses: Smartphone-Based Reconstruction of Humans for Virtual Reality **33**
5.1 Avatar Reconstruction . . . 34
5.1.1 Smartphone-Based Data Acquisition . . . 34
5.1.2 Server-Based Reconstruction Pipeline . . . 37
5.2 Evaluation . . . 42
5.2.1 Quantitative Comparisons . . . 43
5.2.2 Qualitative Comparisons . . . 44
5.3 User Study . . . 47
5.3.1 Apparatus . . . 49
5.3.2 Measures . . . 51
5.3.3 Procedures . . . 52
5.3.4 Participants . . . 53
5.3.5 Data Analysis . . . 55
5.3.6 Results . . . 55
5.4 Discussion . . . 57
5.4.1 Avatar Evaluation . . . 58
5.4.2 Smartphone App Evaluation . . . 59
5.4.3 Enhancements to the Avatar System . . . 59

5.4.4 Limitations . . . 60
5.5 Conclusion . . . 61

6 Compensating Motion-Induced Errors in Smartphone-Based VR Avatar Reconstruction 63
6.1 Method . . . 65
6.1.1 Sub-Scans . . . 65
6.1.2 Photogrammetry Selection . . . 66
6.1.3 Loss Functions . . . 68
6.1.4 Coarse-to-Fine Fitting . . . 70
6.1.5 Texture Generation . . . 72
6.2 Evaluation . . . 73
6.2.1 Qualitative Evaluation . . . 75
6.2.2 Quantitative Evaluation . . . 76
6.2.3 Ablation Studies . . . 77
6.2.4 Limitations . . . 80
6.3 Conclusion . . . 80

II Personalized Facial Animation for Avatars 81

7 Motivation 83

8 Fundamentals 85
8.1 Blendshapes . . . 85
8.2 Deformation Gradients . . . 86
8.3 Example-Based Facial Rigging . . . 87

9 Related Work 93
9.1 Blendshape Personalization . . . 93
9.2 Blendshape Weight Generation for Facial Animation . . . 96

10 Automated Blendshape Personalization for Faithful Face Animations Using Commodity Smartphones 99
10.1 Method . . . 100
10.1.1 Input Data . . . 100
10.1.2 Blendshape Personalization . . . 101
10.1.3 Blendshape Transfer . . . 103
10.2 Results . . . 105
10.2.1 Face Mask Personalization . . . 105
10.2.2 Avatar Blendshape Personalization . . . 107
10.2.3 Computation Time . . . 109
10.3 Discussion . . . 109
10.4 Conclusion . . . 111

11 An Evaluation of Other-Avatar Facial Animation Methods for Social VR 113

11.1 Method . . . 114
11.1.1 Character Creation . . . 114
11.1.2 Performance Capture . . . 114
11.1.3 Animation . . . 116
11.1.4 Immersive Observation . . . 119
11.1.5 Procedure . . . 119
11.1.6 Participants . . . 121
11.2 Results . . . 121
11.2.1 Behavior Naturalness . . . 121
11.2.2 Appearance and Behavior Plausibility . . . 122
11.2.3 Qualitative Feedback . . . 122
11.3 Discussion . . . 123
11.4 Limitations and Future Work . . . 124
11.5 Conclusion . . . 124

12 Summary **125**

Bibliography **129**

A Appendix **151**
A.1 GART Configuration . . . 151
A.2 InstantAvatar Configuration . . . 152
A.3 Body Movement Tasks . . . 153
A.4 Preference Questions . . . 154
A.5 Interview Protocols . . . 154

B Appendix **157**

C Appendix **171**
C.1 Subject 2 . . . 171
C.2 Subject 3 . . . 173
C.3 Subject 4 . . . 174

Tools Used **177**

1 INTRODUCTION

User representation has evolved noticeably over the past decades. Early on, user names allowed people to participate in online communities and forums. They enabled users to distinguish one another but offered only limited possibilities to express personality. With the introduction of social media in the 2000s, profiles began to include more personal information (e.g., images) [Nag17], enabling richer forms of self-representation. As profile similarity is a key factor for trust-based user interaction [Gol09], this development highlights the importance of expressive user representation.

The introduction of virtual reality (VR) and the increasing availability of head-mounted displays have further evolved user representation, moving it from static profiles to embodied forms. Thus, researchers have begun to examine the importance of realistic and personalized avatars for faithful self-representation. This research has shown that realistic, personalized full-body avatars increase the user's sense of presence and embodiment or self-identification with the avatar [Fie+24; Fie+26; KLC23; Sal+23; Wal+18], and can increase emotional response [Gal+21; Wal+18]. They are therefore well-suited for metaverse-like social VR applications [AI21; Lat+19; Mys22; Yoo+19] and therapy scenarios [Döl+22; Sam+21], where faithful user representation is crucial.

Consequently, extensive research has been conducted in avatar reconstruction. Many methods rely on large multi-camera rigs [Ach+17; Ma+21; She+24] to produce photorealistic avatars. However, these systems are expensive, require specialized personnel, and are restricted to a fixed location. As VR devices have become more affordable [RMN21], high costs for professional scanners remain a key limitation [Bar+21]. Therefore, other approaches try to lower the hardware barrier by using monocular cameras or smartphones instead [All+18a; All+18b; Wen+20]. Smartphone-based methods have the potential to increase access to avatars, as high-quality cameras are readily available on smartphones today. Instead of a simultaneous capture process with multiple cameras, these approaches rely on videos or sequentially captured images. However, this sequential scanning process leads to other problems. Often, the resulting avatars are of lower geometric quality [All+18a; All+18b] or scanning procedures are even more difficult [Wen+20]. More recent methods [Jia+23; Mor+24a] based on Neural Radiance Fields [Mil+21] or 3D Gaussian Splatting [Ker+23] demonstrate impressive results but remain limited due to computational demands.

Part of the work presented in this thesis was conducted within the context of the project *VIA-VR: Technology Platform for VR Adventures in Medical Therapy*[1]. The aim of VIA-VR was to build a software framework for creating VR applications, specifically targeting medical therapy scenarios. Through

1 https://hci.uni-wuerzburg.de/projects/via-vr

templating and modeling abstractions, the VIA-VR editor enables non-expert users to create VR scenarios without programming knowledge.

This application context motivated the use case of affordable, easy-to-use, and location-independent avatar reconstruction methods for VR. Despite recent progress in smartphone-based avatar reconstruction, current approaches remain limited for this scenario. To address these limitations, this thesis presents avatar reconstruction methods that fill this gap. The methods we present use smartphone-captured input data. This reduces costs while simultaneously increasing availability due to the ubiquity of smartphones. To ensure high-quality results and ease of use, we introduce measures to reduce the impact of motion during scanning and a scanning app that enables even inexperienced users to capture data of sufficient quality.

For interactive applications, it is insufficient to only depict a static version of the user. Therefore, the avatar is usually equipped with an animation skeleton rig and skinning weights, allowing the user to control its pose by adjusting the animation skeleton's joint angles. In avatar reconstruction, this is commonly done by fitting a parametric template model to the captured data, which provides the animation skeleton and skinning weights that are reused or adapted to drive the resulting avatar [Ach+17; All+18b; Wen+20]. While skeleton rigs can also be used for facial animation, a more prominent approach is to use blendshapes – a set of predefined base expressions that can be combined to form new expressions [Lew+14].

Similar to animation skeletons, it is possible to transfer generic template blendshapes to the resulting avatar. While generic skeletons work well for body pose animation [Lop+15; OBB20], transferring generic blendshapes works only to a certain extent. Facial expressions are a central channel of non-verbal behavior and are therefore especially important for social interaction, as non-verbal behavior conveys a substantial portion of the information communicated [EF69; MFH12; RE20; Rot+16]. If a set of generic template blendshapes is transferred to the user's avatar, new facial expressions can be created, but it is generally not possible to accurately represent the user's unique facial expressions. The resulting subtle differences can lead to incongruent social cues, because humans can notice even small changes in facial expressions [LW22; Rot+16]. The solution is to create a set of personalized facial expressions for the avatar [Cas+15; Gar+16; IBP15; LWP10].

In this thesis, we present an automated and example-driven blendshape personalization method that captures the unique features of the user's facial expressions and transfers them to the resulting avatar. Reconstructed facial expressions created with these personalized blendshapes are more accurate than their generic counterparts in terms of geometric error, enabling more faithful facial animation. Our personalization method neatly fits into our targeted use case of ease of use and affordability, as we again require only smartphone-captured input data and present a fully automated algorithm. In addition, this thesis evaluates the influence of facial tracking and audio-driven blendshape weight generation methods, as personalized blendshapes alone do not guarantee plausible facial animation. Since facial tracking may be

infeasible in VR use cases, we specifically analyze the effectiveness of audio-driven blendshape weight generation for plausible, natural facial animation.

Contribution

The main contributions of this thesis are:

- A system to automatically create animatable VR-ready full-body avatars for self-representation using smartphone-only input targeting non-expert users.
- A method to reduce the influence of micro-motions on the reconstructions for photogrammetry-based template fitting.
- An improved example-based facial rigging algorithm to generate personalized blendshapes for full-body avatars.
- A user study evaluating blendshape weight generation methods, analyzing their impact on naturalism and plausibility for personalized facial animation.

More detailed lists of contributions can be found in each chapter.

Publications and Copyright

This thesis is based on the following publications:

1. T. Menzel, M. Botsch, and M. E. Latoschik. "Automated Blendshape Personalization for Faithful Face Animations Using Commodity Smartphones". In: *Proceedings of the 28th ACM Symposium on Virtual Reality Software and Technology*. 2022. DOI: 10.1145/3562939.3565622 – (CC BY 4.0)
2. P. Kullmann, T. Menzel, M. Botsch, and M. E. Latoschik. "An Evaluation of Other-Avatar Facial Animation Methods for Social VR". In: *Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems*. 2023. DOI: 10.1145/3544549.3585617
3. T. Menzel, E. Wolf, S. Wenninger, N. Spinczyk, L. Holderrieth, C. Wienrich, U. Schwanecke, M. E. Latoschik, and M. Botsch. "Avatars for the Masses: Smartphone-Based Reconstruction of Humans for Virtual Reality". In: *Frontiers in Virtual Reality* 6 (2025). DOI: 10.3389/frvir.2025.1583474 – (CC BY 4.0)
4. F. Runte, T. Menzel, U. Schwanecke, and M. Botsch. "Compensating Motion-Induced Errors in Smartphone-Based VR Avatar Reconstruction". In: *Proceedings of the 2025 31st ACM Symposium on Virtual Reality Software and Technology*. 2025. DOI: 10.1145/3756884.3765995 – (CC BY 4.0)

The publications marked with (CC BY 4.0) were originally published under the Creative Commons Attribution 4.0 International License and are reused or adapted in accordance with the terms of that license.

Other publications that I contributed to that are not directly relevant to this thesis are:

- P. Kullmann, T. Schell, T. Menzel, M. Botsch, and M. E. Latoschik. "Coverage of Facial Expressions and Its Effects on Avatar Embodiment, Self-Identification, and Uncanniness". In: *IEEE Transactions on Visualization and Computer Graphics* 31.5 (2025). DOI: 10.1109/TVCG.2025.3549887

The accompanying videos for the papers listed, which are referenced throughout this thesis, are available through the DOI links to the official publications:

1. Associated with Menzel et al. (2022)
   DOI: https://doi.org/10.1145/3562939.3565622
2. Associated with Kullmann et al. (2023)
   DOI: https://doi.org/10.1145/3544549.3585617
3. Associated with Menzel et al. (2025)
   DOI: https://doi.org/10.3389/frvir.2025.1583474
4. Associated with Runte et al. (2025)
   DOI: https://doi.org/10.1145/3756884.3765995

The website for the avatar reconstruction system is available through this link: https://avatars.cs.tu-dortmund.de

Thesis Outline

The remainder of this thesis is structured as follows:

Part I deals with our work on the reconstruction of full-body avatars from smartphone input. It introduces fundamental concepts, reviews prior work, and presents novel reconstruction methods developed in this thesis.

Chapter 2 motivates the need for accessible and affordable smartphone-based avatar reconstruction and introduces our approach for generating high-quality, VR-ready avatars for non-expert users.

Chapter 3 introduces the concepts of animatable statistical template models, template fitting, and the existing avatar reconstruction pipelines prior to our research, which serve as the foundation of this work.

Chapter 4 reviews the current state of research with respect to reconstruction techniques and avatars for self-representation.

Chapter 5 then presents our fully automatic avatar reconstruction system, *Avatars for the Masses*, which enables non-experts to easily create VR-ready avatars from smartphone input.

Chapter 6 addresses micro-motions during the scan procedure and proposes an effective method to adapt the initial reconstruction pipeline, thereby reducing artifacts in the resulting avatars.

Part II deals with our work on facial expressions. It introduces fundamental concepts in facial animation and expression transfer, reviews prior work, and presents our approach to blendshape personalization and the evaluation of blendshape animation techniques.

Chapter 7 motivates the necessity of realistic, faithful facial animation and introduces our smartphone-based blendshape personalization approach. It also motivates the need for a user study evaluating facial tracking and facial expression synthesis techniques.

Chapter 8 describes linear blendshapes for facial animation, deformation gradients, and the approach *example-based facial rigging* by Li et al. [LWP10], which form the base concepts used in this part.

Chapter 9 reviews recent approaches to personalizing avatars' facial expressions and generating blendshape weights for faithful facial expressions.

Chapter 10 describes our work on example-based blendshape personalization using only smartphone input, which perfectly complements the previously introduced avatar reconstruction methods.

Chapter 11 then investigates the effects of different techniques for generating blendshape weights for facial animation on the naturalness and plausibility of realistic avatars.

Finally, Chapter 12 summarizes the main contributions and limitations of this thesis and gives an outlook on future research possibilities.

Part I

ANIMATABLE AVATAR RECONSTRUCTION FROM SMARTPHONE INPUT

2 MOTIVATION

Building on the motivations discussed in the main introduction of the thesis, this part focuses on the reconstruction of realistic, full-body avatars. As introduced earlier, VR has become increasingly popular and enables faithful self-representation for a variety of applications (e.g., social VR and therapy). However, although modern head-mounted displays (HMDs) have become more affordable, high-quality avatar reconstruction remains less accessible.

High-quality avatars can be generated with professional multi-camera rigs [Ach+17; Ma+21; She+24], but these systems are expensive and location-bound, making them inaccessible for most users. To reduce hardware requirements, many methods use monocular videos or images captured with a smartphone or a single camera [All+18a; All+18b; Wen+20] instead. While this increases affordability, it compromises reconstruction quality or increases usage complexity. More recent approaches employ Neural Radiance Fields (NeRFs, [Mil+21]) or 3D Gaussian Splatting (3DGS, [Ker+23]) to represent avatars. However, these methods remain suboptimal for VR applications due to their high computational cost and limited ability to achieve real-time performance. Thus, there remains a gap for fast, affordable, and easy-to-use avatar reconstruction methods, preventing the use of realistically personalized avatars for many scenarios.

In this part of the thesis, we address this gap by presenting an avatar reconstruction system that uses smartphone-captured input to automatically generate high-quality, VR-ready avatars, making this technology accessible to non-expert users. Furthermore, we propose a reconstruction approach to account for micro-motions during scanning, reducing motion-induced artifacts in the reconstructed avatars.

The remainder of this part is structured as follows:

Chapter 3 introduces the creation of animatable statistical template models and describes the process of template fitting to generate an animatable 3D human avatar from point cloud input. Furthermore, it presents the multi-camera rig and smartphone-based avatar reconstruction methods, which serve as the baselines for this work.

Chapter 4 then reviews and classifies related work in the field of avatar reconstruction and avatars for self-representation in VR.

Chapter 5 presents our end-to-end avatar reconstruction system. It comprises an easy-to-use smartphone application that guides users through the scanning process, and a server-based pipeline that fully automatically reconstructs realistic, personalized, and VR-ready avatars in about 20 minutes. The system enables non-expert users to create high-quality avatars even in uncontrolled outdoor environments. Our user-centric evaluation confirms the usability of our smartphone app and the quality of the resulting avatars, indicating their potential for the widespread adoption of avatars.

Chapter 6 then describes improvements to the initial avatar reconstruction system. Despite generating high-quality avatars, the system is prone to motion-induced errors in the scan results, resulting from micro-motions during the scanning process. We present steps to reduce motion influence on the photogrammetry basis and revise the initial approach to achieve noticeably better results while keeping the same scanning routine.

3 FUNDAMENTALS

In this chapter, we introduce the theoretical foundations underlying our avatar reconstruction approaches. We begin by describing the construction of an animatable statistical human template model and a process for fitting it to 3D input data, e.g., a point cloud. Subsequently, we describe two existing avatar reconstruction approaches [Ach+17; Wen+20] that serve as the technical foundation for the research presented in this thesis. Both approaches reconstruct a textured human mesh from a set of input images using a parametric human body model. The main differences between the two approaches lie in the input data used, the corresponding preprocessing steps, and the employed texture generation methods.

3.1 Statistical Human Templates

The general idea behind template fitting is to match the shape of a given 3D template model to the input data (e.g., a point cloud). The template model is defined by a surface mesh describing the shape of the body. It consists of a set of vertices $\mathcal{V}$ represented by the vector of stacked vertex positions $\boldsymbol{V} \in \mathbb{R}^{3N_V}$, a skeleton rig consisting of a set of joints $\mathcal{J}$, which are controlled by joint angles $\boldsymbol{\theta} \in \mathbb{R}^{3|\mathcal{J}|}$, along with corresponding skinning weights, a set of facial blendshapes (see Section 8.1), and auxiliary meshes to model internal details like eyes and teeth.

3.1.1 Pose Adjustments

The skeleton rig and skinning weights are necessary to adjust the pose of the model. Every joint defines its own coordinate system, specifying its rotational and translational offset from its parent's coordinate system:

$$\begin{bmatrix} \boldsymbol{R}(\boldsymbol{\theta}) & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{bmatrix} \in \mathbb{R}^{4\times4},$$

where $\boldsymbol{R}(\boldsymbol{\theta}) \in SO(3)$ is a rotation matrix obtained from joint angles $\boldsymbol{\theta}$ and $\boldsymbol{t} \in \mathbb{R}^3$ is the translational offset vector. The joints in the skeleton rig are arranged hierarchically, with each joint having a single parent and possibly multiple child joints. Each joint's *local* matrix $\boldsymbol{L}_j$ specifies the joint's transform with respect to its parent (see Figure 3.1, top). Thus, if we adjust the local matrix of a joint (e.g., to lift the arm), this also moves all descendant joints. We denote transformation matrices in the initial pose (also called bind pose) with an overbar, e.g., $\overline{\boldsymbol{L}_j}$ (Figure 3.1, top left), and matrices in the animated pose are denoted without an overbar, e.g., $\boldsymbol{L}_j$ (Figure 3.1, top right). The first joint in the hierarchy (see Figure 3.1, red sphere) is the root joint. Because it has no

parent joint, its local transform $\boldsymbol{L}_1$ is the same as the global transform $\boldsymbol{G}_1$ and specifies the global rotation and translation of the model. By concatenating the local transformation matrices along the unique path from root to the j-th joint, we can compute *global* transformation matrices $\boldsymbol{G}_j$ that map points from the coordinate system of the j-th joint into the world coordinate system (see Figure 3.1, top):

$$\boldsymbol{G}_j = \begin{cases} \boldsymbol{L}_1, & \text{if it is the root joint,} \\ \boldsymbol{G}_{\text{parent}(j)}\, \boldsymbol{L}_j, & \text{otherwise.} \end{cases}$$

This recursive computation of local and global joint transformation matrices from given joint angles is called *forward kinematics*.

We can adjust the pose of the skeleton rig (see Figure 3.1, from top left to top right) by rotating the local transformation matrices and recomputing the global matrices. After computing new transformation matrices for the joints, vertex positions are recomputed in a process called *skinning*. A simple yet effective and widely used method for skinning meshes is *linear blend skinning* (LBS, [MLT89]). In LBS, we assign every vertex a non-negative weight for each joint $w_{i,j}$ describing the influence of each joint's pose on the vertex (see Figure 3.1, bottom left). For every vertex $\boldsymbol{v}_i \in \mathbb{R}^3$, the skinning weights satisfy $\sum_{j=1}^{|\mathcal{J}|} w_{i,j} = 1$. The resulting vertex position can then be computed using:

$$\begin{pmatrix} \boldsymbol{v}_i' \\ 1 \end{pmatrix} = \sum_{j=1}^{|\mathcal{J}|} w_{i,j} \boldsymbol{G}_j(\boldsymbol{\theta})\, \overline{\boldsymbol{G}_j}^{-1} \begin{pmatrix} \boldsymbol{v}_i \\ 1 \end{pmatrix},$$

where $\begin{pmatrix} \boldsymbol{v}_i \\ 1 \end{pmatrix}$ is the i-th vertex position in the initial pose in homogeneous coordinates, $w_{i,j}$ is the skinning weight for the i-th vertex and the j-th joint, $\boldsymbol{G}_j(\boldsymbol{\theta})$ is the current global transformation matrix of the j-th joint obtained from joint angles $\boldsymbol{\theta}$, and $\overline{\boldsymbol{G}_j}^{-1}$ is the inverse of the global transformation matrix of the joint in the bind pose, which transforms the vertices from bind pose into the local coordinate system of the j-th joint.

We define three variants of the LBS operation, depending on the type of input:

- $\text{skin}(\boldsymbol{V}, \boldsymbol{\theta})$: Applies LBS to the full list of vertices $\boldsymbol{V}$, returning the skinned vertex positions $\boldsymbol{V}'$.
- $\text{skin}_v(\boldsymbol{v}_i, \boldsymbol{\theta})$: Applies LBS to a single vertex $\boldsymbol{v}_i$, returning its skinned position $\boldsymbol{v}_i'$.
- $\text{skin}_C(\boldsymbol{v}_c, \boldsymbol{\theta})$: Applies LBS to a point in a triangle $\boldsymbol{v}_c$ defined by barycentric coordinates. The function computes skinned positions of the triangle's vertices using skin_v and then returns the interpolated skinned position $\boldsymbol{v}_c'$.

This allows our template to represent a wide variety of human poses by adjusting its skeletal configuration (see Figure 3.1, bottom right).

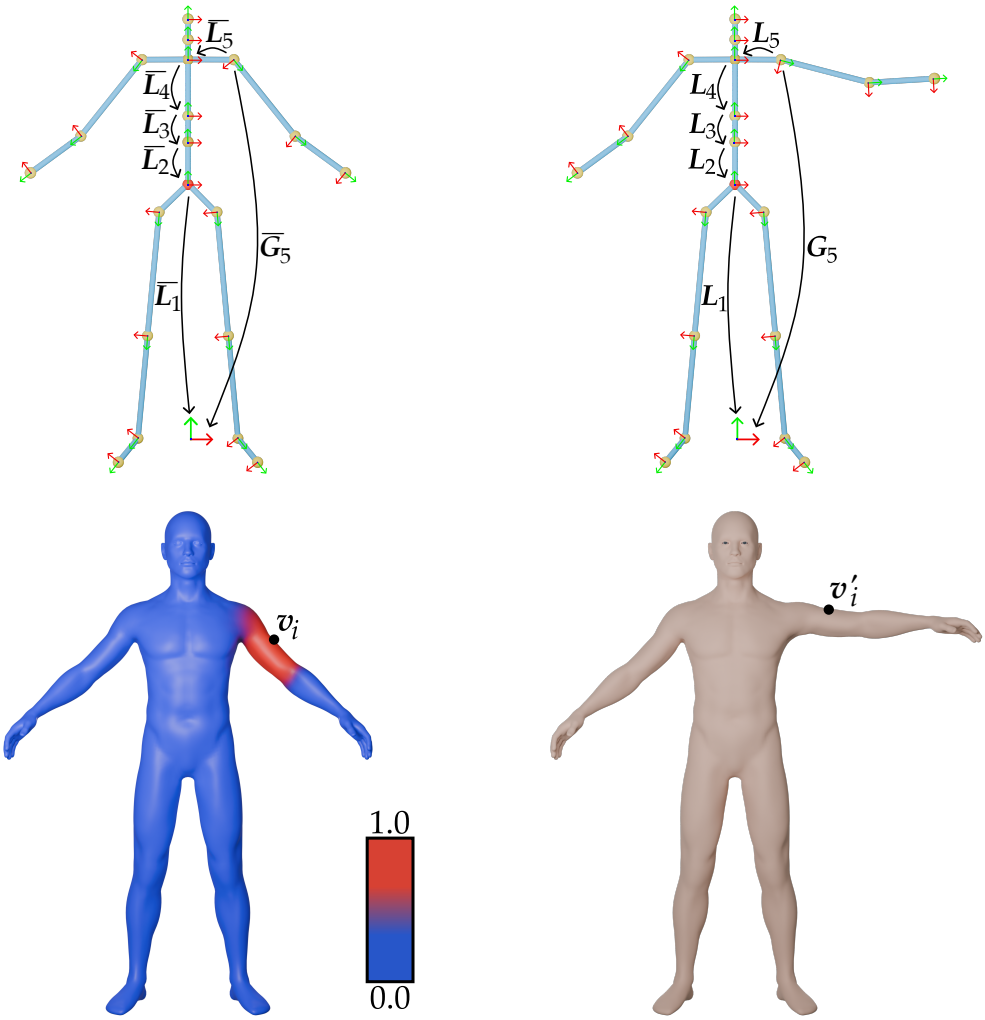


Figure 3.1: This image shows our template model (bottom) and a simplified (i.e., reduced number of joints) skeleton rig (top). The top row shows the skeleton in bind pose (left) and in animated pose (right). The bottom row shows the skinning weights for the left shoulder joint in the bind pose (left) and the textured mesh in the animated pose (right) of our template model. We can use local matrices $\boldsymbol{L}_j$ ($\overline{\boldsymbol{L}_j}$ in bind pose) to map from a child joint (yellow) to its parent's coordinate system or global transformation matrices $\boldsymbol{G}_j$ ($\overline{\boldsymbol{G}_j}$ in bind pose) to map into the world coordinate system. The root joint (red sphere) has no parent joint; hence, its local transform directly maps into the world coordinate system.

3.1.2 Shape Adjustments

We employ a highly detailed human model as a template, serving as a prior that incorporates knowledge about the human body into the fitting process. As we want to scan people of any gender and with different body sizes and shapes, a template model, depicting just one specific shape (modeled by an artist), matches only little with the scanned person in general. To address this, we want a *statistical* template model representing the space of human body shapes. Achenbach et al. [Ach+17] and Wenninger et al. [Wen+20] employ such a statistical template model to obtain a good initial approximation of the shape of the person before adding small details. This is done by first fitting the existing template to a dataset consisting of different people with a variety of shapes and sizes, e.g., the CAESAR dataset [RDP99]. After fitting the model to all the contained scans, we obtain fitted models $\boldsymbol{V}_1, \ldots, \boldsymbol{V}_{n_C}$ with different shapes but with the same topology, meaning all meshes still have the same triangulation and corresponding vertices between the different meshes are located at the same semantic location. This property is crucial for computing statistics on the fitted models, because it allows us to determine how the positions of those vertices vary across individuals.

Since the goal is to use the statistical model to compute a coarse approximation of the scanned person, a low-dimensional linear model is sufficient to capture the variance between the different bodies. Consequently, morphable models based on principal component analysis (PCA, [BV99]) are often used. To compute this morphable model, we first construct a data matrix $\boldsymbol{X}$ from the meshes fitted to the dataset:

$$\boldsymbol{X} = \frac{1}{\sqrt{n_C - 1}} \left[\boldsymbol{V}_1 - \boldsymbol{\mu}, \ldots, \boldsymbol{V}_{n_C} - \boldsymbol{\mu}\right] \in \mathbb{R}^{3N_V \times n_C},$$

where $\boldsymbol{V}_i \in \mathbb{R}^{3N_V}$ is a vector of the i-th mesh containing the x-, y-, and z-coordinates of the N_V vertices, $\boldsymbol{\mu} \in \mathbb{R}^{3N_V}$ is the mean mesh of all fitted models, and n_C is the number of fitted models. After constructing the data matrix, we compute the covariance matrix

$$\boldsymbol{C} = \boldsymbol{X}\boldsymbol{X}^\mathsf{T}.$$

On this covariance matrix, we then perform an eigen-decomposition to obtain

$$\boldsymbol{C} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\mathsf{T},$$

where $\boldsymbol{U} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{3N_V}]$ contains the normalized $3N_V$ eigenvectors in descending order corresponding to their eigenvalues $\boldsymbol{\Lambda} = \operatorname{diag}(\lambda_1, \ldots, \lambda_{3N_V})$. Each eigenvalue λ_i describes the variance in the direction of its eigenvector $\boldsymbol{u}_i$, and the sum of all eigenvalues equals the total variance of the data. Thus, each eigenvalue reveals the importance of its corresponding eigenvector for representing the input data. Note that the matrix $\boldsymbol{\Lambda}$ contains only $P = \min(3N_V, n_C - 1)$ non-zero eigenvalues; the dimensions corresponding to zero eigenvalues therefore do not provide additional information.

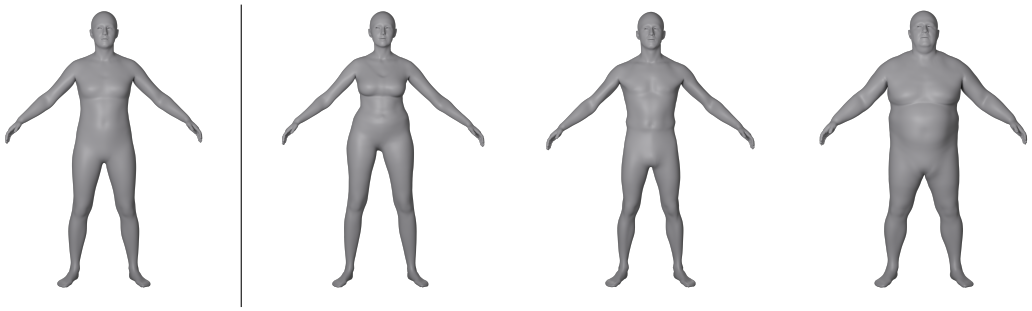


Figure 3.2: This image shows the PCA's mean mesh (left) and three exemplary meshes created with the generative model (right). The PCA was computed on the CAESAR dataset [RDP99]. While the generative model allows us to create diverse body shapes, the models lack fine-scale features and details.

In practice, only a small number of dimensions describe the majority of the information. Coupled with our objective of obtaining a coarse initial approximation, we reduce the dimensionality by taking only the first $d \ll P$ dimensions into account and construct a truncated PCA basis matrix $\boldsymbol{U}_d \in \mathbb{R}^{3N_V \times d}$. We can then create new shapes with the generative model by adjusting the parameters $\boldsymbol{\beta} \in \mathbb{R}^d$, which denote the shape coefficients in the truncated PCA space:

$$\boldsymbol{V} = \boldsymbol{\mu} + \boldsymbol{U}_d \boldsymbol{\beta}.$$

Starting from the mean mesh $\boldsymbol{\mu}$ (see Figure 3.2, left), we add the eigenvector deformations weighted by the parameters $\boldsymbol{\beta}$, resulting in a new body shape (see Figure 3.2, right).

The truncated version of the PCA basis matrix can only approximate the input data instead of reproducing it exactly. To ensure that the generative model explains sufficient variance of the input data, we can validate the expressiveness by computing:

$$\frac{\sum_{i=1}^{d} \lambda_i}{\sum_{i=1}^{P} \lambda_i},$$

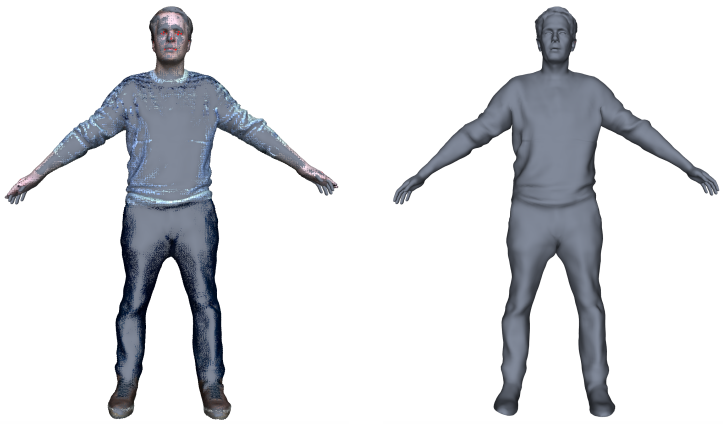
where d is the reduced number of PCA-dimensions and P is the full number of PCA-dimensions. Achenbach et al. [Ach+17] and Wenninger et al. [Wen+20] use at most 30 dimensions as this already represents over 99% of the variance of the input data.

3.2 Template Fitting

We now describe the template-fitting procedure, which deforms our template model to match a given 3D input (e.g., a point cloud; see Figure 3.3a). By fitting an already rigged template, the reconstructed model preserves the structural properties of the template, including the topology, the UV layout, and most importantly, the animation skeleton. Consequently, the fitted model can be animated in the same manner as the template model. The fitting process is



(a) Point Cloud *(b)* Initial Alignment *(c)* Coarse Fitting



(d) Fine-Scale Fitting *(e)* Final Geometry

Figure 3.3: Results of the different steps from the fitting procedure. Top row: raw point cloud (a), initial alignment (b), and coarse fitting (c); bottom row: fine-scale fitting (d) and final geometry (e). With every step, the model's alignment, pose, and shape become more refined. While the pose and shape match only roughly after the initial alignment, small folds in the clothes are represented after the fine-scale fitting.

typically performed in three steps: (1) an initial alignment step to register the model using a set of predefined landmarks (see Figure 3.3b), (2) a coarse fitting step to refine the pose and shape using closest point correspondences (see Figure 3.3c), and (3) a fine-scale fitting step to capture small geometric details (see Figure 3.3d).

In a first step, a 3D point cloud is generated from the input images using a photogrammetry software (e.g., Agisoft Metashape [Agi25]). The photogrammetry software detects and matches feature points in the images to compute intrinsic and extrinsic camera calibration parameters for all images. Using the computed calibrations and matched feature points, a dense point cloud $\mathcal{P} = \{\boldsymbol{p}_1, \ldots, \boldsymbol{p}_{N_P}\}$ is reconstructed, consisting of about $N_P \approx 4\text{M}$ points (see Figure 3.3a). For each point $\boldsymbol{p}_i \in \mathbb{R}^3$, the software outputs additional color information $\boldsymbol{c}_i \in \mathbb{R}^3$ and a normal vector $\boldsymbol{n}_i \in \mathbb{R}^3$.

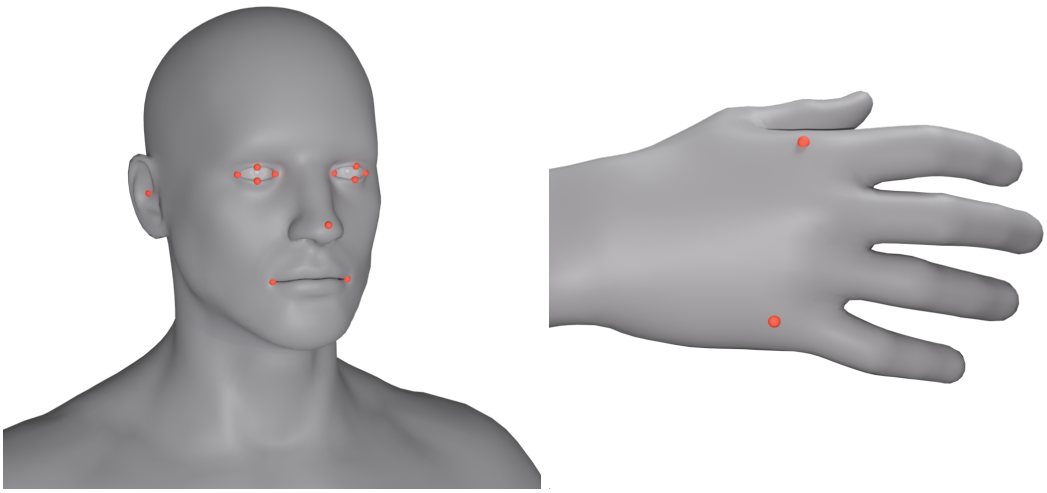


Figure 3.4: We use a set of predefined landmarks to guide the template fitting procedure. Landmarks correspond to identifiable keypoints (e.g., tip of the nose or eye contours) and are preselected once on the template model (red spheres).

Initial Alignment

The initial alignment of the template model and the point cloud is done using a set of predefined landmarks $\mathcal{L}$. These landmarks correspond to identifiable keypoints (e.g., tip of the nose, eye contours, mouth contours, or hand knuckles; see Figure 3.4 for examples) on the template mesh and the point cloud and are used as guidance throughout the template fitting process. While the landmarks on the template model are selected once and reused for all reconstructions, the landmarks for unstructured point clouds must be selected for each scan. These can either be selected manually [Ach+17] or detected automatically [Wen+20]. For the initial alignment of the template model and the point cloud, we define three different losses:

Global Alignment Loss Using the corresponding landmarks on the template and the point cloud, we compute a conformal transformation (consisting of translation, rotation, and uniform scaling) that minimizes the sum of squared distances between the corresponding landmarks in a least-squares sense. The resulting transform aligns the point cloud and the template in a common coordinate system:

$$L_{\text{global}}(s_g, \boldsymbol{R}_g, \boldsymbol{t}_g) = \frac{1}{|\mathcal{L}|} \sum_{i=1}^{|\mathcal{L}|} \left\| s_g \boldsymbol{R}_g \boldsymbol{l}_i + \boldsymbol{t}_g - \boldsymbol{v}_i \right\|^2, \tag{3.1}$$

where $s_g \in \mathbb{R}$ is a uniform global scaling factor, $\boldsymbol{R}_g \in SO(3)$ is the global rotation matrix orienting the point cloud landmarks to the template, $\boldsymbol{t}_g \in \mathbb{R}^3$ is the global translation vector positioning the point cloud landmarks onto

the template, $\boldsymbol{l}_i \in \mathbb{R}^3$ is the position of the i-th landmark in the point cloud, and $\boldsymbol{v}_i \in \mathbb{R}^3$ is the corresponding position of the template model's i-th vertex. Note that we use $\| \cdot \|$ to denote the ℓ_2-norm when applied to vectors and the Frobenius norm when applied to matrices.

Pose Loss Since the pose of the template model does not necessarily match the pose of the subject, we refine the alignment by adjusting the pose parameters (i.e., joint angles $\boldsymbol{\theta}$) of the template. We minimize the sum of squared distances between corresponding landmarks in a least-squares sense with respect to the template's pose parameters $\boldsymbol{\theta}$:

$$L_{\text{pose}}(\boldsymbol{\theta}) = \frac{1}{|\mathcal{L}|} \sum_{i=1}^{|\mathcal{L}|} \| \boldsymbol{l}_i - \text{skin}_v(\boldsymbol{v}_i, \boldsymbol{\theta}) \|^2, \tag{3.2}$$

where skin_v denotes the linear blend skinning function for single vertices of the template model (see Section 3.1.1). Optimizing this loss yields the joint angles $\boldsymbol{\theta}$ that best align the template's pose to the point cloud.

Shape Loss The shape parameters of the template model are optimized to account for differences in body shape between the template model and the subject. The shape model consists of a low-dimensional PCA space (e.g., d = 30 dimensions) computed on a large dataset containing a variety of human body shapes (e.g., 1700 scans from the CAESAR dataset [RDP99]). To match the template model's shape to the subject, we minimize the loss function

$$L_{\text{shape}}(\boldsymbol{\beta}) = \frac{1}{|\mathcal{L}|} \sum_{i=1}^{|\mathcal{L}|} \| \boldsymbol{l}_i - \boldsymbol{Q}_i(\boldsymbol{\mu} + \boldsymbol{U}_d\boldsymbol{\beta}) \|^2, \tag{3.3}$$

where $\boldsymbol{U}_d \in \mathbb{R}^{3N_V \times d}$ is the truncated PCA basis matrix, $\boldsymbol{\beta} \in \mathbb{R}^d$ are the shape coefficients, $\boldsymbol{\mu} \in \mathbb{R}^{3N_V}$ is the mean shape of the PCA model, and $\boldsymbol{Q}_i \in \mathbb{R}^{3 \times 3N_V}$ is a matrix that selects the vertex corresponding to landmark $\boldsymbol{l}_i$ from the full vertex list.

Shape Regularization To prevent overfitting, we add a Tikhonov regularization term that penalizes deformations along low-variance principal components more strongly. Our shape regularization is computed as:

$$L_{\text{tik}}(\boldsymbol{\beta}) = \frac{1}{d} \| \boldsymbol{\Gamma}\boldsymbol{\beta} \|^2, \tag{3.4}$$

where d is the number of dimensions used in the shape model (e.g., 30) and $\boldsymbol{\Gamma} \in \mathbb{R}^{d \times d}$ is a diagonal matrix defined as $\boldsymbol{\Gamma} = \text{diag}\left(\frac{1}{\sqrt{\lambda_1}}, \ldots, \frac{1}{\sqrt{\lambda_d}}\right)$ with λ_i being the eigenvalues of the PCA covariance matrix.

Total Loss The global alignment term L_{global} (Equation (3.1)), the pose term L_{pose} (Equation (3.2)), and the shape term L_{shape} (Equation (3.3)) can be unified into a single landmark loss L_{lm}:

$$L_{\text{lm}}(s_g, \boldsymbol{R}_g, \boldsymbol{t}_g, \boldsymbol{\theta}, \boldsymbol{\beta}) = \frac{1}{|\mathcal{L}|} \sum_{i=1}^{|\mathcal{L}|} \left\| s_g \boldsymbol{R}_g \boldsymbol{l}_i + \boldsymbol{t}_g - \boldsymbol{Q}_i \operatorname{skin}(\boldsymbol{\mu} + \boldsymbol{U}_d \boldsymbol{\beta}, \boldsymbol{\theta}) \right\|^2, \quad (3.5)$$

where skin denotes the linear blend skinning function for the full list of vertices (see Section 3.1.1).

Our overall loss for the initial alignment is then:

$$L_{\text{initial}}(s_g, \boldsymbol{R}_g, \boldsymbol{t}_g, \boldsymbol{\theta}, \boldsymbol{\beta}) = \tau_{\text{lm}} L_{\text{lm}}(s_g, \boldsymbol{R}_g, \boldsymbol{t}_g, \boldsymbol{\theta}, \boldsymbol{\beta}) + \tau_{\text{tik}} L_{\text{tik}}(\boldsymbol{\beta}),$$

where τ_{lm} and τ_{tik} control the influence of each term. We set τ_{lm} to 0.5 and τ_{tik} to 10^{-3} and optimize the objective for two iterations, where each iteration alternates between optimizing global, pose, and shape alignment. The global alignment is computed in a closed-form manner [Hor87], the pose is optimized using inverse kinematics [Bus04], and shape parameters are optimized via linear least-squares on the per-landmark distances between template mesh and point cloud. The result is an initial alignment of the point cloud and the template model, with roughly matching orientation, pose, and shape (see Figure 3.3b). This model is further refined in the next steps.

Changing the shape of the mesh invalidates the joint positions of the initial animation skeleton. For example, if we fit a person smaller than our template model, changing the shape might lead to joints that lie outside the mesh. Therefore, in each iteration after updating the mesh by minimizing our shape loss, we compute new joint positions using precomputed mean-value coordinates [JSW05] defined on the template model. This step ensures that each joint maintains its relative position inside the body mesh compared to the template model.

Coarse Fitting

After this rough initial fit, additional closest-point correspondence constraints are introduced to further refine the alignment. Instead of relying solely on landmarks, correspondences between the point cloud and the template are established by finding, for each point in the point cloud, its closest point on the template model. As the computation of closest points is sensitive to noise and misalignments, we reject unreliable correspondence pairs based on distance and normal direction. Similar to the initial alignment, we minimize the sum of squared distances between all correspondences by alternately optimizing the global alignment, pose, and shape parameters of the template model using the following loss function:

$$
\begin{aligned}
&L_{\text{cpc}}\left(s_g, \boldsymbol{R}_g, \boldsymbol{t}_g, \boldsymbol{\theta}, \boldsymbol{\beta}\right) \\
&= \frac{1}{\sum_{c=1}^{|\mathcal{C}|} w_c} \sum_{c=1}^{|\mathcal{C}|} w_c \left\| s_g \boldsymbol{R}_g \boldsymbol{p}_c + \boldsymbol{t}_g - \boldsymbol{B}_c \operatorname{skin}(\boldsymbol{\mu} + \boldsymbol{U}_d \boldsymbol{\beta}, \boldsymbol{\theta}) \right\|^2,
\end{aligned} \tag{3.6}
$$

where $\mathcal{C}$ is the set of all correspondences, $\boldsymbol{p}_c \in \mathbb{R}^3$ is a point in the point cloud, w_c is a per-correspondence weight, $\boldsymbol{B}_c \in \mathbb{R}^{3 \times 3N_V}$ is a matrix containing barycentric coordinates to select the closest point on the template model corresponding to $\boldsymbol{p}_c$ from the full list of vertices, and skin denotes the linear blend skinning function for all vertices in a mesh (see Section 3.1.1). We add this correspondence term as we only have a small set of landmarks (see Figure 3.4), leaving most of the model's vertices unconstrained. By computing closest point correspondences, we add significantly more constraints, forcing the model to adjust its shape and pose in all of the previously unaffected areas (e.g., the torso).

The combination of this closest point correspondence loss L_{cpc} (Equation (3.6)), the landmark loss L_{lm} (Equation (3.5)), and the Tikhonov regularization L_{tik} (Equation (3.4)) forms the overall loss function for this coarse fitting step:

$$
\begin{aligned}
&L_{\text{coarse}}\left(s_g, \boldsymbol{R}_g, \boldsymbol{t}_g, \boldsymbol{\theta}, \boldsymbol{\beta}\right) \\
&= \tau_{\text{lm}} L_{\text{lm}}\left(s_g, \boldsymbol{R}_g, \boldsymbol{t}_g, \boldsymbol{\theta}, \boldsymbol{\beta}\right) + \tau_{\text{cpc}} L_{\text{cpc}}\left(s_g, \boldsymbol{R}_g, \boldsymbol{t}_g, \boldsymbol{\theta}, \boldsymbol{\beta}\right) + \tau_{\text{tik}} L_{\text{tik}}(\boldsymbol{\beta}),
\end{aligned}
$$

where τ_{lm}, τ_{cpc}, and τ_{tik} are weights controlling the influence of each term.

Again, we minimize the loss function in an alternating manner for global, pose, and shape alignment. In this step, we use $\tau_{\text{lm}} = 0.4$, $\tau_{\text{cpc}} = 1$, and $\tau_{\text{tik}} = 10^{-3}$ and optimize the objective for five iterations. In each iteration, we recompute the correspondences with the result from the previous iteration. After this step, the model coarsely matches the point cloud (see Figure 3.3c). This step finishes the global alignment and pose optimization. However, due to the low-dimensional shape model, fine geometric details (e.g., folds in clothing) are not yet incorporated into the model. We therefore further refine the model's geometry in the next step using non-rigid registration.

Fine-Scale Fitting

To add small details into the model, we apply non-rigid registration [Ach+17; ARV07; BR07]. Instead of adjusting the model's shape coarsely with the low-dimensional statistical model, we now optimize directly for new vertex positions $\boldsymbol{V}$. To do this, we minimize the loss function:

$$
\begin{aligned}
L_{\text{fine}}(\boldsymbol{V}) &= \tau_{\text{lm}}^{\text{fine}} L_{\text{lm}}^{\text{fine}}(\boldsymbol{V}) + \tau_{\text{cpc}}^{\text{fine}} L_{\text{cpc}}^{\text{fine}}(\boldsymbol{V}) + \tau_{\text{reg}} L_{\text{reg}}(\boldsymbol{V}), \qquad (3.7)\\
L_{\text{lm}}^{\text{fine}}(\boldsymbol{V}) &= \frac{1}{|\mathcal{L}|} \sum_{i=1}^{|\mathcal{L}|} \| \boldsymbol{l}_i - \text{skin}_v(\boldsymbol{v}_i, \boldsymbol{\theta}) \|^2,\\
L_{\text{cpc}}^{\text{fine}}(\boldsymbol{V}) &= \frac{1}{\sum_{c=1}^{|\mathcal{C}|} w_c} \sum_{c=1}^{|\mathcal{C}|} w_c \| \boldsymbol{p}_c - \text{skin}_C(\boldsymbol{v}_c, \boldsymbol{\theta}) \|^2,\\
L_{\text{reg}}(\boldsymbol{V}) &= \frac{1}{\sum_{e\in\mathcal{E}} A_e} \sum_{e\in\mathcal{E}} A_e \| \Delta^e \boldsymbol{v}(e) - \boldsymbol{R}_e \Delta^e \bar{\boldsymbol{v}}(e) \|^2,
\end{aligned}
$$

where $\boldsymbol{v}_i$ is the position of the i-th vertex of the model, $\boldsymbol{\theta}$ are the pose parameters from the coarse fitting step, skin_v and skin_C denote the linear blend skinning functions for single vertices and barycentric points, respectively (see Section 3.1.1), w_c is a per-correspondence weight used to downweight unreliably reconstructed body parts in the point cloud (e.g., hands), $\boldsymbol{v}_c$ is a point in a triangle of the model defined by barycentric coordinates corresponding to point $\boldsymbol{p}_c$ in the point cloud, $\mathcal{E}$ is the set of edges in the template mesh, A_e is the edge area of edge e, $\boldsymbol{R}_e \Delta^e \bar{\boldsymbol{v}}(e) \in \mathbb{R}^3$ is the undeformed edge-Laplacian rotated by $\boldsymbol{R}_e \in SO(3)$ to align it with the deformed edge-Laplacian $\Delta^e \boldsymbol{v}(e) \in \mathbb{R}^3$ (see [AZB15] for details).

The first term $L_{\text{lm}}^{\text{fine}}$ minimizes the sum of squared distances between the predefined landmarks and guides the non-rigid fitting optimization, the second term $L_{\text{cpc}}^{\text{fine}}$ minimizes the sum of squared distances between all correspondences, which are recomputed in every iteration, to iteratively add more detail to the geometry. The third term L_{reg} is a regularization term that penalizes differences in curvature from the undeformed state $\bar{\boldsymbol{V}}$ to the deformed state $\boldsymbol{V}$, acting as a bending energy to prevent overfitting to the point cloud. The weights $\tau_{\text{lm}}^{\text{fine}}$, $\tau_{\text{cpc}}^{\text{fine}}$, and τ_{reg} control the influence of each term. This non-linear loss function is minimized using iterative block coordinate descent until convergence. $\tau_{\text{cpc}}^{\text{fine}}$ is fixed at 1. The weights $\tau_{\text{lm}}^{\text{fine}}$ and τ_{reg} are progressively decreased to strengthen the influence of the correspondences, as in each iteration the correspondences must satisfy more restrictive matching criteria (e.g., smaller normal deviation and distance). We initialize $\tau_{\text{lm}}^{\text{fine}}$ to 1 and scale it by a factor of 0.6 per iteration, whereas τ_{reg} is progressively decreased from 10^{-3} to 10^{-9} using a factor of 0.15. When decreasing the weights, the undeformed state $\bar{\boldsymbol{V}}$ is updated with the previously found solution $\boldsymbol{V}$ for the next iteration. After performing the fine-scale fitting, the result closely matches the point cloud (see Figure 3.3d) and contains fine details like folds in the clothes (see Figure 3.3e).

3.3 Multi-Camera Rig-Based Pipeline

The pipeline described in this section is based on the multi-camera rig-based avatar reconstruction approach of Achenbach et al. [Ach+17]. It serves as the

high-quality baseline in the quantitative evaluation of our avatar reconstruction system (Section 5.2.1) and the corresponding user study (Section 5.3). Achenbach et al. use a single-shot multi-camera rig to simultaneously capture high-quality images of a subject standing in A-pose from different viewing angles. The setup consists of a body scanner and a separate face scanner. The body scanner uses 40 DSLR cameras arranged in eight columns in a circular pattern. The face scanner uses an additional eight DSLR cameras arranged in a quarter circle to capture high-resolution images of the face. After capturing the body and face images, the two sets of images are processed separately using a photogrammetry software (Agisoft Metashape [Agi25]) to compute two high-quality point clouds: one for the body and one for the face. Then, on both point clouds, landmarks are manually selected as guidance for the template fitting algorithm, as described in Section 3.2. Achenbach et al.'s reconstruction pipeline is divided into two different parts: (1) the body reconstruction and (2) the face reconstruction.



Figure 3.5: Exemplary result of the multi-camera rig-based pipeline proposed by Achenbach et al. [Ach+17]. The model was provided by Wenninger et al. [Wen+20].

Body Reconstruction In the first step, the template model is fitted to the body point cloud using a template fitting procedure similar to the one described in Section 3.2. After the fine-scale deformation, the resulting model is passed to Agisoft Metashape to compute a texture using the given UV layout of the template model. Due to occlusions during the scanning process, the photogrammetry algorithm reconstructs hands and fingers only unreliably. Therefore, after generating the texture using Agisoft Metashape, Achenbach et al. replace the hand texture with the closest matching texture in a hand

texture database and merge it into the avatar's final texture using Poisson image editing [PGB03].

Face Reconstruction After fitting the template model to the body scan, the face scan is used to refine the facial region using the same fitting procedure guided by the preselected facial landmarks. However, Achenbach et al. noticed that after the correspondence-driven fine-scale deformation, the mouth of the fitted model opened during the fitting procedure, even though all scans were performed with the mouth closed and in a neutral facial expression. They therefore added another penalty term to L_{fine} (Equation (3.7)):

$$L_{\text{mouth}}(\boldsymbol{V}) = \frac{1}{|\,\mathcal{M}\,|} \sum_{(\boldsymbol{v}_u, \boldsymbol{v}_l) \in \mathcal{M}} \| \boldsymbol{v}_u - \boldsymbol{v}_l \|^2 ,$$

where $\mathcal{M}$ is a set of preselected upper ($\boldsymbol{v}_u$) and lower ($\boldsymbol{v}_l$) lip vertex pairs. This penalty term penalizes the sum of squared distances between those upper and lower lip vertices, and therefore effectively prevents the mouth from opening during the fine-scale deformation. As before, after the fitting procedure, the resulting model is passed into Agisoft Metashape to compute a high-resolution texture for the facial region. Similar to the hand texture replacement, the facial texture is merged into the final avatar's texture using Poisson image editing [PGB03]. The template model also contains auxiliary meshes for eyes and teeth. The template teeth mesh is transferred to the final avatar by optimizing for anisotropic scaling, rotation, and translation between the undeformed and deformed mouth region. The template eyes are transferred by optimizing for isotropic scaling (to avoid introducing shearing of the eye meshes), rotation, and translation between the undeformed and deformed eye regions. In addition to the static features, the template model contains a set of blendshapes for facial animation. As the subject is only scanned with a neutral facial expression, the blendshapes are transferred to the final avatar using deformation transfer [SP04]. An exemplary result of this pipeline is shown in Figure 3.5.

3.4 Smartphone-Based Pipeline

The pipeline presented in this section is based on the smartphone-based avatar reconstruction of Wenninger et al. [Wen+20]. Their pipeline is similar to that of Achenbach et al. [Ach+17], but is adjusted to work with smartphone videos as input.

Input Data Similar to Achenbach et al., Wenninger et al. perform a two-step scanning procedure. In the first step, a body video of a subject standing in A-pose is recorded by slowly circling around the subject with a smartphone camera. In the first round, the upper body (from head to hips) is captured, and in the second round, the lower body (from belly to feet) is captured. In the



Figure 3.6: Exemplary result of the smartphone-based pipeline proposed by Wenninger et al. [Wen+20].

second scanning step, Wenninger et al. capture a head scan video of the subject by slowly circling around the head at a closer distance than in the body scan. The head scan is necessary to ensure sufficient pixel coverage of the face/head in the videos for detailed facial geometry and texture reconstruction.

After capturing the body and head videos, Wenninger et al. extract two sets of frames (75 body images, 50 head images). The selected frames should be evenly distributed around the scanning path. To ensure this, Wenninger et al. [Wen+20] use an optical flow analysis [Far03], resulting in a 2D flow field which represents the movement between two consecutive frames. A frame is selected only if the average length of the 2D flow vectors exceeds a certain threshold, ensuring sufficient motion since the previously selected frame. Because the selected frame may be blurry due to the motion, its sharpness is also evaluated. The sharpness is measured as the variance of the Laplacian of the input image. The final frame is selected as the sharpest frame in a k-neighborhood around the initially selected frame.

After extracting the frames from the input videos, following Achenbach et al. [Ach+17], the body and head images are passed separately to Agisoft Metashape to compute body and head point clouds. The reconstructed point clouds are noisier and less complete than those generated from multi-camera rig images. This is due to slight motion during the scan procedure and the lower-resolution input images.

Fitting Procedure For the fitting procedure, Wenninger et al. do not select landmarks manually but use OpenPose [Cao+19] – a framework for detecting 2D body and face landmarks in images – to automatically detect landmarks

on the extracted frames (15 full-body landmarks, and 68 face landmarks). The detected 2D landmarks are then back-projected onto the 3D point clouds using the camera calibration computed by Agisoft Metashape to obtain 3D landmarks. As before, the 3D landmarks guide a template-fitting procedure as described in Section 3.2 to fit the template model to the body and head point clouds in a two-step process. However, due to lower-quality point clouds, stronger regularization is necessary to prevent overfitting to noise. Consequently, the regularization weight τ_{reg} in Equation (3.7) is only decreased to 10^{-7}.

Texture Generation Instead of relying on Agisoft Metashape to generate body and head textures, Wenninger et al. use a different approach. In a first step, they compute partial textures from each image, and in a second step, combine those partial textures into unified textures using a graph-cut algorithm [All+18a]. As in the previous pipeline, the resulting separate head texture is then merged into the body texture using Poisson image editing [PGB03]. Due to poor scanning quality of eyes and hands, Wenninger et al. use the template texture information for those regions instead and adapt them using histogram matching [GW08]. An exemplary result of the smartphone-based pipeline is shown in Figure 3.6.

4 RELATED WORK

Corresponding Publications *This chapter is based on the related work sections of the following publications:*

1. T. Menzel, E. Wolf, S. Wenninger, N. Spinczyk, L. Holderrieth, C. Wienrich, U. Schwanecke, M. E. Latoschik, and M. Botsch. "Avatars for the Masses: Smartphone-Based Reconstruction of Humans for Virtual Reality". In: *Frontiers in Virtual Reality* 6 (2025). DOI: 10.3389/frvir.2025.1583474
2. F. Runte, T. Menzel, U. Schwanecke, and M. Botsch. "Compensating Motion-Induced Errors in Smartphone-Based VR Avatar Reconstruction". In: *Proceedings of the 2025 31st ACM Symposium on Virtual Reality Software and Technology*. 2025. DOI: 10.1145/3756884.3765995

In this chapter, we describe the mechanisms and implications of representing oneself through an avatar in VR (Section 4.1), before discussing different approaches to generating realistic avatars (Section 4.2). In the following, we restrict our discussion to avatars that are *personalized* (as opposed to generic), *realistic* (as opposed to stylized), and *full-body* (as opposed to head-only or upper-body-only), because these are the most challenging with regard to the outlined desiderata.

4.1 Avatars for Self-Representation in Virtual Reality

The egocentric embodiment of avatars for self-representation in VR [Sla+10] can positively impact the user experience (UX) of virtual environments [Mot+23]. This includes improving the key psychometric properties of VR, such as the sense of presence [SFW17; Wal+18; Wol+21], or intensifying emotional responses to virtual content [Gal+21; Wal+18]. Other advantages may include improved spatial perception [Ley+11; Moh+10], reduced cognitive load [Ste+16], or higher performance and accuracy [JH16; Pas+20] when performing tasks in VR.

A crucial aspect in evaluating the effectiveness of avatar embodiment is the sense of embodiment (SoE), consisting of the feeling of owning (ownership), controlling (agency), and being located within (self-location) a virtual body in a virtual environment [de 11; KGS12]. Previous work has shown that realistic and personalized avatars increase the SoE towards the avatar [Fie+23; Fie+26; Rad+23; Sal+23; Wal+18] and thus contribute to an overall plausible VR experience [LW22].

A recent systematic review by Deng et al. [DBH25] further supports the influence of realistic and personalized avatars on the SoE. They analyzed more than 43 studies published over the last ten years. However, they focused on avatar-user incongruities and provided recommendations for the key factors

influencing the SoE. They identified three main factors: (i) Most of the studies reported a higher sense of embodiment if the used avatars closely resembled the user's real-world appearance, with the important distinction that this is even true for low-cost personalization methods. (ii) The likability of the avatar in the VR exposure can have a positive influence on user engagement. More likable, i.e., idealistic, avatars tend to be prioritized over more realistic avatars. (iii) Lower body perspectives (e.g., body size and hands) are significantly more important than facial features for the SoE. Altogether, these findings highlight the importance of balancing realism, personalization, and user preferences in avatar design to positively affect users' sense of embodiment.

Photorealistic and personalized avatars are particularly valuable for maintaining the user's identity, which is beneficial in social VR experiences [AI21; Mys22; Yoo+19] or applications supporting mental health [Döl+22; Sam+21; Tur+21]. Previous work has also shown that self-related cues through avatar embodiment and personalization can significantly increase self-identification with the avatar [Fie+24; Fie+26], potentially maintaining a more accurate self-perception in VR, even in body-swap paradigms [Döl+24]. However, a realistic personalization of avatars can also harm UX, as their human-like realism combined with their high affinity to the user can potentially trigger uncanny valley effects, leading to negative emotional responses such as eeriness towards the avatars [Döl+23; MMK12].

The described effects are confirmed in a recent study by Liu et al. [LHY25], and they also create an important link to the familiarity with the used avatars. On the one hand, a higher degree of appearance realism increases embodiment factors, resulting in a stronger connection between users and their avatars; on the other hand, especially for self-avatars or avatars of familiar people, users tend to be more critical and are more dissatisfied. This suggests the importance of balancing realism with likability when reconstructing realistic personalized avatars.

Overall, there is comprehensive evidence of notable effects of photorealistic, personalized avatars on important user states. Consequently, we will employ representative psychometric measures for a prominent selection of the aforementioned effects of avatars to evaluate the 3D reconstruction quality achieved with our developed system. Therefore, Section 5.3 reports on a user study evaluating our avatars with respect to the sense of embodiment, plausibility, and a potential uncanny valley effect. In addition, this user study also evaluates the general usability and user satisfaction of the smartphone front-end to ensure appropriate ease of use and user satisfaction.

4.2 Realistic Avatar Reconstruction

The growing demand for virtual avatars stimulated extensive research in scanning-based avatar reconstruction in recent years. This section provides an overview of related approaches and the current state of research in this area. We first present approaches based on the type of representations used

for the resulting avatar models, and then describe methods categorized by the scanning device.

4.2.1 Representation of Avatars

In the last few years, two types of avatar representations have overwhelmingly defined the developments in personalized and realistic avatar reconstruction. With the introduction of Neural Radiance Fields (NeRF, [Mil+21]), many shifted the focus from mesh-based avatars to avatars based on NeRFs [Jia+22; Jia+23; Liu+21; Pan+26; Pen+21; Wan+23; Wan+24; Wen+22; Yu+23; Zhe+22; Zhe+23], because those neural representations are capable of reconstructing fine details since they are not restricted to a fixed mesh topology. They are therefore better suited for reconstructing clothing and hair, although requiring long training (e.g., four hours for [Wan+24], and 25 hours for [Zhe+22]) and slow inference times (e.g., only 25.6 fps for [Pan+26], 15 fps for [Jia+23], or 0.2 fps for [Zhe+22]).

With the introduction of 3D Gaussian Splatting (3DGS, [Ker+23]), numerous approaches make use of it [Che+25a; Don+25; Guo+25; Hu+24; Jia+25; Li+24; LKJ24; Mor+24a; MSS25; Pau+25; Pen+25; Sha+24; Son+26; Wen+24; Xu+25; Zha+25; Zha+26], as they offer noticeably faster inference compared to NeRF-based methods. Starting from a mesh or a point cloud, these approaches use a mixture of 3D Gaussians to represent the avatar, enabling the reconstruction of fine details (e.g., wrinkles or hair). The appearance of these Gaussians is view-dependent, which further enhances realism by accounting for occlusion and perspective effects that vary with the camera angle. Early 3DGS-based approaches require significant time to compute Gaussians [Hu+24; Mor+24a; Wen+24]; however, new methods reduce this to minutes [Che+25a; Don+25; Lei+24; Pau+25; Sha+24; Shi+25]. Furthermore, recent approaches achieve 150–300 fps in rendering performance, which allows the use of these models for real-time applications [HHL24; Lei+24; Sha+24]. However, as current VR devices typically render at 70–120 fps, the usage for VR applications remains a challenging task, as the rendered scenes typically consist of more than just one avatar (e.g., multiple avatars and the surrounding environment). Iandola et al. [Ian+25] showed that it is necessary to reduce the number of Gaussians to make 3DGS-avatars usable on VR devices. However, reducing the number of Gaussians also reduces the rendering quality. The view-dependence of Gaussians, typically a key advantage of this representation, also poses challenges for VR. If the training data lacks sufficient coverage of possible views and poses, novel views create artifacts that undermine the realism of the avatar, as we show in Sections 5.2.2 and 6.2.1. This is a particularly challenging limitation, since in multi-user social VR applications, there is no control over viewing directions and avatar poses, which can quickly lead to visual artifacts.

More recent approaches target this problem by combining Gaussian avatar reconstruction with diffusion-based models [Che+25b; Don+25; SM25; Tan+25a; Tan+25b]. From a reference video or image, they generate pose-conditioned multi-view videos or images to generate unobserved training views in the

first step. After that, it is possible to reconstruct more stable Gaussian avatars even from poses and views not contained in the originally captured data. While these approaches address sparse input when using 3DGS, hallucinating novel views from multiple directions raises questions about view consistency (textures may shift from one view to another, or geometric details may not align properly).

In contrast, many approaches still employ mesh-based representations for the resulting avatar models, as mesh-based models are highly efficient in terms of rendering speed, can be easily animated into novel poses, and are view-consistent [Ach+17; All+18a; All+18b; Wan+25; Wen+20]. They are widely used to represent 3D models and characters, making them compatible with current game engines and 3D programs.

Our goal is to generate realistic, VR-ready full-body avatars with broad usability and applicability to democratize personalized avatars in general. For this, the disadvantages of 3DGS-based methods outweigh their benefits. Ultimately, *users* will control their avatar, leading to unpredictable new poses and views. Due to the guaranteed spatial consistency and the broad compatibility of mesh-based 3D models with current game engines (e.g., Unity, Unreal Engine), we utilize a mesh-based representation for our avatars.

4.2.2 Input Devices

Many approaches for avatar reconstruction use expensive multi-camera rigs consisting of up to 100 high-end cameras to capture images of a person simultaneously from all perspectives [Ach+17; Che+25a; FRS17; Kwo+23; Ma+21; Mor+24b; Pan+24; Sal+23; She+24]. Although these approaches yield visually appealing results, the capture method is only available in well-equipped laboratories due to the high cost of the scanner setup.

To reduce costs and make avatar reconstruction more accessible, monocular cameras or smartphones could be used instead of multi-camera rigs [All+18a; All+18b; All+19; Wan+25; Wen+20; Yan+24]. Early approaches by Alldieck et al. [All+18a; All+18b; All+19] capture monocular videos of a subject and use silhouette constraints to fit a mesh-based avatar. Fitting to the silhouette of each frame makes the pipeline more robust, but reduces geometric quality because concave details do not influence the silhouette. This is most visible in the face region. Wenninger et al. [Wen+20] (see Section 3.4) address this by incorporating a separate close-up of the head into the avatar reconstruction, producing a quality that is objectively quite close and subjectively similar to those of camera rigs [Bar+21]. However, their method is rather complicated to operate, is intended for controlled indoor environments, and relies on commercial components, which prevents widespread use by researchers and non-experts.

More research on avatars from monocular capture has been conducted in recent years. New approaches improve visual quality [Wan+25; Yan+24] but often require long training times (e.g., 2 days [Wan+25]).

Our goal is to democratize the use of personalized, realistic avatars; i.e., we want a method that quickly produces realistic avatars from low-cost hardware. We therefore build our method on the approach of Wenninger et al. [Wen+20], which we advance in several important aspects. First, our smartphone application visually guides the user through the capture process, thereby ensuring high-quality input data. Second, we technically improve the data acquisition, image pre-processing, and template fitting, leading to more accurate and more robust avatar reconstructions. Third, we replace the commercial components of Wenninger et al. [Wen+20] with carefully selected non-commercial alternatives, allowing us to make our system publicly available. Finally, we evaluate our approach (i) by qualitative and quantitative comparisons to state-of-the-art avatar reconstruction methods, and (ii) in terms of a carefully designed user study, in which first-time users successfully reconstruct and evaluate avatars.

5 AVATARS FOR THE MASSES: SMARTPHONE-BASED RECONSTRUCTION OF HUMANS FOR VIRTUAL REALITY

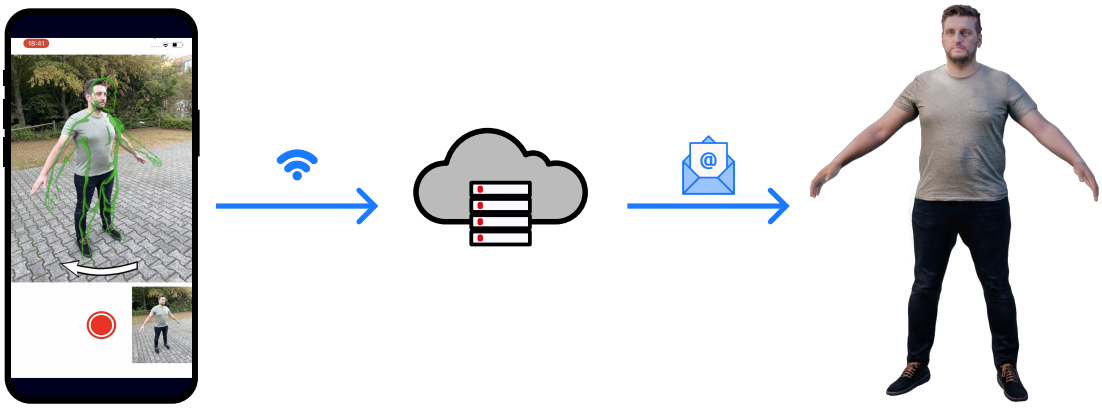


Figure 5.1: Avatars for the Masses enables non-experts to easily create realistic full-body avatars. Our iOS application visually guides through the scanning process, and our server generates VR-ready avatars in 20 minutes.

Now that we have introduced statistical template models and the concept of template fitting, and reviewed the current state of the art in avatar reconstruction, this chapter presents our fully automatic avatar reconstruction system *Avatars for the Masses*. Instead of relying on expensive and location-bound hardware, we leverage the ubiquity of smartphones to make avatar reconstruction more affordable. Modern smartphones offer high-quality cameras, possibilities for visual feedback during the scanning routine, and are portable and lightweight devices, allowing their use in arbitrary locations. This chapter first describes the proposed avatar reconstruction system in detail. We then quantitatively and qualitatively evaluate the resulting avatars against the two baseline approaches (see Sections 3.3 and 3.4) and recent state-of-the-art NeRF- and 3DGS-based methods. Finally, we present the results of our user study demonstrating the effectiveness of our reconstruction system. In this study, non-expert first-time users scanned other participants using our smartphone app, after which the scanned participants embodied and rated their reconstructed avatars in VR.

Individual Contribution *The template fitting codebase is based on the work of Stephan Wenninger from the paper [Wen+20]. In collaboration with Stephan Wenninger, I adapted the code to work with our new template model. Additionally, I implemented the remaining steps of the processing pipeline, including segmentation, photogrammetry, offscreen rendering, landmark detection, and texture transfer. The*

iOS scanning application and the server were developed with Niklas Spinczyk in a joint effort. The user study to evaluate the system was designed by Erik Wolf and implemented and conducted by Erik Wolf and Lena Holderrieth. I supported them with the integration of the reconstructed avatars into the VR application.

Corresponding Publication *This chapter is based on the following publication:*

> T. Menzel, E. Wolf, S. Wenninger, N. Spinczyk, L. Holderrieth, C. Wienrich, U. Schwanecke, M. E. Latoschik, and M. Botsch. "Avatars for the Masses: Smartphone-Based Reconstruction of Humans for Virtual Reality". In: *Frontiers in Virtual Reality* 6 (2025). DOI: 10.3389/frvir.2025.1583474

5.1 Avatar Reconstruction

Our approach extends and improves the work of Wenninger et al. [Wen+20]. To provide context for our improvements, we briefly recap the main steps of their smartphone-based pipeline (see Section 3.4 for details). They record two videos of the subject: a *body video*, which circles around the person twice to capture both the lower and the upper body, and a *head video*, which circles around the head in a close-up manner to capture facial details. From these two videos, individual frames are extracted and fed into Agisoft Metashape [Agi25], a commercial photogrammetry reconstruction tool, resulting in two point clouds: one for the body and one for the head. A template mesh is then fitted to the point clouds in a two-step process: The template is first fitted to the body point cloud (for the overall shape) and then to the head point cloud (for fine-tuning facial features). Landmarks detected by OpenPose [Cao+19] guide the template fitting process. In a final step, the avatar texture is generated from the input images.

Our approach, as outlined in Figure 5.2, introduces guided smartphone-based data capturing (Section 5.1.1) and a fully automatic server-based reconstruction pipeline (Section 5.1.2). In the following, we describe the components of both phases and point out the main contributions and technical improvements compared to Wenninger et al. [Wen+20].

5.1.1 Smartphone-Based Data Acquisition

Analogous to Wenninger et al. [Wen+20], we capture people by performing (i) a full-body scan in A-pose and (ii) a close-up head scan using a smartphone (see Figure 5.2, top left). However, our approach differs in the kind of data that is captured and how the user performs the scanning.

Capturing Videos vs Images

Videos captured with current smartphone cameras are compressed using H.264 or H.265. These algorithms are optimized for viewing each video frame for

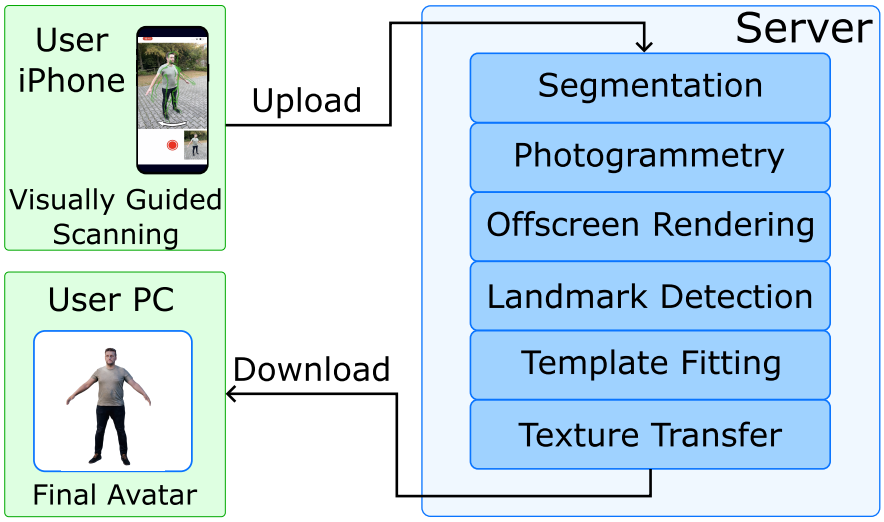


Figure 5.2: The user scans a subject with our smartphone application (top left). The captured images are uploaded to our processing server (right), where a fully automatic reconstruction pipeline generates an avatar in about 20 minutes. The user can then download their avatar into any VR application (bottom left).

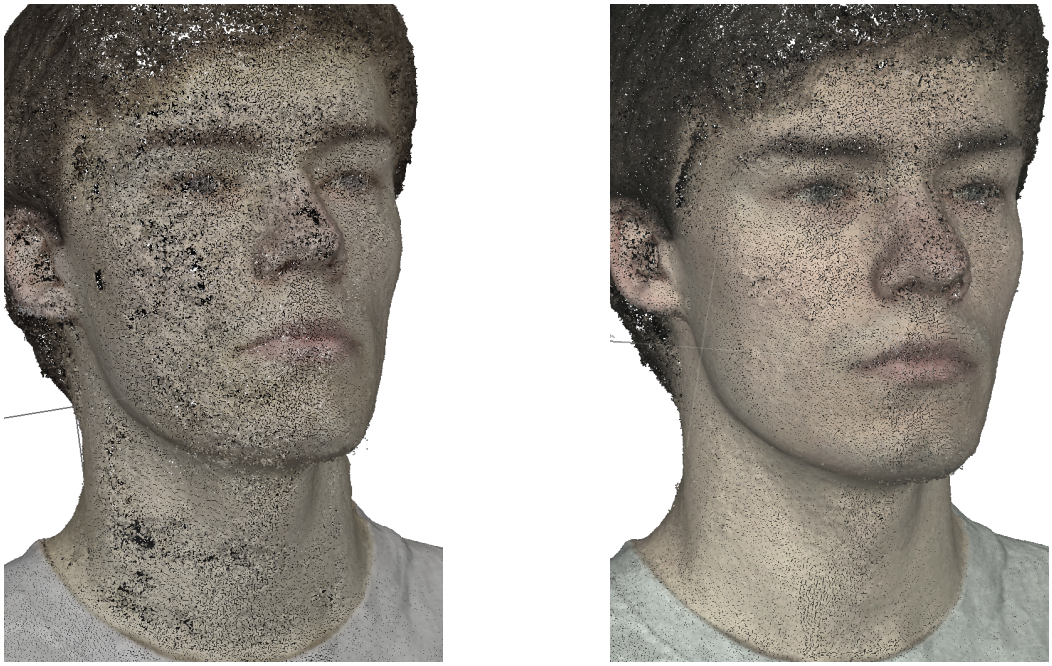


Figure 5.3: Point clouds reconstructed via photogrammetry from video frames suffer from compression artifacts (left). The higher quality of individually captured images yields more accurate point clouds (right).



Figure 5.4: The guided scanning procedure of the smartphone application. Top row, from left to right: Initial overlay before starting the body scan; overlay during the body scan; end of body scan. Bottom row, from left to right: Initial overlay before starting the head scan; overlay during the head scan; end of head scan.

a fraction of a second only, hence allowing for rather aggressive per-frame compression. In addition, inter-frame compression exploits blockwise similarity of consecutive frames, which further degrades image quality [Wie+03]. As photogrammetry algorithms use image gradients to detect feature points, the block edges can negatively influence the quality of the resulting 3D point cloud (Figure 5.3, left). Furthermore, extracted video frames can be affected by motion blur, which Wenninger et al. [Wen+20] had to handle explicitly.

In contrast, individual photographs can be captured at considerably higher quality, since they are less prone to motion blur, avoid the inter-frame block compression artifacts, and allow the use of less aggressive compression in general. Higher-quality images, in turn, yield more accurate photogrammetry results (Figure 5.3, right), which will eventually result in more accurate avatars with fine geometric detail and higher-quality textures. Our capture process (described next) therefore records individual photographs instead of videos, at a resolution of 3024×4032 pixels.

In addition to the high-resolution RGB images, we also capture coarse depth images (576×768 pixels) using the smartphone's depth sensor and the phone's orientation (resp. gravity vector). The former helps to determine the reconstructed subject's correct size/scaling, and the latter to determine its correct orientation. Our scanning application is designed for Apple iOS devices, and this additional information is conveniently included in the metadata of Apple's HEIC image format.

Scanning UI

Our extensive experience with the approach of Wenninger et al. [Wen+20] revealed that the quality of the photogrammetry reconstruction strongly depends on the correct distance and orientation of the camera to the subject. If the camera is too close, some regions (e.g., feet, hands) might not be captured. If the camera is too far away, valuable image resolution is wasted, and the point cloud becomes less dense and noisier. Those errors typically occurred to inexperienced users – despite detailed previous instructions.

To avoid these errors, our smartphone application visually guides the user through the scanning process: On top of the camera feed, we overlay in green the silhouette of a virtual human model (of average size and shape) as seen from the intended camera position (see Figure 5.4). The user adjusts the phone/camera to roughly match the silhouette of the subject and the model. It is not necessary to precisely fit the overlay to the subject. The purpose of the overlay is only to guide the user to maintain proper distance and orientation. To also guide the user's movement around the scanned subject, the virtual camera moves around the green virtual model in the same way that the user should move around the scanned subject. In addition, the direction of the movement is indicated by a white arrow.

During the scanning process, the app captures images at a frequency of 1 Hz and a resolution of 3024×4032 pixels. The speed of the virtual camera's movement is chosen to result in 105 images for the entire scanning process, as this number experimentally turned out to be the best compromise: Fewer images degraded the point cloud quality, and more images did not improve the results but increased the computation time. Thanks to this well-controlled capture process, we require just one circle around the subject for the full-body scan and one for the head scan – thus reducing the scanning time to about two-thirds of Wenninger et al. [Wen+20]. Since a shorter scanning time reduces artifacts caused by subject movement, it also improves geometric accuracy.

A dialog informs the user when the full-body and head scans are complete (Figure 5.4, right column), after which the captured images are uploaded to the reconstruction server. All further user instructions or hints during the scanning procedure are displayed in Figure 5.4. The entire scanning process can also be seen in the accompanying video of the paper publication[1]. To further minimize scanning errors, the app displays a step-by-step tutorial before the scanning process, covering subject preparation (hairstyle, accessories, shoes, and clothes), scan pose requirements (A pose), and scan process explanations (body/head scan procedure).

5.1.2 Server-Based Reconstruction Pipeline

The avatar reconstruction pipeline, whose individual tasks are described in this section, includes several computationally expensive tasks. To speed up

1 https://doi.org/10.3389/frvir.2025.1583474

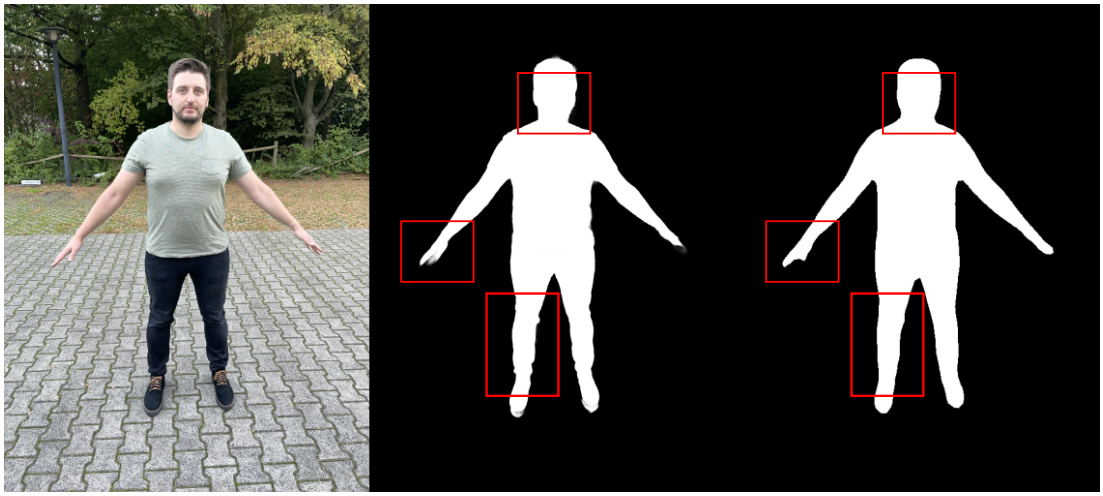


Figure 5.5: When segmenting foreground and background of the input images, Apple's person segmentation (center) better preserves small details, such as hair and clothing creases, compared to DeepLabV3 (right).

the avatar generation process and to reduce the load on the smartphone's resources, the captured images are uploaded to a compute server, where the avatar is automatically reconstructed and can be downloaded by the user.

Image Preprocessing

Our experiments with Wenninger et al. [Wen+20] revealed that their method works well in controlled indoor environments, but in outdoor environments, it often gives noticeably worse results. This is due to non-static background, such as leaves moving in the wind or cars driving by. These background movements violate the photogrammetry assumption of a static scene, leading to incorrect extrinsic camera parameters and consequently to errors in the reconstructed 3D point cloud.

To eliminate these problems and thereby make the reconstruction process much more robust with respect to "in-the-wild" capture environments, we segment the input images into foreground and background and mask out the background before passing the images to the photogrammetry process. To this end, we compared DeepLabV3 [Che+17] and Apple's person segmentation [App23c] (on macOS 15.3.1), and decided for the latter since it produced slightly more accurate and more detailed masks in our experiments (see Figure 5.5). Moreover, as the image background is excluded from the reconstruction, the number of image features to be matched by the photogrammetry is significantly reduced (accelerating this process by 30%), the resulting point clouds contain considerably fewer points (accelerating later template fitting), as well as significantly less noise and outliers (improving accuracy and robustness of the template fitting). Overall, this image preprocessing leads to faster computations and much cleaner point clouds – in particular in uncontrolled outdoor environments (see Figure 5.6).



Figure 5.6: Point clouds reconstructed from unprocessed input images captured in outdoor environments suffer from significant noise, outliers, and misalignments in the face region (A, C). By processing the images through person segmentation and background removal, these artifacts are largely eliminated (B, D).

Photogrammetry

The captured and segmented images are passed to the photogrammetry stage, which reconstructs a dense 3D point cloud (see Figure 5.2). Wenninger et al. [Wen+20] employ Metashape [Agi25] for this task, a widely used commercial photogrammetry software. Unfortunately, its license restrictions explicitly prohibit its use in server-based reconstruction scenarios (which we aim for). In order to make our system publicly available for research purposes, we compared several non-commercial alternatives for photogrammetric reconstruction, including MeshRoom [Ali25], COLMAP [SF16], and Apple's RealityKit [App23b] (on macOS 15.3.1). From these frameworks, Apple's RealityKit consistently produced the highest-quality results, which are very similar in terms of geometric fidelity, texture quality, and processing time to those of Agisoft Metashape (see Figure 5.7).

Apple's high-level photogrammetry API allows to specify different configuration options, where we found the settings *raw* detail, *high* feature sensitivity, and *unordered* samples to yield the best results. In addition to the high-resolution RGB images, we also provide segmentation masks, coarse depth images, gravity vectors, and EXIF data, which significantly improve the photogrammetry results compared to processing only the RGB images. The depth data and gravity vectors help to determine the correct scale and orientation of the reconstructed object, which make the upcoming steps more reliable.

Landmark Detection

The previous photogrammetry stage produces a (static) high-resolution textured triangle mesh (see Figure 5.7). This mesh can suffer from artifacts in insufficiently scanned regions and is lacking animation controls (body skeleton, facial blendshapes). The well-established approach is to fit a high-quality template mesh with all required animation controls to the photogrammetry



Figure 5.7: The (meshed) photogrammetry reconstructions of Agisoft Metashape (left) and Apple's RealityKit (right) are very similar in geometric fidelity, texture quality, and computational performance.

output (point cloud or mesh). This results in a reconstructed avatar mesh that inherits its triangulation, UV texture layout, and animation controls from the template model, while closely resembling the geometric shape and the texture/material of the photogrammetry scan. This template fitting process (described in the next subsection) has to be initialized and guided by a set of landmarks on both the template model (where they are pre-selected once) and the photogrammetry output (where they have to be manually selected or automatically detected).

Wenninger et al. [Wen+20] detect body and face landmarks in the 2D input images using OpenPose [Cao+19] and Dlib [Dli22], respectively, where they select the best-suited (e.g., most frontal) input images based on heuristics. The detected 2D landmarks are re-projected onto the photogrammetry point cloud. While working well in most cases, their approach can fail if wrong images are selected for landmark detection or if a 2D landmark in sparsely sampled regions back-projects to the wrong surface part. We avoid both issues by rendering the photogrammetry mesh (Figure 5.7) from several camera positions and performing landmark detection on the resulting synthetic images. Our controlled capturing process enables the straightforward selection of suitable camera views, and the back-projection onto the rendered 3D mesh is well-defined for any detected 2D landmark.

We detect 4 hand landmarks (two knuckles on the left and right hands) and 37 face landmarks (eye contours, tip of the nose, and mouth features), which are passed on to the template-fitting stage. To ensure accurate and reliable landmark detection, we compared on a wide range of examples the face/hand landmark detection of OpenPose [Cao+19], Dlib [Dli22], Apple's Vision Framework [App23a; App25], and Google's MediaPipe [Lug+19]. Since MediaPipe produced the most reliable results in our experiments, we chose this landmark detector for our reconstruction pipeline.

Template Fitting

The previous two stages result in two high-resolution textured photogrammetry meshes from the body and head scans of the subject standing in A-pose with neutral facial expression, as well as a set of 37 face and 4 body landmarks. We perform curvature-adaptive point sampling on the two photogrammetry meshes to convert them into two point clouds for the body and the head, respectively. To reconstruct the avatar, we fit a fully rigged statistical template mesh to the photogrammetry data, guided by the landmarks.

Our template mesh was designed by a skilled artist (to be free of license restrictions) and has a slightly higher resolution (23 752 vertices) than the template from the Autodesk Character Generator in Wenninger et al. [Wen+20]. Its animation rig consists of a full-body skeleton with 59 joints, as well as 52 facial blendshapes that are compatible with ARKit [App22]. The template was fit to 1700 scans of the CAESAR database [RDP99] to derive a 30-dimensional PCA subspace of human body shapes (see Section 3.1.2 for details).

In a first step, the template is coarsely fitted to the point clouds by iteratively optimizing *alignment* (position, orientation, scale), overall *body shape* (PCA weights), and *body pose* (skeleton joint angles). In the second step, the initial template fit is refined by optimizing all individual *vertex positions*. Both optimization phases minimize the sum of squared distances of photogrammetry points to their closest points on the template mesh in a non-rigid ICP manner, guided by the landmark points. Both steps are regularized to prevent overfitting: the first step by Tikhonov regularization on the PCA weights, the second step by a discrete bending energy (see Achenbach et al. [Ach+17] and Wenninger et al. [Wen+20] for details). After the fine-scale deformation, the template's blendshapes are transferred to the fitted mesh using deformation transfer [SP04]. More details on the fitting procedure can be found in Section 3.2.

Our method differs from Wenninger et al. [Wen+20] in two aspects: Wenninger et al. first fit the template to the body point cloud and then refine the result by fitting it to the head point cloud. Since in their approach the absolute scaling of these point clouds is unknown, the proportions of body to head can be slightly wrong. In contrast, our coarse depth images determine the absolute scale. We also pre-align the body and head point clouds using landmark-guided ICP and then fit the template to both point clouds *simultaneously*. In this process, the closest point correspondences to the head/body regions of the template mesh are computed from the head/body point clouds only, respectively. This approach effectively avoids the wrong body-head proportions (see Figure 5.10). In addition, since our advanced scanning process yields more accurate point clouds, we require less regularization in the fine-scale fitting step, resulting in more geometric details.

Texture Transfer

After reconstructing the geometric shape of the avatar in the previous step, the final step reconstructs the texture image. The two photogrammetry meshes



Figure 5.8: Avatars reconstructed with our approach, all being scanned in uncontrolled outdoor settings.

already feature high-quality textures generated from the input images, but with a rather poor UV texture layout. To have a uniform texture layout for all avatars, we transfer $\texttt{texture}_P$ of the photogrammetry mesh to $\texttt{texture}_A$ of the avatar mesh (having the high-quality texture layout of the template). For each texel $\boldsymbol{u}_A$ in the avatar's UV layout we determine the corresponding 3D point $\boldsymbol{x}_A$ on the avatar mesh (based on texture coordinates), find its closest point $\boldsymbol{x}_P$ on the photogrammetry mesh, and copy its color by its texture coordinate $\boldsymbol{u}_P$:

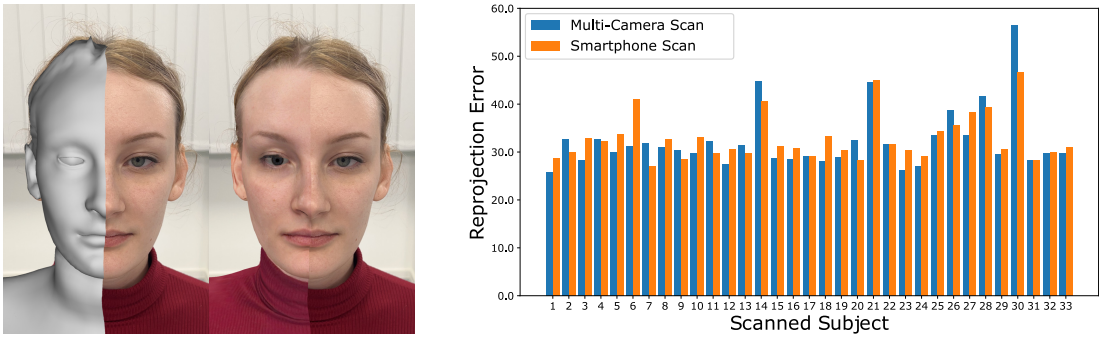
$$\texttt{texture}_A\left[\boldsymbol{u}_A\right] \leftarrow \texttt{texture}_P\left[\boldsymbol{u}_P\right]$$

Note that we actually fill two textures, from the body and head scan, respectively. These two texture images are then combined into one using Poisson Image Editing [PGB03]. This final step of the reconstruction pipeline results in a textured avatar mesh (see Figure 5.8 for some examples).

5.2 Evaluation

Reconstructing an avatar with our approach starts by capturing a person using our iPhone application (iPhone 12 Pro and iPhone 13 Pro Max in our experiments). The app visually guides the user through the scanning procedure and takes 105 images (45 full-body and 60 head images), which takes about two minutes and is shown in the accompanying video[2]. The captured image data (about 320 MB) is then uploaded to our server, which takes less than one minute over WiFi. The reconstruction is performed in a fully automatic manner on the server (Mac Studio, M1 Ultra 20 core CPU 64 core integrated GPU, and 64 GB RAM) and takes about 19 minutes (1 min segmentation, 7 min photogrammetry, 5 min offscreen rendering, 2 min landmark detection, 4.5 min template fitting and texture generation). The entire process, therefore, takes approximately 22 minutes only, after which the avatar can then be downloaded in file formats compatible with VR and game engines.

2 https://doi.org/10.3389/frvir.2025.1583474



(A) (B)

Figure 5.9: (A): We evaluate the accuracy of our approach by reprojecting our avatars (left half of images) into the captured images (right half of images). (B): Despite the significant difference in hardware complexity, the reprojection errors of our smartphone scans are only slightly worse than those of the multi-camera rig.

In the following, we compare our results to those of a complex multi-camera rig [Ach+17], to the smartphone-based method of Wenninger et al. [Wen+20], and to three recent avatar reconstruction techniques based on NeRFs or 3D Gaussian Splatting [Lei+24; Mül+22; Sha+24].

5.2.1 Quantitative Comparisons

Following Wenninger et al. [Wen+20], we evaluate the accuracy of our avatar reconstruction by reporting reprojection errors. To this end, we render the resulting textured avatar onto the images captured during the scan process using the camera calibration data from the photogrammetry process (see Figure 5.9), and then compute the root-mean-square errors over all rendered pixels in CIELab color space, averaged over all images. This metric allows us to measure errors resulting from inaccuracies in both color and geometry.

We perform this evaluation on the 33 subjects that were scanned during the user study described in Section 5.3. These participants were scanned by (i) another 33 non-expert first-time users of our smartphone application, as well as (ii) an expert using the multi-camera rig at the Embodiment Lab of the University of Würzburg (106 Canon EOS 1300D DSLR cameras, based on Achenbach et al. [Ach+17], see Section 3.3 for details on the full reconstruction pipeline). Generating the avatars using the expert-operated multi-camera rig took about 15 minutes. Our pipeline, on the other hand, took around 22 minutes. Despite the tremendous difference in expertise of the scanning person and in cost and complexity of the scanner setup, the results obtained with the camera rig are only slightly better than our smartphone scans (see Figure 5.9). Averaging over all 33 scans and comparing our RMSE (M = 32.83, SD = 4.88) with that of the multi-camera rig (M = 32.29, SD = 6.36) reveals that the error increases by less than 2%, while the financial cost decreases by more than 98%.

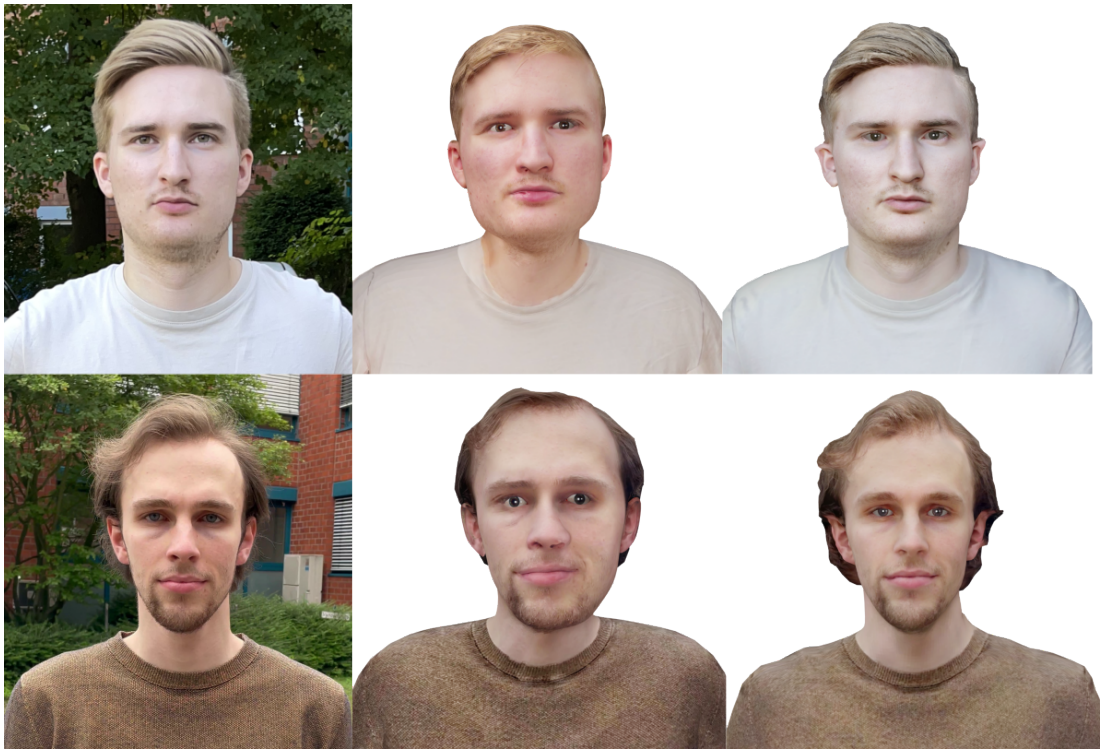


Figure 5.10: Two subjects captured in an outdoor environment (left), with avatars reconstructed using the approach of Wenninger et al. [Wen+20] (center) and our approach (right). Our avatars are considerably more accurate in terms of geometry and texture, while those of Wenninger et al. [Wen+20] suffer from photogrammetry misalignments, (required) strong regularization, and wrong body-to-head proportions.

5.2.2 Qualitative Comparisons

Besides the easy-to-use visually guided scanning procedure, our method improves the approach of Wenninger et al. [Wen+20] by several technical contributions, as described in Section 5.1. To evaluate the effect of these contributions, we compare with their method in Figure 5.10. The two subjects were captured in an outdoor environment by recording videos (for their method) and images (for our method) on the same iPhone 12 Pro. The avatar generation took 15 minutes (their method) and 22 minutes (our method). Our method produces noticeably more accurate results, with more geometric detail and higher-quality textures. This results from more accurate photogrammetry point clouds (due to recording higher-resolution images instead of videos and due to background removal) and from better template fitting (due to simultaneously fitting to body and head point clouds, and requiring less regularization). The results of Wenninger et al. [Wen+20] suffer from considerably fewer geometric details and wrong body-to-head proportions. These differences are even more prevalent for the subject represented in the bottom row of Figure 5.10, where the head is unnaturally deformed due to camera misalignments in the photogrammetry step (see Figure 5.6C).

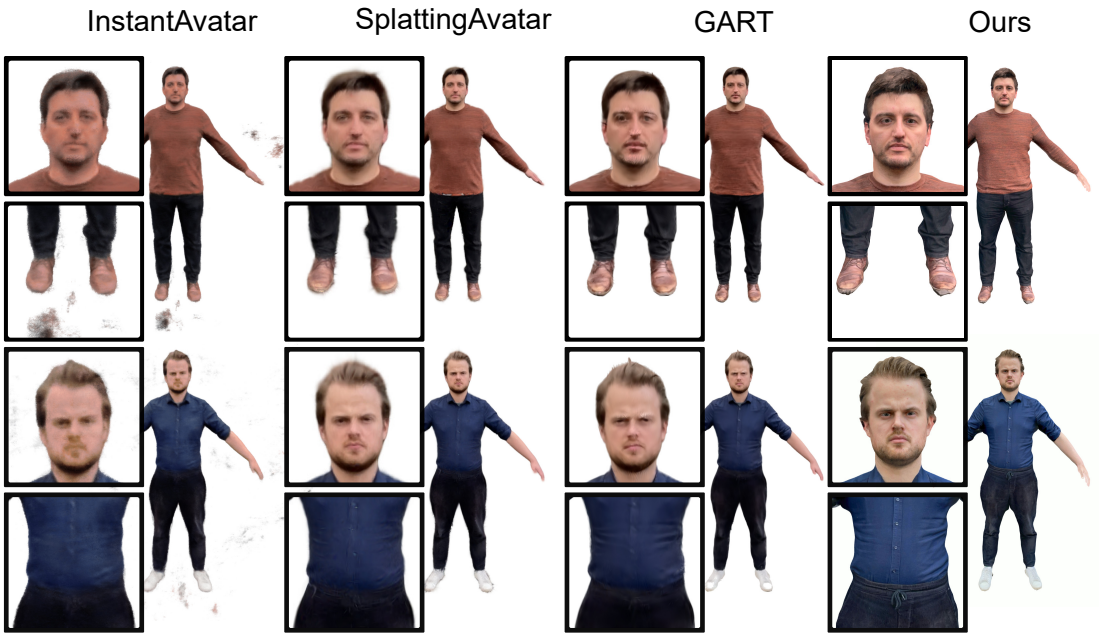


Figure 5.11: Reconstructed avatars rendered in A-pose and view from the training data. From left to right: InstantAvatar [Jia+23], SplattingAvatar [Sha+24], GART [Lei+24], and ours. InstantAvatar produces visual clutter. SplattingAvatar and GART are visually appealing, but are blurrier than our reconstructions.

In order to evaluate whether NeRF- or 3DGS-based avatars are a viable alternative to mesh-based avatars for VR applications, we experimented with three recent approaches: the NeRF-based InstantAvatar [Jia+23] and the 3DGS-based methods SplattingAvatar [Sha+24] and GART [Lei+24] – since those methods reported fast training times and high rendering performance. To achieve optimal results, we followed the recommendations of these projects and used their training scripts. Since all three methods can reconstruct avatars from the PeopleSnapshot format [All+18b], we recorded equivalent videos (1080×1080 pixels, subject rotating in A-pose) using the same iPhone as for our scans. These videos are then converted to the PeopleSnapshot format using VideoAvatars [All+18b], and the per-frame SMPL poses are refined using Anim-NeRF [Che+21]. On this prepared data, we ran InstantAvatar using their provided scripts. The data resulting from InstantAvatar then acts as input for running SplattingAvatar and GART. We used batch size 4 for InstantAvatar and 2 min training of GART, as these produced the best results. The results from different configurations are shown in Appendix A.

The (required) video pre-processing (landmark detection, segmentation, VideoAvatars pipeline, and Anim-NeRF) took about 17 hours on a compute server with three NVIDIA RTX 6000 having 48 GB GPU memory each. Training of InstantAvatar, SplattingAvatar, and GART took another 2–5 minutes, 25 minutes, and 2 minutes, respectively, on a different server with an NVIDIA RTX A5000 and 24 GB GPU memory. With more than 17 hours, the overall reconstruction time of these methods is 45 times longer than ours.

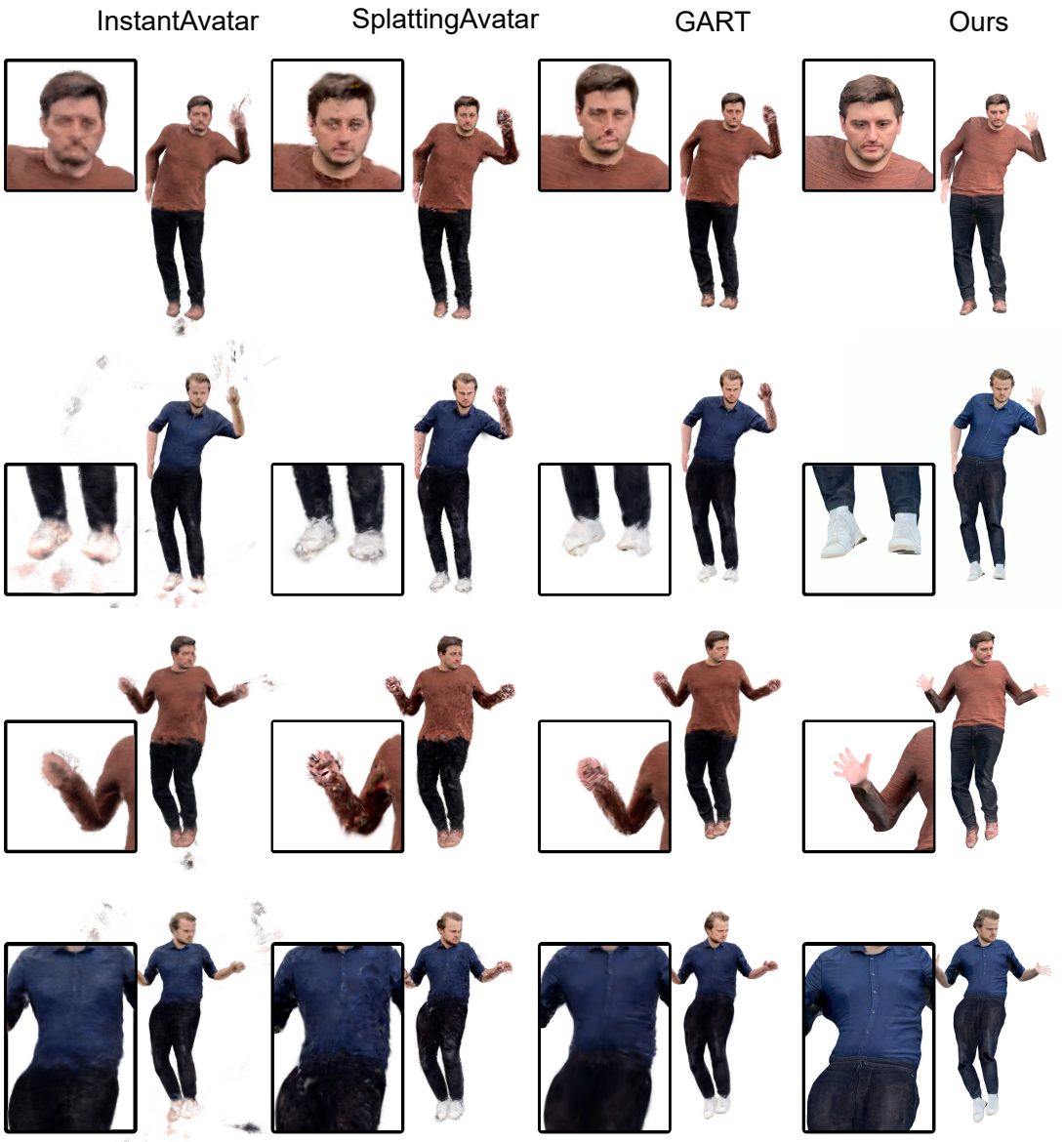


Figure 5.12: Avatar reconstructions animated in novel poses. From left to right: InstantAvatar [Jia+23], SplattingAvatar [Sha+24], GART [Lei+24], and our result. Although InstantAvatar, SplattingAvatar, and GART produced visually appealing results in the training poses, the reconstructions get noisier and blurrier in novel poses. Details, e.g., hands and faces, are hardly recognizable anymore and the renderings get even blurrier. In contrast, our results are as sharp and detailed as in the training poses.

Figure 5.11 shows the resulting avatars in a training pose and from a training camera view. While InstantAvatar produces visual clutter, both SplattingAvatar and GART are visually appealing – although more blurry than our reconstruction. However, as shown in Figure 5.12 and the accompanying video[3], when it comes to novel poses and/or novel viewpoints, the quality of NeRF- and 3DGS-based avatars quickly degrades to a level not acceptable for social VR applications. In particular, for multi-avatar VR applications, where users/avatars can take on arbitrary poses and be viewed from arbitrary camera positions, the artifacts shown in Figure 5.12 are more the rule rather than the exception. Avoiding these generalization problems would require much more training data, i.e., capturing the subject in significantly more poses and from significantly more camera views, which in turn would make the scanning significantly more complex and the reconstruction significantly more expensive – therefore requiring a complex multi-camera video recording setup. Our mesh-based avatars, in contrast, are sufficiently regularized by the statistical human body template and its animation controls to enable generalization to novel views and novel poses, even when captured from 105 smartphone images only.

In addition to long reconstruction times and suboptimal visual quality, the rendering performance of NeRF- and 3DGS-based avatars is not (yet) sufficient for VR applications, where currently HMDs require around 90 fps stereoscopic rendering (i.e., 180 fps monoscopic rendering) at about 2k resolution per eye. We therefore evaluated the rendering performance at a resolution of 2160×2160 pixels on a VR workstation (AMD Ryzen 9 7950X CPU, RTX 4090 24GB GPU, 64GB RAM). For this monoscopic rendering, InstantAvatar achieved 1.22 fps (819 ms/frame), SplattingAvatar 171 fps (5.9 ms/frame), and GART 245 fps (4.1 ms/frame). While InstantAvatar was far from the required frame rate for VR applications, the rendering performances of SplattingAvatar and GART were just on the edge of VR applicability. However, contrary to these rather low performance results, our mesh-based representation can be rendered at 4615 fps – therefore, comfortably allowing even multiple avatars in the same virtual environment at the same time.

5.3 User Study

We conducted a comprehensive user study following a multi-method approach. The study evaluated the quality of our generated avatars (called below *smartphone avatars*) by measuring their impact on a prominent selection of well-known and often studied avatar effects. In addition, it also evaluated the general usability and user satisfaction of the smartphone front-end. The purpose was to assess the quality of the avatars subjectively and to improve the user experience of scanning and being scanned with the smartphone app.

3 https://doi.org/10.3389/frvir.2025.1583474

To this end, we arranged the participants into dyads, where one participant had to perform a smartphone app scan of another participant. While the scanning participant evaluated the app's usability afterward (in the following called *smartphone app evaluation*), the scanned participant assessed the perception of the scanning processes and the generated avatar (in the following called *avatar evaluation*).

For the smartphone app evaluation, participants performing the smartphone scan were asked to assess the app's usability using standardized questionnaires, allowing for comparison with validated benchmarks. Additionally, we conducted semi-structured interviews to gather more feedback on the user experience of both scanning and being scanned with the smartphone app. The results are used as part of a user-centered design process to improve the app.

For the avatar evaluation, we adopted and extended the approach from Bartl et al. [Bar+21] and utilized a counterbalanced within-subject design comparing our generated smartphone avatars to (a) photorealistically reconstructed personalized avatars from a state-of-the-art expert system (in the following called *camera rig avatar*; based on the multi-camera rig pipeline of Achenbach et al. [Ach+17], see Section 3.3 for details) and (b) gender- and ethnicity-matched generic avatars. We chose condition (a) to compare the quality of our *smartphone avatars* to the quality of personalized avatars frequently used in recent avatar research but reconstructed by a rather costly and complex technical setup. This allowed us to assess the impact of the proposed method's reconstruction quality on typical well-known and often studied avatar effects in relation to the much lower technical requirements of our method. We chose condition (b) to assess the quality of both reconstruction methods in terms of self-similarity and self-attribution of the resulting avatars and to measure the overall effect of personalization. An additional in-VR comparison to any of the NeRF- or 3DGS-based avatars (see Section 5.2.2), unfortunately, was still unreasonable. The given state of the art was inadequate for a VR evaluation due to the low rendering performance, but most importantly due to the significant artifacts (see Figure 5.12 and video), specifically given the arbitrary poses and camera positions typical for the VR exposure.

During individual *one-by-one exposures*, the scanned participants embodied each of the three avatar types successively while engaging in various body-centered movement tasks in front of a virtual mirror within a VR environment. Then, they evaluated the avatars regarding (a) sense of embodiment and self-identification, (b) plausibility, and (c) uncanny valley effects. In a final *side-by-side exposure*, participants simultaneously embodied each type of avatar while observing them exclusively from an allocentric perspective in three different virtual mirrors (one for each type) and answering different preference questions. Afterward, we asked the participants why they preferred their chosen avatars.

5.3.1 Apparatus

This section presents the apparatus of our study. We detail the integration of the three different avatar types utilized in the study, the virtual reality system, the realization of avatar embodiment, and the virtual environment and tasks participants have to perform.

Avatars

Each participant attending the smartphone app evaluation (in the following called *scanning participant*) used our smartphone app to create a personalized avatar for the corresponding participant attending the avatar evaluation (in the following called *scanned participant*). We maintained uniform lighting conditions to enhance the avatars' comparability with the camera rig avatars. The scanning participant received instructions from the smartphone app tutorial and was directed to guide the scanned participant accordingly. No further post-processing was performed on the smartphone avatars.

We created a personalized camera rig avatar for each participant in the avatar evaluation using the expert body scanner of the Embodiment Lab at the University of Würzburg (see Section 5.2.1). No further post-processing was performed on the camera rig avatars.

Since avatars that do not match the user's gender and ethnicity have been shown to impact SoE particularly negatively [DIM24] and would, consequently, lead to an unequal comparison with personalized avatars that are matched in gender and ethnicity, we decided to match both between user and generic avatars. To this end, we chose the Validated Avatar Library for Inclusion and Diversity (VALID, [Do+23]). Through a LimeSurvey questionnaire, each participant in the avatar evaluation was asked to select the VALID avatar that most closely matched their own gender and ethnicity. As the participants typically attend studies dressed casually, they could choose between 42 casually dressed VALID avatars, consisting of three male and three female avatars, each of seven different ethnicities.

Virtual Reality System

The VR system was realized using Unity 2020.3.25f1 [Uni20]. We utilized a Valve Index head-mounted display (HMD) featuring a resolution of 1440×1600 pixels per eye and a total field of view of $114.1 \times 109.4°$ [Wol+22a]. Its refresh rate was set to 90 Hz. Participants' hand and finger movements were tracked through two Index controllers and their built-in proximity sensors. Four SteamVR base stations covered the 3×3 m tracking area. All mentioned components were integrated into the VR system using SteamVR version 2.3 [Val24a] and its corresponding Unity plug-in version 2.7.3 [Val24b]. We routed the HMD's cable to a VR-capable workstation (Intel Core i7-7700K CPU, NVIDIA GeForce GTX 1080, 16 GB RAM) running the VR system on Windows 10. For body tracking, we utilized the markerless body tracking system from Captury. Body poses were captured using eight FLIR Blackfly S BFS-PGE-16S2C RGB

cameras running at 100 Hz, which have been connected via two 4-port 1 GBit/s ethernet frame-grabber to a high-end workstation (NVIDIA GeForce RTX 3080 Ti, 32 GB RAM, AMD Ryzen 9 5900x) running Captury Live in version 259 [Cap23a] on Ubuntu 18 LTS. The body poses were continuously integrated into the VR system using Captury's corresponding Unity plug-in [Cap23b].

Avatar Embodiment

We realized avatar embodiment by retargeting the participant's tracked body pose to the used avatar in real-time, following the joint approaches described in previous work [Döl+22; Wol+22b]. During a short calibration process, in which the participant had to stand rigidly and upright, the embodied avatar was calibrated to continuously follow the position of the HMD and scaled to match the participant's eye height. To avoid sliding feet and inaccuracies in hand and feet positions caused by variations in skeletal structure, segment lengths, or insufficient hand tracking, we utilized an IK-supported end-effector optimization using FinalIK version 2.1. Due to a higher accuracy and sampling rate, hand positions and finger poses were taken from the Index controllers, while elbow, knee, and foot positions were taken from Captury.

Virtual Environment and Tasks

Our virtual environment was based on different Unity assets, which we adapted to create a realistically rendered setting. Figure 5.13 depicts the virtual environment, accommodating up to three virtual mirrors. Following the guidelines for self-observation mirror placement by Wolf et al. [Wol+22a], each virtual mirror was placed at a distance of 1.5 m from the participant during the study.

During each **one-by-one exposure**, participants embodied one of the three avatars in the virtual environment, where only the middle virtual mirror was shown. They could either observe their embodied avatar directly from an egocentric perspective or look into the virtual mirror to receive an allocentric perspective. Participants were asked to perform various body movement tasks in front of the virtual mirror to promote visuomotor coupling and induce SoE [Gon+10; Sla+10]. The body movement tasks adhered to a structured protocol adapted from Roth et al. [RL20] and can be found in Appendix A.

During the **side-by-side exposure**, participants embodied all three avatars simultaneously in the virtual environment, where all three virtual mirrors were shown. While they received no egocentric perspective on the avatars, they could observe each avatar through a virtual mirror. The mirrors were labeled with small numbers, and participants responded to four different preference questions by identifying the mirror number displaying their preferred avatar. The assignment of avatars to mirrors changed randomly after each question. The preference questions can be found in Appendix A. Figure 5.13 depicts the side-by-side exposure.



Figure 5.13: The three mirrors showing the expert (left), smartphone (middle), and generic (right) avatar of a female participant during the side-by-side exposure.

5.3.2 Measures

In the following, we describe the quantitative and qualitative measures used for the evaluation of the generated avatars and smartphone application.

Quantitative Measures

We assessed all quantitative measures using previously published questionnaires. When available, we used validated translated German versions of the utilized questionnaires. Otherwise, we used back-and-forth translations to translate the items into German. Participants answered all questionnaires on a MacBook Pro using LimeSurvey [Lim24].

We captured the **usability** of the smartphone app using the System Usability Scale (SUS, [Bro96]). It provides a fast and simple way to assess a system's usability using ten questionnaire items, each answered on a 5-point Likert scale. The calculated overall score ranges between 0 and 100 (*100 = highest usability*) and can be compared with benchmarks provided by previous work [BKM09; KS15; SL16].

For assessing **Sense of Embodiment and Self-Identification** (SoE) towards the avatars, we captured virtual body ownership (VBO) and agency (AG) utilizing the corresponding items of the Virtual Embodiment Questionnaire (VEQ, [RL20]) and self-location (SL) using the additional items introduced by Fiedler et al. (VEQ+, [Fie+23]). We used the items capturing self-similarity (SS) and self-attribution (SA) from the VEQ+ to assess self-identification towards

the avatars. Each factor measured comprises four items rated on a 7-point Likert scale (*7 = highest VBO, AG, SL, SS, and SA*).

We captured the avatars' **plausibility** utilizing the Virtual Human Plausibility Questionnaire (VHPQ, [Mal+22; Mal+24]). It consists of seven items that assess the avatars' appearance and behavior plausibility (ABP) and four items for matching the virtual environment (MVE). Each item is rated on a 7-point Likert scale (*7 = highest ABP and MVE*).

We captured tendencies of the avatars' appearance towards the **uncanny valley** using the revised version of the Uncanny Valley Index (UVI, [HM17]). It comprises four items each to assess the avatars' humanness (HU) and attractiveness (AT), and eight items to capture the avatars' eeriness (EE). While the items are answered on a range between -3 and 3, we report them on a range between 1 and 7 (*7 = highest HU, AT, EE*).

As a control measure, we captured participants' physical symptoms associated with **VR sickness** in a pre-post comparison using the Virtual Reality Sickness Questionnaire (VRSQ, [Kim+18]). It consists of nine items, each of which represents a typical symptom of VR sickness and is answered on a scale between 0 and 3 (*3 = highest symptomatology*). The total score of the VRSQ ranges between 0 and 100 (*100 = highest VR sickness*).

Qualitative Measures

We conducted semi-structured interviews to assess the user experiences related to both scanning and being scanned with the smartphone app. The interview protocols incorporated a retrospective thinking-aloud approach [BS90; SE93] to comprehensively analyze the interactions with the smartphone app while not influencing the scan experiences. We further included predefined questions to query positive and negative feelings experienced during the use of the app and while being scanned, the app's functionality and its intended purpose, the impact of the scanning participant on comfort or discomfort when being scanned, and the clarity and comprehensibility of the scanning process. Additionally, participants described aspects of the process they found efficient or challenging and reported any problematic incidents they faced. Finally, participants could suggest improvements to both the scan app functionality and the scanning process, and were asked about their scan preferences and if they would participate in a body scan again. Participants in the avatar evaluation were further asked which avatar they preferred regarding self-representation similarity, fidelity, plausibility, and suitability, along with reasons behind their choices. The complete interview protocols and exact phrasing of the preference questions can be found in Appendix A.

5.3.3 Procedures

In the following, we describe the standardized experimental procedures of our smartphone app and avatar evaluations. Figure 5.14 visualizes both procedures and highlights their intersection during the smartphone app scan.

Initially, participants in both procedures received information about the study and privacy, consented to participate, and generated two pseudonymization codes to store personal (i.e., voice recordings and avatars) and evaluation data separately. Subsequently, they proceeded with their respective evaluation procedures.

Smartphone App Evaluation

Each participant in the smartphone app evaluation (i.e., the scanning participant) first completed a tutorial on how to perform a body scan using the smartphone app. As soon as the other participant arrived for the scan in the laboratory, both participants were introduced to each other. The participant performing the scan verified that all requirements for the scan were met and instructed the scanned participant not to speak or move during the scan. To ensure that an assessable avatar was generated, the scanning participant performed two successive scans. After scanning, the scanned participant left the laboratory, and the scanning participant answered the SUS questionnaire using LimeSurvey. Following that, the participant was interviewed and completed demographics. On average, the entire smartphone app evaluation took approximately 41 min.

Avatar Evaluation

Each participant in the avatar evaluation (i.e., the scanned participant) first participated in a smartphone and expert scan conducted in a counterbalanced order. After the scans, the participant was interviewed about the scan processes, chose a generic avatar as described above, completed the demographics, and answered the pre-VRSQ. The one-by-one exposures followed in a counterbalanced order, each lasting on average 7.6 min. After each exposure, the participant answered the VEQ, VEQ+, and UVI. The following side-by-side exposure averaged 4.2 min and was accompanied by the preference questions answered verbally in VR. For each exposure, a vision test and the avatar embodiment calibration were performed following the instructions on a virtual whiteboard. In addition, the participant received audio instructions for all tasks. Finally, the participant completed the post-VRSQ. On average, the entire avatar evaluation lasted 103 min.

5.3.4 Participants

Adhering to the ethical standards of the Declaration of Helsinki, our study received approval from the ethics review board of the Institute Human-Computer-Media (MCM) at the University of Würzburg [4]. We recruited a total of 66 participants organized into 33 dyads using the local participant management system and compensated them either by course credits or cash, both depending on the duration of their participation. In none of the dyads

4 https://www.mcm.uni-wuerzburg.de/forschung/ethikkommission

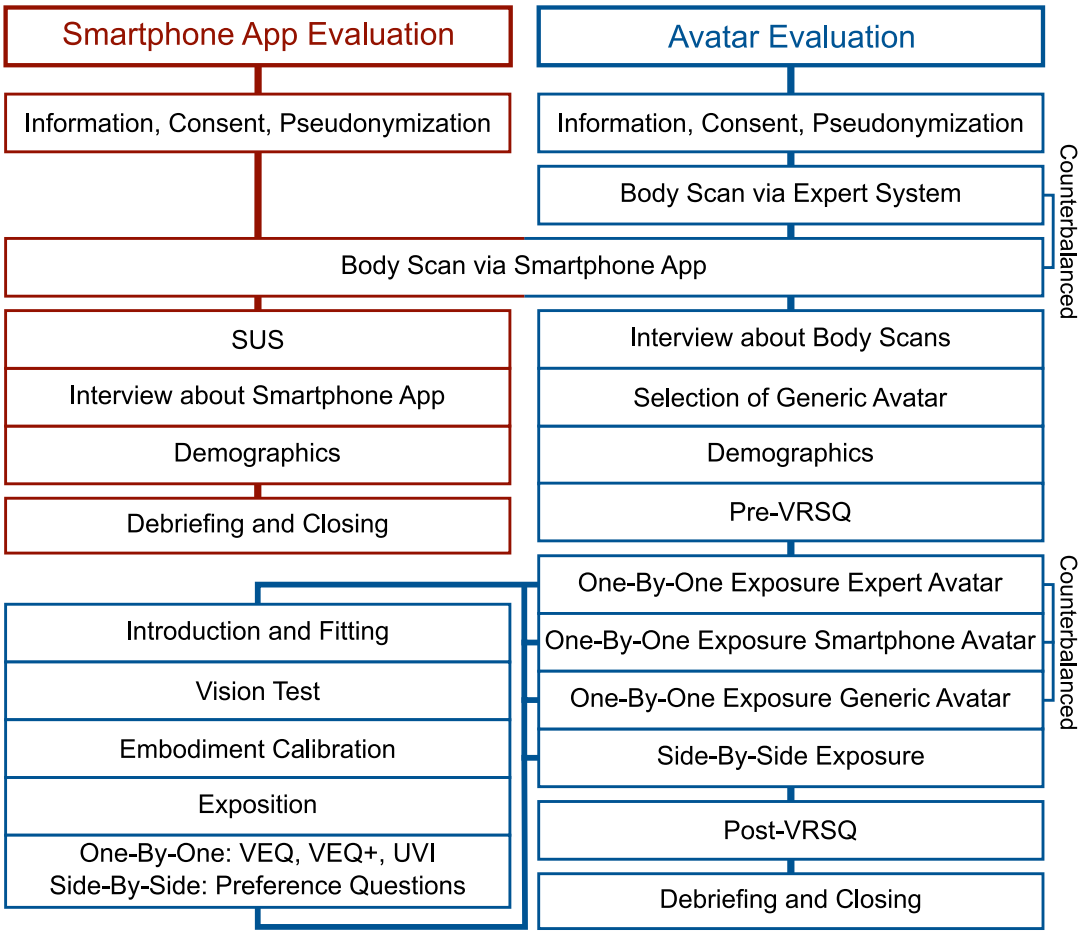


Figure 5.14: Experimental procedure of a dyad, illustrating the process of evaluating the smartphone app (left) and the avatars (right).

did participants know each other before the study. All participants had normal or corrected vision and no hearing impairment. Participants evaluating the smartphone app (19 female, 14 male) were aged between 19 and 41 ($M = 26.60, SD = 5.48$). None of them had used the smartphone app before. Participants evaluating the avatars (25 female, 8 male) were aged between 20 and 49 ($M = 27.64, SD = 6.90$). While none of them had been scanned with the smartphone app before, nine participants had previously taken part in an expert scan. Most participants in the avatar evaluation (29 White, 2 Asian, 1 MENA) chose a generic avatar that matched their ethnicity. Only one White participant chose a Hispanic avatar. Ten participants used VR for the first time, 20 up to ten times, one more than ten times, and two more than 20 times.

We excluded one dyad from our statistical analysis as one participant used the smartphone app contrary to the instructions, resulting in an unusable avatar. While all participants stated that they had more than five years of experience with the German language, we had to exclude another participant from the avatar evaluation as the experimenter felt that the participant did not understand the questions and instructions correctly, which was confirmed by

implausible answers and outliers in the data. Hence, 32 datasets remained for the smartphone app and 31 for the avatar evaluation.

5.3.5 Data Analysis

We conducted all quantitative analyses using SPSS version 29.0.2.0 [IBM22]. Before running the statistical tests, we checked whether our data met the assumptions of normality and sphericity for parametric testing. Shapiro-Wilk tests showed clear violations of the normality assumption for both dimensions of the VHPQ and minor violations for VEQ agency and VEQ+ self-location. Mauchly's test for sphericity confirmed homoscedasticity between the groups for all of our measures. Since variance analysis shows robustness to slight violations of normality for groups with $N \geq 30$ [Wil22], we decided to perform parametric tests for all measures except those from the VHPQ. All main tests have been performed against an α of .05, while post-hoc tests have been Bonferroni adjusted.

The qualitative feedback has been analyzed following the principles of thematic analysis [BC06]. We report the results mainly based on the frequency of certain feedback while mostly refraining from direct quotes.

5.3.6 Results

This section reports the results of our study. We first present the smartphone app evaluation results, followed by the avatar evaluation results.

Smartphone App Evaluation

The quantitative evaluation of the smartphone app's usability resulted in a reasonably high SUS score ($M = 78.83, SD = 12.23$). We compared the results to absolute benchmarks from existing literature. According to Sauro et al. [SL16], our smartphone app shows above-average usability. While a score between 77.2 and 78.8 leads to a usability grade of *B+*, a score between 78.9 and 80.7 relates to an *A-*. This grade matches the classifications of the adjective rating scale of Bangor et al. [BKM09], where a score above 71.4 is considered *good*, while a score above 85.5 would be *excellent*. According to the work of Kortum et al. [KS15], our smartphone app's usability can almost keep up with the usability of the ten most-used iPhone apps, which have an average SUS score of 79.3.

When analyzing interviews about the usability of the smartphone app, the majority of the 32 participants performing the smartphone app scan found it highly usable. Twenty-nine participants stated that the app's functionality and purpose were easy to understand, while 26 reported that they constantly knew how to use it. As particularly useful features, 20 participants highlighted the overlay for controlling scan distance and movement, 16 participants the initial tutorial, and five participants the arrows indicating the movement direction. Nonetheless, challenges were also noted. Twenty-three participants

	Smartphone	Camera Rig	Generic	Group Comparisons
	M (*SD*)	*M* (*SD*)	*M* (*SD*)	
Sense of Embodiment				
VEQ Ownership (VBO)	4.10 (1.50)	4.79 (1.28)	3.75 (1.44)	$F(2,60) = 11.011, p < .001, \eta_p^2 = .268$
VEQ Agency (AG)	5.57 (0.96)	5.97 (0.79)	5.79 (0.81)	$F(2,60) = 2.845, p = .066, \eta_p^2 = .087$
VEQ+ Self-Location (SL)	4.11 (1.07)	4.14 (1.03)	3.73 (1.13)	$F(2,60) = 3.502, p = .036, \eta_p^2 = .275$
VEQ+ Self-Similarity (SS)	5.69 (1.11)	5.85 (0.68)	2.69 (1.23)	$F(2,60) = 82.651, p < .001, \eta_p^2 = .734$
VEQ+ Self-Attribution (SA)	4.69 (1.21)	4.99 (0.99)	3.37 (1.16)	$F(2,60) = 31.390, p < .001, \eta_p^2 = .511$
Plausibility				
VHPQ Appearance/Behaviour (ABP)	4.84 (0.82)	5.33 (0.78)	5.27 (0.78)	$\chi^2(2) = 4.581, p = .101, W = .074$
VHPQ Match to VE (MVE)	5.15 (0.98)	5.47 (1.14)	5.80 (0.65)	$\chi^2(2) = 5.782, p = .056, W = .093$
Uncanny Valley				
UVI Humanness (HU)	3.46 (1.16)	3.84 (1.08)	3.36 (0.90)	$F(2,60) = 2.444, p = .095, \eta_p^2 = .075$
UVI Eeriness (EE)	4.05 (0.79)	3.87 (0.97)	3.15 (0.79)	$F(2,60) = 19.313, p < .001, \eta_p^2 = .392$
UVI Attractiveness (AT)	3.90 (1.20)	4.13 (0.84)	4.52 (0.69)	$F(2,60) = 3.264, p = .045, \eta_p^2 = .098$

Table 5.1: Exact descriptive values for each measure of the avatar evaluation per group and statistical results of the group comparisons.

reported difficulties maintaining an appropriate moving pace while scanning, with six participants emphasizing this problem, especially for the head scan. Similarly, seven and six participants reported issues with aligning the overlay while moving and keeping the correct distance, respectively. Six participants mentioned the need for high concentration, and 18 felt a bit uncomfortable due to the close proximity to the scanned participant. Six participants considered the relatively long duration of the scan process as unpleasant. To address the mentioned aspects, eight participants proposed a more detailed tutorial, and another four suggested an initial overlay mapping to the height of the scanned participant. To improve the scan process, five participants recommended more interaction with the scanned person, five more additional feedback on pacing their movement during the scan, and another five stressed the need to shorten the scan duration.

In addition to feedback on performing the scan, we obtained reports from the 32 scanned participants on their scanning experience. Overall, the process was clear and manageable, with 30 participants completely understanding the required actions. All participants confirmed their willingness to participate in a smartphone app scan again. However, compared to expert scans, 21 participants noted the smartphone app scan was slower, and 22 found it less comfortable. Prolonged posing discomfort was mentioned by twelve participants, while wardrobe and hairstyle constraints were issues for another four. Fourteen participants anticipated a difference between an expert and a beginner performing the smartphone scan, with four believing the expert would be faster. When asked about suggestions for improvement, four participants indicated that they would accelerate the process to reduce the discomfort of holding the scan pose. Regarding the head scan, four participants suggested a fixation to aid focus, and three to increase the distance between the camera and the head.

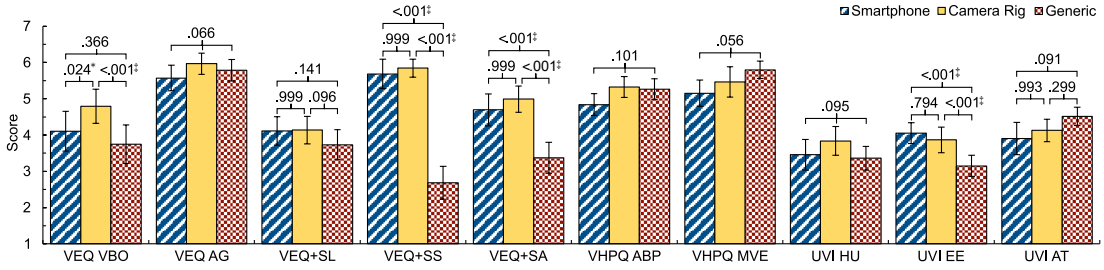


Figure 5.15: Bar charts for each measure and each group of the avatar evaluation, including statistical test results of the group comparisons and post hoc tests where applicable. Error bars represent 95 % confidence intervals. Statistical significance indicators: $^{*}\ p < .05$; $^{\dagger}\ p < .01$; $^{\ddagger}\ p < .001$.

Avatar Evaluation

To perform group comparisons on our avatar evaluation data, we calculated either a repeated-measures ANOVA for measures that met the requirements for parametric analysis or Friedman tests as a non-parametric alternative. The descriptive data and the results of the group comparisons can be found in Table 5.1. For all tests revealing significant differences between groups, we calculated Bonferroni-corrected pairwise post-hoc comparisons that are reported in Figure 5.15.

During the side-by-side exposure, we asked participants about their preferences regarding self-representation similarity, fidelity, plausibility, and suitability, along with reasons behind their choices. Out of the 31 participants included in the analysis, 16 perceived the smartphone avatars to be more similar to themselves, while 13 preferred the camera rig avatars. Regarding self-representation fidelity, 11 participants preferred the smartphone avatars, 19 chose the camera rig avatars, and one favored the generic one. To feel most plausibly represented in VR, 12 participants chose the smartphone avatars, 18 the camera rig avatars, and one the generic one. When asked which avatar the participants would prefer to be represented in VR, 10 chose the smartphone avatars, 17 the camera rig avatars, and four the generic ones. When asked for their reasoning, participants who favored smartphone avatars mostly mentioned a detailed facial reconstruction and realism as key factors. Those participants who preferred camera rig avatars highlighted the accuracy of body shape reconstruction, noting issues with smartphone avatars' body proportions, particularly the arms. Participants who chose generic avatars consistently did so because of overall dissatisfaction with their personal appearance rather than avatar quality.

5.4 Discussion

In this section, we discuss the results of the comparisons with the different avatar reconstruction methods and the results of our user study. Furthermore,

Method	Easy	Fast	Affordable	Realistic	Full-Body	VR-Ready
[Ach+17]	✗	✓	✗	✓	✓	✓
[Wen+20]	✗	✓	✓	✓	✓	✓
[Jia+23]	●	✗	✓	●	✓	✗
[Sha+24]	●	✗	✓	●	✓	✗
[Lei+24]	●	✗	✓	●	✓	✗
Ours	✓	✓	✓	✓	✓	✓

Table 5.2: Comparison of the different avatar reconstruction methods regarding our requirements. The symbols represent: ✓ completely, ● partially, ✗ not fulfilled.

we present enhancements to the avatar reconstruction system since the initial publication and discuss the limitations of our work.

5.4.1 Avatar Evaluation

Our quantitative and qualitative comparisons in Section 5.2 demonstrate that our smartphone application enables even non-experts to reconstruct avatars of a similar quality and accuracy as those produced with an expert-operated multi-camera rig – at a fraction of the price, complexity, and required expertise. Compared to the previous smartphone-based reconstruction [Wen+20], our proposed method is easier to use and gives higher-quality results even in more challenging in-the-wild scenarios.

Our experiments with InstantAvatar [Jia+23], SplattingAvatar [Sha+24], and GART [Lei+24] revealed that avatars based on NeRF or 3DGS generalize rather poorly to poses and camera views far from training data – a situation that cannot be avoided in multi-avatar VR applications. Although these generalization problems can be reduced with more training data, this is beyond the capabilities of a simple smartphone-based scanning solution. Also, in terms of reconstruction times and rendering performance, NeRF- and 3DGS-based avatars are not yet suitable for VR applications, such that classical mesh-based avatars appear to still be the preferred representation. Table 5.2 summarizes the fulfillment of our requirements with respect to avatar reconstruction. Although many methods show their strengths in a subset of the criteria, only our system fulfills all of them.

Compared to the work of Waltemate et al. [Wal+18], our user study confirmed that realistic avatars still offer substantial benefits over generic avatars for self-representation, even when the generic avatars are also personalized in gender and ethnicity [DIM24; Do+23]. With regard to the comparison, some further notable findings need to be addressed. The statistically significant difference in virtual body ownership between the smartphone and camera rig avatars can potentially be attributed to observed motion artifacts, which can

degrade the avatars' appearance. However, smartphone avatars still perform descriptively better than generic avatars. Regarding self-identification, the smartphone and camera rig avatars both show significant advantages over generic avatars, although the smartphone avatars were generated using a significantly cheaper method than the camera rig avatars. For the smartphone avatars, participants emphasized particularly the high similarity of the head. However, results also showed that the eeriness of realistic avatars was significantly higher than that of generic avatars. This is likely attributable to an Uncanny Valley effect originating from the emotional relatedness to self-personalized avatars, which has also been observed in other research [Döl+23; MMK12]. When considering the plausibility of the avatars, it is noticeable that the reconstruction described most realistically had the lowest match with the perceived plausibility. This discrepancy might be attributed to the incongruence between the virtual environment's realistic style and the avatars' photorealistic style [LW22].

5.4.2 Smartphone App Evaluation

We evaluated the usability of our smartphone app quantitatively using the SUS questionnaire and qualitatively using semi-structured interviews, including a retrospective thinking-aloud approach. The SUS results showed that our smartphone app is already well usable. The qualitative feedback confirmed this impression and highlighted the overlay and tutorial as particularly positive features. However, the qualitative feedback also revealed areas for improvement.

As part of the user-oriented design process, we already incorporated suggested improvements. To address comments regarding the duration of the scan and the pace, we added the option to adjust the scan speed in the settings. The unclear parts of the tutorial have been improved to better prepare users for the scan. Furthermore, we have also added warnings if the scanned person is not sufficiently centered. Other feedback could not be implemented due to technical limitations or requires further research. For example, the distance between the smartphone and the scanned person, especially during the head scan, could only be increased by the loss of detail in the reconstructed avatars. However, since the high quality of the faces is a significant advantage of our system, we decided to keep the required distance. Furthermore, the interaction between the scanning and scanned person and visual aids (e.g., fixation point) for the scanned person lie outside the influence of our smartphone application.

5.4.3 Enhancements to the Avatar System

Since the publication of our avatar reconstruction system, we have further enhanced the app's usability features. These improvements, implemented after the initial study and publication, are described here to provide a complete overview of the system. We refined the scan overlay, which guides the scanning

person during the scanning process. During the study, we set the overlay's color to green; however, in real-world scenarios, green might not be ideal for all users. If a color-blind user is using the system or if the scanning location has a background color similar to the overlay, a different color can increase the contrast and visibility of the overlay. We therefore added an option to adjust the overlay's color (green, yellow, red, or blue). With this setting, every user can adjust the color for each scan based on their individual needs, making the scanning app more accessible.

We occasionally experienced overlay update delays during the scanning phase; i.e., the overlay was not updated in time, or some frames were skipped. These delays can reduce users' trust, as the app appears unresponsive. To address this, we changed the overlay feature to render an overlay video instead of individual frames. This video starts at the beginning of a scan and does not rely on timer-based update calls to change individual frames, effectively eliminating lag.

We also adjusted the delivery process of the computed avatars. Initially, we uploaded images, computed an avatar, and sent a download link to the user. This design had three disadvantages: (1) anyone with the link could download the avatar, (2) users could revoke their usage permissions by mail only, and (3) users were unable to view the status of the avatar reconstruction. Therefore, we created an additional distribution website[5] and implemented an account system for the website and the scanning app. Now, a scan is uploaded to a specific user profile, processed, and only the associated user can download the resulting avatar through the distribution website. The website also allows us to display additional information about the uploaded scans. For each scan, we display a progress bar that shows its status in the reconstruction pipeline. Furthermore, we summarize the usage permissions, which can be specified at the beginning of a scan in the smartphone app. In contrast to the initial system, users now have three interaction possibilities. Through clickable buttons, users can *revoke permissions*, *download* the result, or *delete* the result. Revoking permissions or deleting the result is performed automatically after the user's request, giving users full control over their data.

5.4.4 Limitations

Since our method uses photogrammetry software to generate point clouds from images, the input images must contain as little movement as possible. If movement occurs in the background, the segmentation significantly improves the photogrammetry results. However, the motions of the scanned subject violate the photogrammetry assumption, i.e., that the scanned object is rigid and not moving, leading to less accurate point clouds and, therefore, geometric deformations in the final avatar. Figure 5.8 shows this problem in more detail, as the arms of the second avatar (from left) have visible differences in thickness.

5 https://avatars.cs.tu-dortmund.de

We use a mesh-based representation for our avatars. On the one hand, this enables high-performance rendering and novel pose generation. On the other hand, we represent cloth, hair, and skin by a single textured mesh, which can lead to visual artifacts. An interesting direction for future work would be to combine mesh-based avatars (potentially with multiple layers for skin and cloth) with volumetric details (such as hair) represented by Gaussian Splatting – as this would combine the strengths of both representations.

Our system uses image segmentation to preprocess the input and mask out regions that do not contain people. For that reason, people in the background are a challenging task, as they are not removed. We want to explore the capabilities of the iPhone's depth sensor to remove people in the background from the masks.

We rely on Apple frameworks for both our scanning client and reconstruction server. For the client, an Android-based app would be possible, but we chose iOS because Apple's photogrammetry yields correctly scaled results thanks to the LiDAR sensor of the pro-level iPhones. On the reconstruction server, only the photogrammetry and the segmentation frameworks are Apple-specific; all other parts of the reconstruction pipeline are cross-platform compatible. We chose Apple's RealityKit and Vision frameworks since, in our extensive tests, this platform produced the best results while not being limited by restrictive licensing.

The sample in our study consisted of white participants only. As this potentially limits the generalizability of our results, in future work, a larger population sample with greater variability in age, sex, and ethnicity should be tested.

5.5 Conclusion

In this chapter, we presented a system that allows non-expert users to scan people and automatically reconstruct realistic VR-ready full-body avatars that achieve similar perception results compared to avatars reconstructed with expensive expert-operated systems. Inspired by the approach of Wenninger et al. [Wen+20], we presented methods to resolve present obstacles that prevent the wide accessibility of realistic full-body avatars. Our custom smartphone application enables laypeople to easily and quickly capture high-quality input images, which, together with background segmentation and an improved template fitting algorithm, result in more convincing reconstructions while reducing restrictions on scanning locations. Our end-to-end solution computes VR-ready avatars that can be easily integrated into existing VR pipelines. To further empower people to create realistic full-body avatars and encourage more avatar-related studies, our system is publicly available for research purposes.

6 COMPENSATING MOTION-INDUCED ERRORS IN SMARTPHONE-BASED VR AVATAR RECONSTRUCTION

In the previous chapter, we presented our fully automatic avatar reconstruction system, enabling even non-expert users to create realistic, personalized full-body avatars using only a smartphone. The system significantly reduces costs, and its evaluation demonstrated both the suitability of the resulting avatars for self-representation in VR scenarios and their superiority over existing NeRF- and 3DGS-based methods in terms of reconstruction time, rendering performance, and novel pose animation.

However, the results also indicated problems when the scanned subject moved during the scanning procedure. Since the system uses photogrammetry, it relies on the assumption that the scanned object remains static (i.e., not moving) during the capture. While the subjects are instructed to hold still, it is improbable that no motion occurs during the relatively long scanning routine. These unintentional movements are visible in the resulting point cloud as noise. When capturing a human standing in A-pose using a smartphone, this is mainly observable in the arms, especially around the lower parts, which potentially move throughout the scan (see Figure 6.1, left). Since this problem



Figure 6.1: Point cloud from all images (left) and sub-scan capturing the left part of the subject (right). The full point cloud exhibits noticeable motion artifacts in both arms.



Figure 6.2: Comparison with Avatars for the Masses (A4M, see Chapter 5): Input image (left), result from A4M (center), result from our new approach (right). The A4M result displays a noticeably bigger arm than the input image.

was ignored in the previous chapter, this noise is incorporated into the resulting avatars (see Figure 6.2, center). Here, the arm is considerably thicker than the original (see Figure 6.2, left).

In this chapter, we therefore explore methods to relax the static object assumption and create realistic avatars even from challenging smartphone scans, in which the scanned subject made slight (unintentional) movements. The central idea is to split the input image sequence into subsequences containing only minimal motion. Thereby, we can continue to use photogrammetry to generate high-quality partial point clouds that do not contain severe motion-induced artifacts. After that, we adjust the existing fitting procedure to simultaneously fit to those partial point clouds. Furthermore, we extend our fitting optimization with silhouette constraints to compensate for the remaining motion within each subsequence, and compute a high-quality texture using inverse rendering.

Individual Contribution *The template-fitting codebase is based on the reimplementation of the previous character-fitting projects (Wenninger et al. [Wen+20] and our previous work* Avatars for the Masses *(see Chapter 5)) by Friedemann Runte. Friedemann Runte and I designed and implemented the reconstruction method in this chapter in a joint effort. Moreover, I was responsible for preparing the comparison methods for evaluation, including setting up available implementations from public repositories, resolving installation and configuration issues, and preprocessing the required data. I also generated the qualitative visual comparisons, while the quantitative evaluation metrics were computed in collaboration with Friedemann Runte.*

Corresponding Publication *This chapter is based on the following publication:*

> F. Runte, T. Menzel, U. Schwanecke, and M. Botsch. "Compensating Motion-Induced Errors in Smartphone-Based VR Avatar Reconstruction". In: *Proceedings of the 2025 31st ACM Symposium on Virtual Reality Software and Technology*. 2025. DOI: 10.1145/3756884.3765995

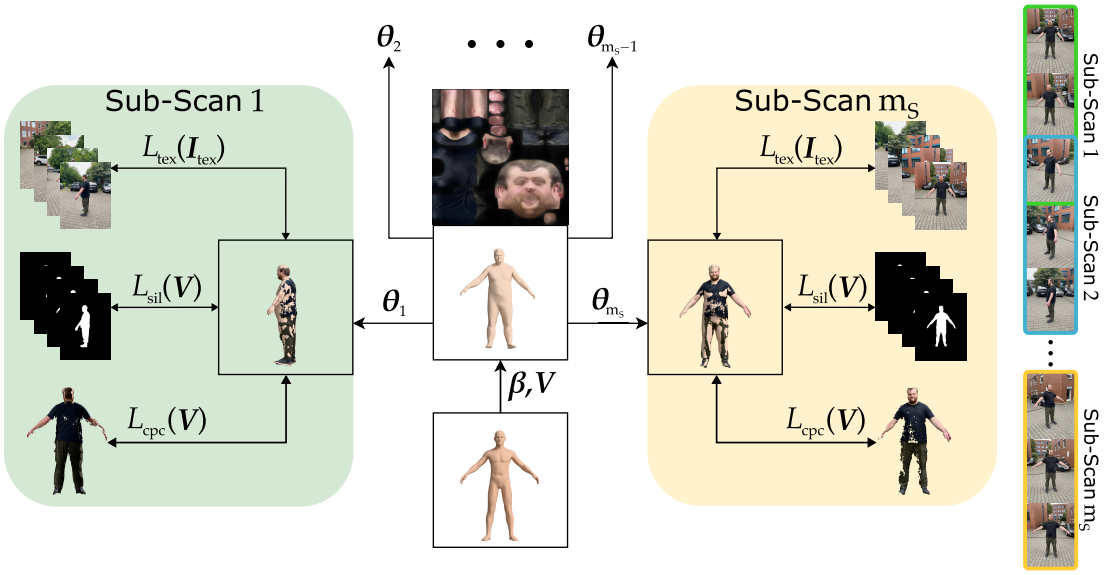


Figure 6.3: Overview of our sub-scan fitting with m_S sub-scans and sub-scan construction. We deform our template mesh using the PCA parameters $\boldsymbol{\beta}$ and the vertex positions $\boldsymbol{V}$ to get a coarsely fitted model. This is posed with joint parameters $\boldsymbol{\theta}_j$ to bring the model into the correct pose for each sub-scan. We then minimize the losses L_{cpc} (Equation (6.1)) and L_{sil} (Equation (6.2)) for each sub-scan simultaneously to compute the final geometry. After that, we minimize L_{tex} (Equation (6.5)) simultaneously for all sub-scans to generate the avatar's final texture.

6.1 Method

We combine the key ideas from different avatar reconstruction approaches into a single pipeline. We use photogrammetry to achieve high geometric detail, silhouette constraints to improve the robustness, and inverse rendering to generate a high-quality texture. To ensure reliable point clouds, despite the subject moving during the scan procedure, we introduce a new concept called *sub-scans*. We divide the captured image sequence into smaller consecutive subsequences. This reduces the subject's motion in each subsequence, resulting in point clouds with significantly fewer motion artifacts.

In the following, we first introduce our concept of sub-scans (Section 6.1.1), justify our choice of photogrammetry software (Section 6.1.2), define the loss functions used in our mesh optimization (Section 6.1.3), provide an overview of our mesh fitting pipeline (Section 6.1.4, see Figure 6.3), and finally present our inverse rendering based texture generation (Section 6.1.5).

6.1.1 Sub-Scans

We use our smartphone app (see Section 5.1.1) to capture two separate scans: one of the *body* and one of the *head*. The app captures 105 RGBD images in HEIC format (45 body images, 60 head images). All images are taken

sequentially over a period of two minutes. For the body scan, the subject stands in an A-pose, which makes it susceptible to arm movement. This is not the case for the head scan, as the head is more rigid and easier to keep in the same pose. Therefore, motion artifacts most likely arise in the body scan. As this motion occurs over the full duration of the scanning procedure, it has a reduced impact over shorter time intervals. To minimize the influence of motion, from the set of all body images $\mathcal{I}_b = \{\boldsymbol{I}_1, \ldots, \boldsymbol{I}_{45}\}$ we form m_S subsets $\mathcal{I}_b^1, \ldots, \mathcal{I}_b^{m_S}$ consisting of n_I consecutive images such that

$$\bigcup_{i=1}^{m_S} \mathcal{I}_b^i = \mathcal{I}_b.$$

We refer to these subsets as *sub-scans* (see Figure 6.3, green and yellow boxes). They form the basic concept for the loss functions, explained in Section 6.1.3. Note that we use an overlap of one image between two consecutive sub-scans. For instance, in the case of $m_S = 3$ sub-scans the 45 body images are split into sub-scans $\mathcal{I}_b^1 = \{\boldsymbol{I}_1, \ldots, \boldsymbol{I}_{15}\}$, $\mathcal{I}_b^2 = \{\boldsymbol{I}_{15}, \ldots, \boldsymbol{I}_{30}\}$, and $\mathcal{I}_b^3 = \{\boldsymbol{I}_{30}, \ldots, \boldsymbol{I}_{45}\}$.

In a first step, we generate point clouds for the body and head scans using Agisoft Metashape [Agi25]. We align both point clouds using landmarks detected with MediaPipe [Lug+19] in the input images, similar to our previous approach (see Section 5.1.2). This ensures that the head scan matches the position, orientation, and scaling of the head in the body point cloud. Furthermore, for each sub-scan, we generate corresponding point clouds $\mathcal{P}_b^1, \ldots, \mathcal{P}_b^{m_S}$. As shown in Figure 6.1, the sub-scan point cloud (right) covers a smaller area of the scanned person, but contains fewer artifacts than the full body point cloud (left), particularly in the lower arm regions where movement occurred.

In order to precisely fit our template to the sub-scans, we first align the sub-scan point clouds. We use the camera calibration of the full-body point cloud as initialization for the sub-scan photogrammetry. Despite the motion artifacts in the point cloud, the information from the camera calibration remains valid because large parts of the subject (e.g., legs, torso, and head) are static during the capture. This results in the sub-scan point clouds being aligned by construction.

6.1.2 Photogrammetry Selection

In our previous approach (Chapter 5), we aimed to build an accessible avatar reconstruction system. We opted for Apple's photogrammetry API [App23b] instead of Agisoft Metashape [Agi25] because of Metashape's restrictive licensing. However, for our new approach, we use Metashape, since Apple's photogrammetry API does not meet our requirements.

Apple's photogrammetry API always outputs a textured mesh, while Metashape offers more output possibilities (e.g., point clouds or (textured) meshes in varying qualities and densities). This flexibility is crucial for our new approach, since we aim to reconstruct partial point clouds from sub-scans.



Figure 6.4: Comparison between the results of Apple's photogrammetry API (left, [App23b]) and Agisoft Metashape (right, [Agi25]) on two exemplary sub-scans. Apple's photogrammetry API produces a triangle mesh; Agisoft Metashape offers more options, among them the option to produce point clouds. The results show that Apple's photogrammetry API attempts to fill in missing information from the sub-scan sequences to create a watertight mesh; however, this creates artifacts (e.g., area below the arm in the upper example, thickness of the arm in the lower example). In contrast, the results from Agisoft Metashape show that Metashape does not fill in the missing information. In our use case, the behavior of Metashape is the preferred one, as our intention is to reduce the amount of artifacts with our sub-scans, not increase them.

As shown in Figures 6.1 and 6.4, point clouds generated from sub-scans using Agisoft Metashape cover only the visible parts of the scanned subject. Unseen parts are left empty. In contrast, Apple's photogrammetry API attempts to produce a watertight surface, possibly introducing new artifacts in our sub-scan scenario. Figure 6.4 shows this behavior. Since we fit our template model to the photogrammetry results, any hallucinated information is directly incorporated into the final mesh. Thus, Metashape is the more favorable option

for our sub-scan scenario.

6.1.3 Loss Functions

We use a template-fitting approach to reconstruct avatars. Our template mesh, created by an artist, is defined by its $N_V \approx 24$k vertices represented by the vector of stacked vertex positions $\boldsymbol{V} \in \mathbb{R}^{3N_V}$, a set of 59 joints $\mathcal{J}$, and 52 blendshapes modeled after the ARKit blendshapes [App22]. It can be skinned using standard linear blend skinning (see Section 3.1.1). Furthermore, we can adjust the template's shape with a PCA model built from the CAESAR database [RDP99] (see Section 3.1.2 for details).

We use different loss functions to match the pose and shape of our template to the sub-scans. In the following, we define the loss functions used in our optimization and give a short description of the influence of each loss.

Closest Point Correspondences

We build sets of closest point correspondences $\mathcal{C}_i$ computed for each sub-scan $\mathcal{I}_b^i$ from point cloud $\mathcal{P}_b^i$ to our template mesh $\boldsymbol{V}$ with joint angles $\boldsymbol{\theta}_i \in \mathbb{R}^{177}$. The set of vectors of joint angles for all sub-scans is denoted as $\Theta = \{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_{m_S}\}$. Each correspondence $c = (\boldsymbol{p}_c, \boldsymbol{v}_c) \in \mathcal{C}_i$ is defined by a point in the point cloud $\boldsymbol{p}_c \in \mathcal{P}_b^i$ with $\boldsymbol{p}_c \in \mathbb{R}^3$ and a point $\boldsymbol{v}_c \in \mathbb{R}^3$ on a template triangle defined by barycentric coordinates. We define the closest point-correspondence loss function as:

$$L_{\text{cpc}}(\boldsymbol{V}, \boldsymbol{\theta}_i, \mathcal{C}_i) = \frac{1}{|\,\mathcal{C}_i\,|} \sum_{c \in \mathcal{C}_i} \| \, \boldsymbol{p}_c - \text{skin}_C(\boldsymbol{v}_c, \boldsymbol{\theta}_i) \, \|^2 , \tag{6.1}$$

where skin_C applies linear blend skinning to a point in a triangle defined by barycentric coordinates $\boldsymbol{v}_c$, and then computes the interpolated position as defined in Section 3.1.1. This loss measures the sum of squared distances between all correspondences in sub-scan i. It serves two purposes: optimizing joint angles $\boldsymbol{\theta}_i$ to match the pose of our template with the pose of the subject in sub-scan i and non-rigidly deforming the vertex positions $\boldsymbol{V}$ of our A-pose template model to match the shape in the point cloud.

Silhouette Correspondences

We introduce a silhouette constraint to improve the robustness of the optimization, as proposed in earlier approaches [All+18a; All+18b; All+19]. Similarly to the closest point correspondences, we define silhouette correspondences. The silhouette is defined by the outline of the subject in the input image. We find the outline of the subject by first segmenting the input images to separate the background from the subject and then computing the contour of the segmentation mask using OpenCV [Bra00]. To compute the segmentation masks, we compared DeeplabV3 [Che+17] and Segment Anything Model (SAM, [Kir+23]). We decided for SAM, as this model produced more accurate results.

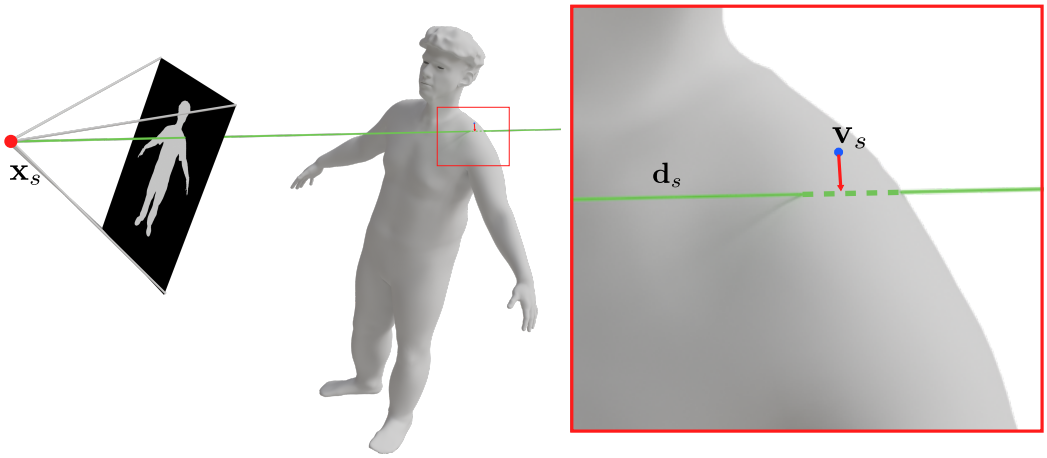


Figure 6.5: Visualization of one silhouette correspondence. The red dot is the camera center $\boldsymbol{x}_s$, the green line is the direction vector $\boldsymbol{d}_s$, and the blue dot is the template vertex $\boldsymbol{v}_s$ that will be projected onto the ray. The shoulder of the template mesh is outside of the mask.

We find correspondences by projecting the vertices of our posed model into each image, collecting all vertices outside of the segmentation mask, and finding their closest point on the outline (see Figure 6.5). The set of silhouette correspondences for sub-scan i is denoted as $\mathcal{S}_i$. $\mathcal{S}_i^j$ is the set of correspondences for image j of sub-scan i. As silhouette correspondences are calculated in image space, a single correspondence $s = (\boldsymbol{x}_s, \boldsymbol{d}_s, \boldsymbol{v}_s) \in \mathcal{S}_i^j$ is defined by a camera center $\boldsymbol{x}_s \in \mathbb{R}^3$, a direction vector $\boldsymbol{d}_s \in \mathbb{R}^3$ from the camera center through the image plane at the pixel coordinate of the contour point, and the vertex position $\boldsymbol{v}_s \in \mathbb{R}^3$ of the posed model. We define the silhouette loss as:

$$L_{\text{sil}}(\boldsymbol{V}, \boldsymbol{\theta}_i, \mathcal{S}_i) = \frac{1}{n_I} \sum_{j=1}^{n_I} \sum_{s \in \mathcal{S}_i^j} \frac{1}{\left| \mathcal{S}_i^j \right|} \left\| \left(\boldsymbol{I} - \boldsymbol{d}_s \boldsymbol{d}_s^\mathsf{T} \right) \left(\boldsymbol{x}_s - \text{skin}_v(\boldsymbol{v}_s, \boldsymbol{\theta}_i) \right) \right\|^2, \tag{6.2}$$

where n_I is the number of images in a sub-scan, $\boldsymbol{I}$ is the 3×3 identity matrix and skin_v applies linear blend skinning to a single vertex $\boldsymbol{v}_s$, returning its skinned position as defined in Section 3.1.1. This loss measures the sum of squared distances between all silhouette correspondences (vertex $\boldsymbol{v}_s$ and the line defined by camera center $\boldsymbol{x}_s$ and direction $\boldsymbol{d}_s$) over all sub-scans. It is used to restrict the template model to remain inside the visual hull defined by the masks of the input images.

Regularization

Similar to other template fitting approaches, we use a Laplacian-based regularization to penalize deviation from the template's curvature (i.e., bending):

$$L_\Delta(\boldsymbol{V}) = \frac{1}{\sum_{v \in \boldsymbol{V}} A_v} \sum_{v \in \boldsymbol{V}} A_v \left\| \Delta \boldsymbol{v} - \boldsymbol{R}_v \Delta \overline{\boldsymbol{v}} \right\|^2, \tag{6.3}$$

where $\Delta v \in \mathbb{R}^3$ is the cotangent Laplacian of the deformed vertex v, $R_v \Delta \overline{v} \in \mathbb{R}^3$ is the rotated cotangent Laplacian of the undeformed vertex $\overline{v}$, and A_v is the Voronoi area associated with vertex v [Bot+10].

6.1.4 Coarse-to-Fine Fitting

After defining the sub-scan-based loss functions, we now give a detailed description of our fitting process. The sub-scan point clouds cover less area than the full-body point cloud or miss parts of the scanned subject (e.g., arms). Therefore, we use the full-body point cloud, generated from all body images $\mathcal{I}_b$, to initialize the template model. After initialization, we can use the sub-scans to finely fit the model's geometry.

Template Initialization

After aligning the head, body, and sub-scan point clouds, we align our template to the point clouds by computing a conformal transformation to match landmarks detected with OpenPose [Cao+19] and positions of the template's joints $\mathcal{J}$. Depending on the number of sub-scans, the point clouds only contain areas seen from a few directions and thus lack volume information. This makes approximating proportions of body parts much more difficult. To compensate for this, we use the full-body point cloud for initialization, as large parts are static and can be used to coarsely match the shape of the body.

We minimize the distance between joint positions and the detected landmarks using inverse kinematics [Bus04] to get a first approximation of the subject's pose. The pose and shape are then further refined by alternately optimizing L_{cpc} (Equation (6.1)) with respect to joint angles $\boldsymbol{\theta}_i$ as well as vertex positions $\boldsymbol{V}$ through the parameters $\boldsymbol{\beta}$ of our PCA model (see Figure 6.3, bottom to center).

Coarse Body Fitting

Accurate pose estimation, which is crucial for fine-scale sub-scan fitting, requires that the template and subject share similar body proportions. Our PCA shape model, trained on minimally clothed CAESAR scans, cannot explain the extra volume introduced by loose garments. After the initialization, we perform a coarse non-rigid fit of the model to the full-body point cloud. This lets the surface expand to match the clothing, providing a better initialization for subsequent pose refinement. This is done by minimizing L_{cpc} (Equation (6.1)) with respect to vertex positions using correspondences $\mathcal{C}$. We additionally fine-fit to the head point cloud, as it does not contain motion-induced artifacts.

We use L_Δ (Equation (6.3)) to penalize deviation from the template model to ensure a smooth surface. Thus, the fitting loss is defined as

$$L_{\text{fit}}(\boldsymbol{V}, \boldsymbol{\theta}) = L_{\text{cpc}}(\boldsymbol{V}, \boldsymbol{\theta}, \mathcal{C}) + \tau_\Delta L_\Delta(\boldsymbol{V}) \,.$$

This optimization is performed iteratively, with the regularization weight τ_Δ being gradually reduced. As the full-body point cloud may contain motion-

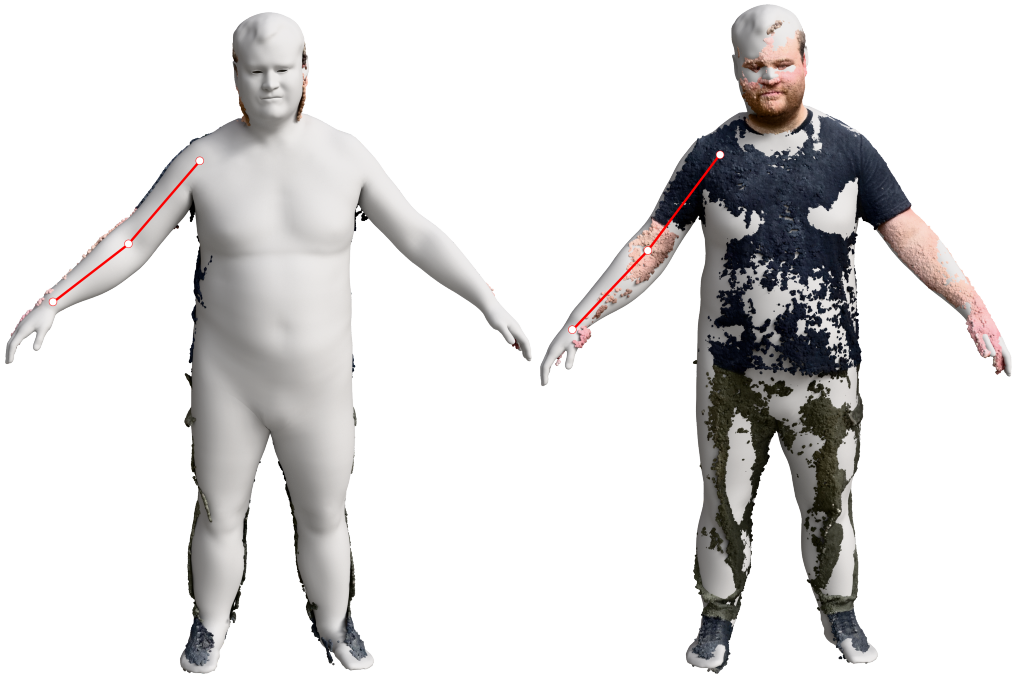


Figure 6.6: The template's pose is fitted to each point cloud individually while not differing too much from the first initialization. Left: Sub-scan 3, Right: Sub-scan 5. Slight movements of the subject's right arm are noticeable. The moving right arm is marked with red.

induced artifacts, which are incorporated into the fitted mesh when the regularization weight becomes too low, we fit with τ_Δ in the range from 1 to 10^{-4}. In contrast, we fit the head point cloud until τ_Δ reaches 10^{-9}, as it does not contain motion-induced artifacts. We use iterative block coordinate descent for optimization.

Fine-Scale Sub-Scan Fitting

Although the point clouds of the sub-scans are aligned in the same coordinate system, the poses of the sub-scans do not match. Similar to the full-body point cloud, we need to adjust the pose per sub-scan. We therefore optimize L_{cpc} (Equation (6.1)) with respect to $\boldsymbol{\Theta}$ (see Figure 6.6).

After adjusting the model's pose $\boldsymbol{\theta}_i$ to each sub-scan, we fit the model using non-rigid registration by minimizing L_{cpc} (Equation (6.1)) and L_{sil} (Equation (6.2)) with respect to $\boldsymbol{V}$ over all sub-scans (see Figure 6.3, left and right). Our overall fitting loss is then defined as:

$$L_{\text{fit}}(\boldsymbol{V}) = \frac{1}{m_S} \sum_{i=1}^{m_S} \left[L_{\text{cpc}}(\boldsymbol{V}, \boldsymbol{\theta}_i, \mathcal{C}_i) + \tau_{\text{sil}} L_{\text{sil}}(\boldsymbol{V}, \boldsymbol{\theta}_i, \mathcal{S}_i) \right] + \tau_\Delta L_\Delta(\boldsymbol{V}) \,. \tag{6.4}$$

Similarly to the coarse body fitting, this loss is optimized using iterative block coordinate descent. We use $\tau_{\text{sil}} = 10^{-4}$ and, as before, we gradually reduce λ_Δ from 1 to 10^{-9}. This ensures the same level of detail as for the



Figure 6.7: Original masked image (left), rendered image (middle), weighting by view direction (right).

head scan. Additionally, we transfer the template's blendshapes to the fitted model via deformation transfer [SP04]. This completes the geometry fit of our pipeline.

6.1.5 Texture Generation

After fine-scale fitting the avatar's geometry, we compute a high-resolution texture using an inverse rendering framework. We define a loss function between our rendered mesh and the input image as:

$$L_{\text{tex}}(\boldsymbol{I}_{\text{tex}}, \boldsymbol{I}_{\text{in}}, \boldsymbol{I}_{\text{ren}}) = L_{\text{Ch}}(\boldsymbol{I}_{\text{in}}, \boldsymbol{I}_{\text{ren}}) + L_{\text{SSIM}}(\boldsymbol{I}_{\text{in}}, \boldsymbol{I}_{\text{ren}}) + L_{\text{TV}}(\boldsymbol{I}_{\text{tex}}),$$

where $\boldsymbol{I}_{\text{ren}}$ is a rendered image of the fitted mesh (see Figure 6.7, center), $\boldsymbol{I}_{\text{tex}}$ is the texture of the mesh, $\boldsymbol{I}_{\text{in}}$ is the segmented input image with white background (see Figure 6.7, left), L_{Ch} is the differentiable Charbonnier L_1 loss [Cha+94], L_{SSIM} is the structural similarity index (SSIM, [Wan+04]), and L_{TV} is the total variation loss (TV). We additionally weight L_{SSIM} and L_{Ch} per image by view-direction. The weighting is used to downweight parts of the image that do not face the camera and are thus less likely to be accurate. The loss function for texture optimization over all sub-scans is then defined as

$$L_{\text{tex}}(\boldsymbol{V}, \boldsymbol{I}_{\text{tex}}) = \sum_{i=1}^{m_S} \sum_{j=1}^{n_I} L_{\text{tex}}\left(\boldsymbol{I}_{\text{tex}}, \boldsymbol{I}_i^j, \Psi(\boldsymbol{V}, \boldsymbol{I}_{\text{tex}}, \boldsymbol{\theta}_i)\right), \tag{6.5}$$

where $\boldsymbol{I}_i^j$ is the j-th image of the i-th sub-scan, $\Psi(\boldsymbol{V}, \boldsymbol{I}_{\text{tex}}, \boldsymbol{\theta}_i)$ is a function to render the fitted mesh $\boldsymbol{V}$ with texture $\boldsymbol{I}_{\text{tex}}$, posed with $\boldsymbol{\theta}_i$, on a white background.

We begin the texture generation by using Metashape to generate a texture based on our fitted geometry and the complete set of body images $\mathcal{I}_b$. However, due to subject movement, this initial texture often contains artifacts, such as

parts of the background. This texture is then refined using the inverse rendering framework provided by PyTorch3D [Rav+20] – a library for deep learning with 3D data. We optimize our loss L_{tex} by minimizing the difference between our rendered avatar and the input images through adjusting the texture $\boldsymbol{I}_{\text{tex}}$. Because scans are performed in environments with uncontrolled lighting conditions, the lighting and material parameters are unknown. Therefore, the avatar is rendered without lighting, which leads to baked-in lighting in the texture. This optimization takes advantage of the SSIM, TV, and differentiable Charbonnier L_1 loss functions from Kornia [Rib+20]. For optimization, we use the Adam optimizer [KB14]. Afterward, a head texture is generated via Metashape from the set of head images and merged with the body texture using Poisson image editing [PGB03], similar to our previous approach (see Section 5.1.2). The texture generated by Metashape already provided sufficient visual quality, thus, we did not use our inverse rendering approach for the head texture, as this would introduce additional computational cost without a clear benefit.

6.2 Evaluation

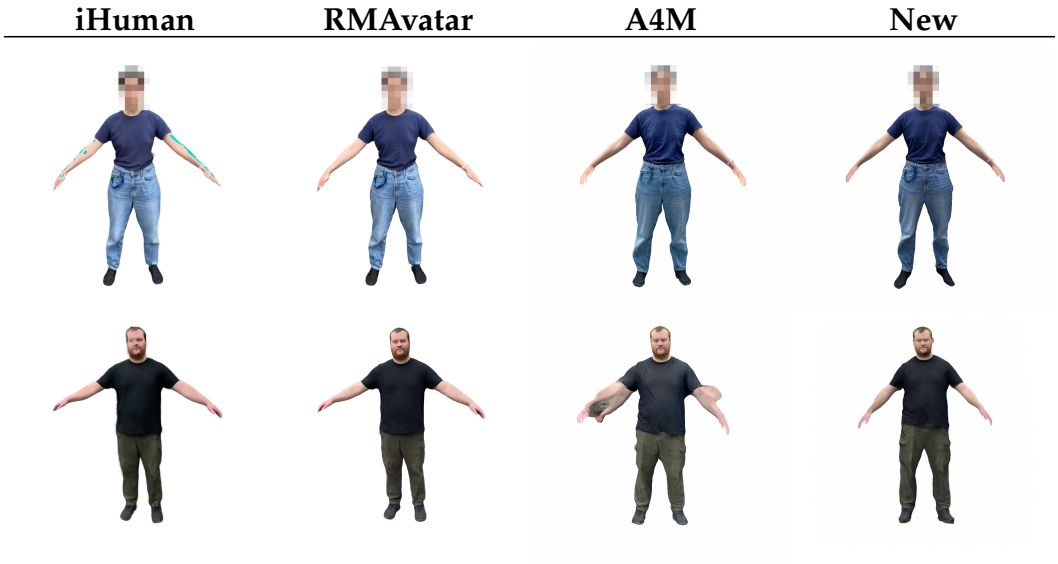


Figure 6.8: Comparison of the different avatar reconstructions in A-pose from the training sequence. The results from iHuman are blurry and can contain differently colored artifacts, RMAvatar produces the most photorealistic results, A4M (Chapter 5) contains visible motion artifacts in the arms. Our new result is geometrically accurate and has a sharp texture. It does not contain clearly visible motion artifacts, although it is also a purely mesh-based approach.

We compare our new approach against three other recent smartphone-based avatar reconstruction methods. *RMAvatar* [Pen+25] and *iHuman* [Pau+25] are

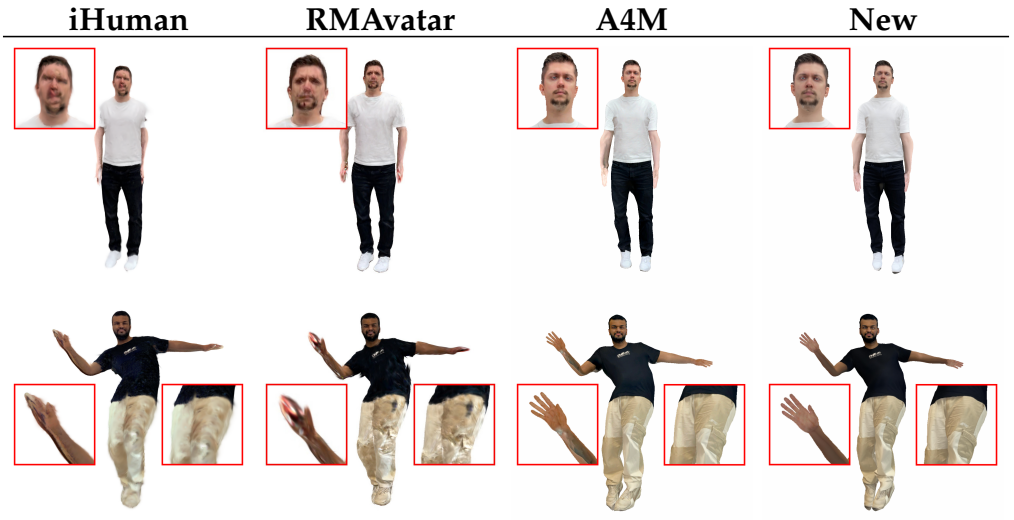


Figure 6.9: Comparison of the different avatar reconstructions in a novel pose. All resulting avatars can be animated using a precomputed animation, but the results of iHuman and RMAvatar have visible artifacts in parts that were not seen (from this angle) in the training sequence (e.g., face, arm, or leg). In contrast, the purely mesh-based approaches (A4M and our new approach) do not produce these noticeable artifacts in geometry.

3DGS-based approaches, *Avatars for the Masses* (A4M) is our previous mesh-based reconstruction approach (see Chapter 5). Our avatars consist of a set of rigged triangle meshes with 52 blendshapes and are about 25 MB in size in GLB format, including texture. In this section, we show that while 3DGS-based avatar reconstruction methods are excellent in reconstructing views and poses similar to training data, they struggle with creating *novel* poses and views. This, however, is crucial for VR applications as users control their avatar, creating unpredictable poses and views.

All methods use monocular cameras/smartphones to capture the necessary data for their algorithms. A4M and our new approach use photos that are taken by circling a person standing in A-pose. To capture the images, we used the smartphone app from our previous approach A4M (see Section 5.1.1). RMAvatar and iHuman use short videos as input, where the subject is moving in front of the camera. We followed the input protocol suggested by the authors and recorded a short 1440×1440 video of a person rotating in A-pose in front of a static smartphone. iHuman's and RMAvatar's provided reconstruction pipelines expect scans in PeopleSnapshot [All+18b] format with refined SMPL [Lop+15] poses through Anim-NeRF. We therefore convert the videos by using VideoAvatars [All+18b] and Anim-NeRF [Pen+21] and subsequently process the resulting data using the provided scripts of RMAvatar and iHuman, as proposed by the authors. We recorded the videos for iHuman and RMAvatar with the same iPhone that we used to capture the input images for A4M and our new approach. We also tried to train RMAvatar and iHuman with our input images, using the same preprocessing pipeline as for the videos.

However, this resulted in reconstructions of lower quality. Therefore, for a fair comparison, we used the input data that resulted in the best reconstructions for each method.

6.2.1 Qualitative Evaluation

Figures 6.8 and 6.9 show the results of iHuman [Pau+25], RMAvatar [Pen+25], our previous approach A4M (see Chapter 5), and our new method. Figure 6.8 shows the reconstructions in a pose of the training sequence. For the female subject[1], all methods created plausible shapes/geometry. Regarding the color/texture, the results are different. iHuman has produced visible turquoise artifacts on the arms and the result is blurry, e.g., at the legs. The other methods computed better colors/textures, with RMAvatar producing the most photorealistic results. For the male subject, the result is mostly similar. iHuman produced blurrier results than RMAvatar and our new method (e.g, in the face). However, A4M failed to correctly reconstruct the person's geometry, with severe artifacts on both arms. The model's right arm contains a partial image of a car tire/headlight from the background. These artifacts result from motion during scanning. Note that the male subject was not instructed to keep his arms still, highlighting the effect of motion on reconstruction. A4M assumes that during the photo capture no movement occurs, leading to severe deformations if the assumption is violated.

Figure 6.9 depicts two subjects in a novel pose that was not part of the training sequence. A4M's and our new results produce plausible geometries and sharp textures, although the texture of our new result exhibits fewer artifacts, e.g., the lower arm of the bottom subject. In contrast, the results of iHuman and RMAvatar are blurrier than in the training pose and show strong deformations in the arms, legs, and face, resulting in the loss of identifiable features. These differences between the training pose and the new pose result from the 3DGS reconstruction method. In contrast to the purely mesh-based approaches, the 3DGS-based approaches incorporate view-dependent material appearance through spherical harmonics. This is usually a key advantage for modeling fine details (e.g., hair); in this case, however, it becomes a disadvantage due to the lack of training data for the novel pose and novel view. Our experiments show that a short smartphone video is not sufficient to capture enough views to create novel poses faithfully. However, this is a key necessity for VR usage, as users control their avatars, leading to a variety of novel views and poses. Therefore, current smartphone-3DGS-based methods cannot be used to generate VR-ready avatars. More results and edge cases are shown in Appendix B and the accompanying video of the paper publication[2].

Metric	iHuman	RMAvatar	A4M	New
SSIM ↑	0.957	**0.983**	0.937	0.948
PSNR ↑	25.677	**31.780**	23.747	24.395
IoU ↑	98.90 %	99.21 %	99.04 %	**99.24 %**
Time ↓	25 h	26.5 h	21 min	**19 min**
Frame Rate ↑	161 fps	115 fps	**4615 fps**	**4615 fps**

Table 6.1: Comparison of SSIM, PSNR, IoU errors, computation time, and rendering performance (fps) with three state-of-the-art avatar reconstruction methods and our new method averaged over 11 subjects (see Appendix B, subjects 1–11). Best values are printed in bold. In SSIM and PSNR, the 3DGS-based iHuman and RMAvatar are more accurate. Our new method is the most accurate with respect to IoU and is the fastest, with 19 min processing time. iHuman and RMAvatar need more than 25 h, while A4M takes slightly longer with 21 min. We improve upon our previous approach, A4M, in every metric. iHuman and RMAvatar have considerably lower rendering performance than A4M and our new results.

6.2.2 Quantitative Evaluation

We quantitatively evaluate the avatars by reprojecting them onto the masked input images. The masked background is replaced with a plain white background color, and the metrics are evaluated by comparing the rendered and the input image. We report SSIM, peak-signal-to-noise ratio (PSNR), intersection over union (IoU), and computation time results in Table 6.1.

PSNR and SSIM measure the reprojection accuracy of the avatar on RGB images, thereby evaluating geometry and texture reconstruction quality. IoU, in contrast, is evaluated on binary images, therefore measuring the difference between the rendered avatar's silhouette and the silhouette of the scanned person. To compute the metrics, we used OpenCV's [Bra00] *quality* module. Compared to 3DGS-based approaches, our new avatars achieve similar image metrics: The SSIM and PSNR values are slightly below iHuman's and considerably lower than RMAvatar's, but we outperform A4M in both metrics. Our method achieves the highest IoU, with RMAvatar close behind.

Our pipeline was run on a workstation equipped with an AMD Ryzen 7950X CPU and an NVIDIA RTX 4090 GPU. The preprocessing and training of iHuman and RMAvatar were performed on a compute server with an AMD Ryzen Threadripper PRO 5975WX CPU and three NVIDIA RTX 6000 GPUs. Avatars from A4M were computed by the provided Mac Studio server (M1 Ultra 20-core CPU, 64-core GPU, and 64GB RAM). We report computation times, including all preprocessing steps and the actual training/fitting of the different pipelines, to compare the full duration from scanning to using the avatars. Rendering speed was evaluated on an NVIDIA RTX 4090 GPU-

1 The subject asked to remain anonymous
2 https://doi.org/10.1145/3756884.3765995

powered machine in 2160 × 2160 resolution. Over 11 subjects, RMAvatar took the longest, with average computation times of 26.1 hours. This includes 25 hours of preprocessing with Anim-NeRF and VideoAvatars, as well as around 1.1 hours of additional training time. As iHuman uses the same preprocessing, it shares the 25 hours of preprocessing time but only adds approximately two minutes of extra training time. Results from A4M were received after around 21 minutes, while our new method took around 19 minutes when using 3 sub-scans. Therefore, our new approach outperforms every compared approach in processing time. Rendering is 28–40× faster than the 3DGS-based methods and matches our previous approach A4M, since both use the same template model. The iHuman and RMAvatar avatars consist of approximately 200k and 120k Gaussians, respectively, while A4M and our new mesh-based avatars have around 24k vertices and 48k triangles.

Overall, our new results are slightly worse in SSIM and PSNR compared to RMAvatar and iHuman, but improve on IoU. However, as shown in Figure 6.9, our new results do not exhibit novel pose artifacts in contrast to the 3DGS-based approaches. Our pipeline runs slightly faster than A4M, while iHuman and RMAvatar take more than a day to finish the reconstruction. Compared to A4M, our new results exhibit no motion artifacts and improve on reconstruction accuracy in every metric measured.

6.2.3 Ablation Studies

In this section, we show the influence of the number of sub-scans and the usage of silhouette constraints on the fitting quality.

Sub-Scans

First, we show the impact of using sub-scans by varying the number of sub-scans from two to five with one image overlap between two consecutive sub-scans. As we have 45 images for the body, higher amounts of sub-scans would result in using fewer than seven images per sub-scan, which are too few perspectives to cover enough area of the scanned person in each sub-scan. Figure 6.10 shows how the usage of sub-scans influences the pose alignment of the avatar to the input images and the geometric accuracy. The subject's right arm is misaligned and malformed when using no sub-scans (left). Using two sub-scans greatly improved the alignment (center), and with five sub-scans, the pose and geometry of the scanned person are matched almost perfectly (right). The geometric detail of the model is preserved in most regions. Cloth detail and skin structure are local features that are reconstructed even when using partial point clouds from five sub-scans. This is due to the fact that increasing the number of sub-scans increases the likelihood of getting motion-free sequences. Our per-sub-scan aligned template fitting is then able to reconstruct pose and geometry faithfully. This is also visible in the texture. Without sub-scans, the texture of the subject's right arm contains areas of the



Figure 6.10: Comparison of avatar renderings in scan pose over input images (transparent) using 0 (left), 2 (middle), and 5 (right) sub-scans. An increasing number of sub-scans improves pose alignment and geometric accuracy (right arm).

background of the input images. With sub-scans, this issue is significantly reduced.

Although the error-reduction effect is generally observed, the data indicate that using too many sub-scans can increase uncertainty. When increasing the number of sub-scans from two to four, the average PSNR increases from 25.35 to 25.58. However, at five sub-scans, PSNR falls to 25.49, because the effective overlap per scan becomes too small. Similar effects can be observed with SSIM and IoU. Data from our eleven scanned subjects suggest that three to four sub-scans produce the best results. A larger number of sub-scans can help in cases of exaggerated motion (e.g., Figure 6.10); however, with proper instructions, three sub-scans are sufficient to reduce the impact of motions in general. As computation time increases with the number of sub-scans, we recommend the usage of three sub-scans for a one-minute body scan.

Silhouettes

We use silhouette constraints to make the fitting process in our new pipeline more robust and reduce the influence of motions during the scan procedure even further. Figure 6.11 shows the influence of silhouette constraints on the reconstruction of the lower arm region. We can see that the proportions of the arms are better matched when using silhouette constraints. While the use of sub-scans already eliminates most artifacts, silhouette constraints lead to a better alignment with the original shape and position of the body. The choice of silhouette weight τ_{sil} is crucial. When working with exaggerated motions, the overlap of silhouettes becomes extremely thin. This leads to arms becoming thinner the higher τ_{sil} is chosen. However, our experiments show that 10^{-4} is an appropriate weight for all of our scans. Silhouette constraints have a positive effect on every image metric. This effect is more pronounced when dealing with a lot of motion. On average, PSNR improves from 25.2 to 25.7. SSIM also improves from 0.959 to 0.960.

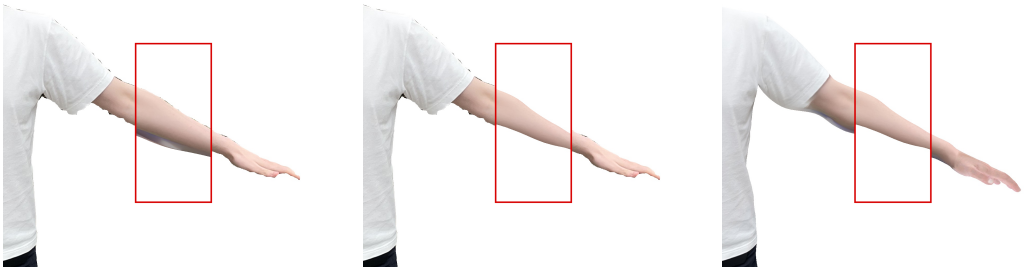


Figure 6.11: Comparison of reconstructions with and without silhouette constraints. The left image shows the lower arm without silhouette constraints (red box) on the input image, the middle image shows it with silhouette constraints (red box) on the input image, and the right image displays the lower arm reconstructed with silhouette constraints (red box) on a rendering of the avatar reconstructed without silhouette constraints. The silhouette constraints restrict the geometry to the visual hull defined by the masks, leading to a more realistic reconstruction.



Figure 6.12: Texture-map comparison: Metashape (left) and graph-cut (center) textures from all images versus our inverse-rendering texture generated from sub-scans (right). Only the sub-scan version removes background bleed-through and stitching artifacts, yielding a smooth, photorealistic texture.

Texture Generation

We compare texture generation from all images with our texture generation from the sub-scans (see Figure 6.12). The textures from all images (body and head scan) are computed using Metashape (left) or a graph-cut algorithm [Wen+20] (center). Our texture from sub-scans is generated with inverse rendering (right). The Metashape and graph-cut textures contain visible dark artifacts and/or parts of the background. In contrast, our texture does not contain these kinds of artifacts. Due to the pose alignment through the sub-scans, the influence of motion is reduced, as the fitted avatar's model is aligning properly with the input images. This results in a smooth, realistic, and coherent texture.

6.2.4 Limitations

Our approach reduces motion-induced errors in photogrammetry-based, template-fitting avatar reconstruction. However, the following limitations remain:

Imprecise Masks Silhouette constraints make the correspondence-driven template fitting algorithm more robust. A necessity for the successful usage is accurate segmentation masks of the scanned subject. If the masks are too imprecise (e.g., missing limbs), these constraints may create additional artifacts.

Motion Our approach reduces the problem of motion-induced errors, but still assumes the person to be standing in A-pose, without intentional motion. While our presented method reduces the influence of slow motion, fast movements during a sub-scan can still lead to artifacts.

Clothes and Hair Clothes, skin, and hair are all represented by the same mesh. This simplification can lead to artifacts for loose clothing or long hair, in particular during animations (see Appendix B and accompanying video of the paper publication[3]).

6.3 Conclusion

In this chapter, we presented measures to account for motion-induced artifacts in our avatar reconstruction. Through the usage of silhouette constraints and sub-scans, we relaxed the assumption that persons remain static during scanning. Our improvements effectively reduced the influence of micro-motion without using more complex scanning procedures or more expensive hardware. Paired with our inverse rendering texture generation, we outperform our initial system in even more challenging conditions. Comparisons with recent 3DGS-based methods show the superiority of our results for the targeted use case of location-independent avatar reconstruction and VR-ready full-body avatars in terms of computation time, accuracy, and animation quality. Our avatars are view-consistent and compatible with common game engines and VR applications without further postprocessing to faithfully represent the users.

Hair and clothing could be improved by using body-part segmentation, which would allow separate fitting to the skin mesh. This approach could also reduce motion restrictions by enabling the fitting of rigid parts individually. An in-headset user study could be conducted to validate the improvements for VR applications. Unlike existing 3DGS-based methods, we would like to combine textured mesh and 3DGS-based rendering to improve the rendering quality of fine details of mesh-based avatars.

3 https://doi.org/10.1145/3756884.3765995

Part II

PERSONALIZED FACIAL ANIMATION FOR AVATARS

7 MOTIVATION

In the previous part, we presented a system to automatically create realistic, full-body avatars from smartphone input, enabling personalized self-representation in VR applications. The avatars contain a skeleton rig, allowing users to adjust the pose and posture of the avatars using VR controllers or more elaborate solutions such as body motion capture systems (e.g., OptiTrack [Nat25] or Captury [Cap23a]). A set of facial blendshapes can be used to create new facial expressions. Both the skeleton rig and the blendshapes are important to convey information, as humans use language together with bodily motion to express thoughts and ideas [WMK14].

While blendshapes are easy to use, only facial expressions within the space spanned by the predefined facial expressions can be created. As we have transferred a set of *generic* blendshapes to the avatars using deformation transfer [Bot+06; SP04], the avatar's facial blendshapes do not contain the unique features of the scanned person. Thus, all newly created facial expressions will not contain those features either. The absence of those unique movements is particularly problematic since humans can detect even subtle changes and deviations in human faces, which can potentially lead to incongruent social cues [LW22; Rot+16]. This problem can be solved through dynamic avatar personalization, i.e., creating a set of personalized blendshapes [Cas+15; Gar+16; IBP15; LWP10] that capture these unique features. However, existing methods differ in terms of usability, reconstruction accuracy, and applicability.

In this part, we address the described issues by introducing an automated blendshape personalization pipeline. As before, we use smartphone technology (i.e., iOS devices equipped with ARKit), making our approach accessible since no expensive camera rig is needed. We begin by capturing example expression masks using a custom iOS application. From these expression masks, we compute personalized blendshape masks using a position-based version of the widely used example-based facial rigging (EBFR) approach by Li et al. [LWP10]. In the final step, we seamlessly implant the blendshape masks into the personalized avatar. Our position-based EBFR variant reduces the reconstruction error in some cases to 10%, while keeping computational cost comparable to the original method.

However, personalized blendshapes alone are not sufficient for faithful face animation. To complement verbal behavior, it is necessary to generate a congruent set of blendshape weights. Otherwise, the resulting facial expression will not be faithful, even when using the personalized blendshapes. Depending on the situation, mainly two methods are used for this task. When the face is visible, it is possible to track the actor's facial expression directly. This can be done using consumer devices (e.g., a smartphone) or more professional setups specifically designed for face capture (e.g., Faceware [Fac]). When the face is not visible, e.g., in VR scenarios where the user wears an HMD (without a

face tracking option) or when an avatar is controlled by a conversational agent, blendshape weights can be synthesized from text or audio input.

This motivates the question of which method yields more natural and plausible facial expressions. To answer this, we evaluate representative methods for blendshape weight generation by comparing ARKit-based facial capture [App26] and Meta's Oculus Avatar SDK [Met23] for facial expression synthesis based on speaker audio, head movement, and gaze targets, as well as a static baseline.

The remainder of this part is structured as follows:

Chapter 8 introduces the fundamental concepts for this part. We first define linear blendshapes for facial animation and introduce the concept of deformation gradients. We then explain EBFR by Li et al. [LWP10], which serves as the basis for our blendshape personalization approach.

Chapter 9 then reviews recent advancements regarding face expression personalization techniques and gives an overview of state-of-the-art blendshape weight generation techniques.

Building on EBFR, Chapter 10 presents a position-based blendshape personalization approach that reconstructs facial expressions more accurately than standard EBFR and deformation transfer, requiring only a smartphone for data capture.

Chapter 11 then evaluates different techniques to animate facial expressions, particularly ARKit [App26] and Meta's Oculus Avatar SDK [Met23]. We assess the naturalness and plausibility of avatar facial animation using smartphone-based expression tracking, synthesizing expressions from audio input, and no animation.

8 FUNDAMENTALS

In this section, we introduce the concepts of blendshapes for facial animation (Section 8.1) and deformation gradients (Section 8.2). We then describe the example-based facial rigging approach by Li et al. [LWP10] (Section 8.3), which forms the foundation of our blendshape personalization method in Chapter 10.

8.1 Blendshapes

A simple yet effective way to animate the faces of 3D characters is to use blendshapes, also called morph targets [Lew+14]. A facial blendshape represents the character's mesh in a predefined facial expression (e.g., a smile or an open mouth). With a set of blendshapes covering a variety of expressions, new facial expressions can be generated by linearly blending the different shapes. A facial blendshape rig consists of a base mesh with a neutral facial expression with N_B vertices $\widetilde{\boldsymbol{B}}_0 \in \mathbb{R}^{N_B \times 3}$ and a set of m_B expression poses $\{\widetilde{\boldsymbol{B}}_1, \ldots, \widetilde{\boldsymbol{B}}_{m_B}\}$, called blendshapes. Each blendshape $\widetilde{\boldsymbol{B}}_j \in \mathbb{R}^{N_B \times 3}$ has the same triangulation as the neutral expression. A new facial expression can then be generated by blending the different expression poses using a set of blendshape weights $\boldsymbol{\alpha} = (\alpha_0, \ldots, \alpha_{m_B})$ with $\alpha_j \in [0,1]$ and $\sum_j \alpha_j = 1$. The resulting facial expression $\boldsymbol{E}(\boldsymbol{\alpha})$ can be computed using the following equation:

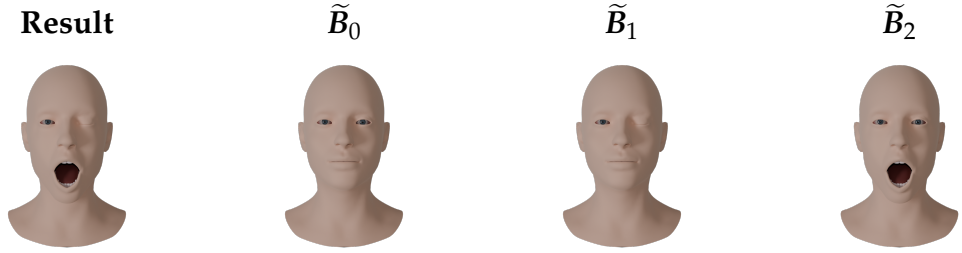
$$\boldsymbol{E}(\boldsymbol{\alpha}) = \sum_{j=0}^{m_B} \alpha_j \widetilde{\boldsymbol{B}}_j.$$

This formulation allows us to generate an arbitrary expression within the space of the predefined blendshapes. However, controlling the blendshape weights $\boldsymbol{\alpha}$ to create a desired expression is a non-trivial task. Therefore, an equivalent formulation exists, where the generation formula is rewritten to blend displacements $\boldsymbol{B}_j$ relative to the neutral expression $\boldsymbol{B}_0$ to create a new facial expression:

$$\begin{aligned} \boldsymbol{E}(\boldsymbol{w}) &= \boldsymbol{B}_0 + \sum_{j=1}^{m_B} w_j \boldsymbol{B}_j, \\ \boldsymbol{B}_0 &:= \widetilde{\boldsymbol{B}}_0, \\ \boldsymbol{B}_j &:= \widetilde{\boldsymbol{B}}_j - \boldsymbol{B}_0. \end{aligned}$$

Here, the blendshape weights $\boldsymbol{w} = (w_1, \ldots, w_{m_B})$ with $w_j \in [0,1]$ are independent and do not have to sum up to one. This allows more intuitive control of the blendshape weights (see Figure 8.1).

For clarity, we use the term *blendshape* to denote delta-blendshapes throughout the remainder of this thesis.



$E(-1,1,1)$	$=$	$-1 \cdot \widetilde{B}_0$	$+$	$1 \cdot \widetilde{B}_1$	$+$	$1 \cdot \widetilde{B}_2$
$E(1,1)$	$=$	B_0	$+$	$1 \cdot B_1$	$+$	$1 \cdot B_2$
	$=$	B_0	$+$	$1 \cdot (\widetilde{B}_1 - B_0)$	$+$	$1 \cdot (\widetilde{B}_2 - B_0)$

Figure 8.1: New facial expression generation using linear blendshapes (top equation) and linear delta-blendshapes (bottom equation). Linear blendshapes compute a weighted average from all predefined facial expressions, including the neutral pose. Linear delta-blendshapes start from the neutral pose and add weighted differences to it. Both models represent the same space of facial expressions, but delta-blendshapes allow more intuitive control.

8.2 Deformation Gradients

A blendshape rig uses per-vertex displacement vectors to change a source mesh (neutral expression) to a target mesh (e.g., mouth open). Another way to describe the deformation from the source to the target mesh is through *deformation gradients* [SP04]. Deformation gradients describe the local transformation of a surface patch and are invariant to global translation. They are therefore well-suited for optimization problems on a per-triangle basis. In the continuous setting, the deformation gradients correspond to the Jacobian of the local deformation mapping. For a triangle mesh, deformation gradients are defined per triangle and map each source triangle to the corresponding target triangle. We can approximate the deformation gradient by constructing local frames for the mesh's N_F triangles. The local frame $\boldsymbol{D}_k \in \mathbb{R}^{3\times3}$ for the k-th triangle is computed as follows:

$$\boldsymbol{D}_k = \begin{pmatrix} \boldsymbol{v}_{k,3} - \boldsymbol{v}_{k,1} & \boldsymbol{v}_{k,2} - \boldsymbol{v}_{k,1} & \boldsymbol{n}_k \end{pmatrix},$$

where $\boldsymbol{v}_{k,1}, \boldsymbol{v}_{k,2}, \boldsymbol{v}_{k,3} \in \mathbb{R}^3$ denote the vertex positions of the k-th triangle and $\boldsymbol{n}_k = (\boldsymbol{v}_{k,3} - \boldsymbol{v}_{k,1}) \times (\boldsymbol{v}_{k,2} - \boldsymbol{v}_{k,1})$ is the unnormalized normal vector of the k-th triangle. Therefore, a local frame comprises two edges of a triangle and its normal vector, thereby describing the triangle's shape and orientation. We compute the local frames for all triangles in the source mesh and the target mesh, and can compute a deformation gradient $\boldsymbol{F}^{s\to t} \in \mathbb{R}^{3\times3}$ mapping a source triangle s to the corresponding target triangle t with:

$$\boldsymbol{F}^{s\to t} = \boldsymbol{D}_t \cdot \boldsymbol{D}_s^{-1}.$$

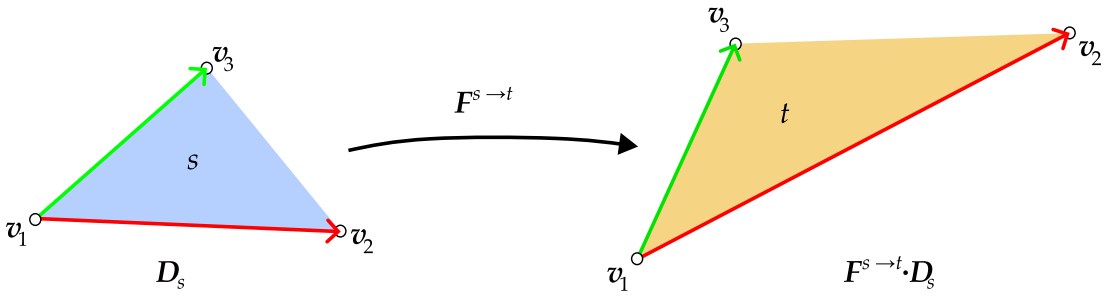


Figure 8.2: The local frame $\boldsymbol{D}_s$ (red and green arrows) of a source triangle (blue) is deformed using the deformation gradient $\boldsymbol{F}^{s\to t}$. The orange triangle shows the deformed local frame $\boldsymbol{F}^{s\to t} \cdot \boldsymbol{D}_s$. For clarity, the normal component of the local frames is not shown.

With these deformation gradients and the local frames of the source triangles, we can compute the local frames of the target triangles (see Figure 8.2). This can also be applied to the previously introduced blendshape model. The deformation gradient for a triangle from the neutral face $\boldsymbol{B}_0$ to the triangle in the j-th blendshape $\boldsymbol{B}_j$ is computed with:

$$\boldsymbol{F}_k^{B_0 \to B_0 + B_j} = (\boldsymbol{D}_k^{B_0} + \boldsymbol{D}_k^{B_j})(\boldsymbol{D}_k^{B_0})^{-1}, \tag{8.1}$$

where $\boldsymbol{D}_k^{B_0}$ is the local frame of a triangle in the neutral pose and $\boldsymbol{D}_k^{B_j}$ is the local frame offset of the corresponding triangle in the j-th blendshape.

After computing the local frames for each triangle, we have a description of the complete mesh in the deformed state; however, we do not yet have the vertex positions. Those have to be recalculated from the resulting local frames by solving the following Poisson problem:

$$\boldsymbol{G}^\mathsf{T} \boldsymbol{A} \boldsymbol{G} \boldsymbol{V} = \boldsymbol{G}^\mathsf{T} \boldsymbol{A} \begin{pmatrix} (\boldsymbol{F}_1 \boldsymbol{D}_1)^\mathsf{T} \\ \vdots \\ (\boldsymbol{F}_{N_F} \boldsymbol{D}_{N_F})^\mathsf{T} \end{pmatrix}, \tag{8.2}$$

where $\boldsymbol{G} \in \mathbb{R}^{3N_F \times N_B}$ is the standard discrete gradient matrix, $\boldsymbol{A} \in \mathbb{R}^{3N_F \times 3N_F}$ is a diagonal weighting matrix containing per-triangle areas, $\boldsymbol{V} \in \mathbb{R}^{N_B \times 3}$ is a matrix containing the unknown vertex positions, and $\boldsymbol{F}_k \boldsymbol{D}_k \in \mathbb{R}^{3\times 3}$ is the deformed local frame of the k-th triangle. As shown by Botsch et al. [Bot+06], the normal component of each local frame does not affect the solution, and can therefore remain in the right-hand side.

8.3 Example-Based Facial Rigging

We have now seen the widely used linear blendshape model for face animation and the concept of deformation gradients to describe the deformation of a source to a target mesh on a per-triangle basis. We now explain the approach *example-based facial rigging* (EBFR) by Li et al. [LWP10], which uses deformation

gradients to compute a set of personalized facial blendshapes, as this is the baseline to our method in Chapter 10. The following description follows the method of Li et al. [LWP10], but is reformulated using the notation and deformation-gradient framework introduced in the previous sections.

The key concept in EBFR is to create a set of personalized blendshapes for a character in a neutral pose $\boldsymbol{S}_0$ from a set of n_S example facial poses $\{\boldsymbol{S}_1, \ldots, \boldsymbol{S}_{n_S}\}$. These example poses can be 3D scans of a real person or a 3D model in different poses created by an artist. Now, to get the personalized blendshapes, we have several different problems at the same time. On the one hand, we aim to compute the best possible blendshapes to recreate the example poses; and on the other hand, the blendshapes should have a semantic meaning, i.e., a computed blendshape should correspond to an interpretable motion (e.g., mouth open). Furthermore, although we have the example poses, we do not have the blendshapes or the set of blendshape weights corresponding to the (unknown) blendshapes for the example poses. Regarding the semantic interpretability, EBFR solves this issue by introducing a template character model already containing a neutral pose $\boldsymbol{T}_0$ and a set of blendshapes $\{\boldsymbol{T}_1, \ldots, \boldsymbol{T}_{m_B}\}$. This template blendshape set serves as a prior for the unknown blendshapes to ensure interpretable results.

The previously shown blendshape model and deformation gradients require all models to share the same triangulation (in case no mapping is used). Therefore, the first step is to transfer the topology of the template character to the target character and to all example poses. This can be done using non-rigid registration (see Chapter 3). After transferring the template's topology, all vertices and all triangles correspond to the semantically same location of each model. Since we do not know the blendshape weights for the example poses and the blendshapes, the subsequent optimization is split into two alternating steps. We first optimize for the blendshapes directly and, second, use the resulting blendshapes to compute blendshape weights corresponding to the example pose $\boldsymbol{S}_i$.

Optimizing Blendshapes

EBFR optimizes blendshapes and corresponding weights in an alternating manner, starting with the optimization of blendshapes; however, before the first optimization step is possible, an initial guess of blendshape weights for the example poses is necessary (e.g., manually specified by the user), which corresponds to the template character's set of blendshapes. After that, we can optimize for the unknown blendshapes using the following per-triangle loss function:

$$L_{\text{fit}}^{\text{geo},k} = \left\| \boldsymbol{D}_k^{S_i} - \left(\boldsymbol{D}_k^{B_0} + \sum_{j=1}^{m_B} w_{i,j} \boldsymbol{D}_k^{B_j} \right) \right\|^2, \tag{8.3}$$

where $\boldsymbol{D}_k^{S_i}$ is the local frame of the k-th triangle in the example pose $\boldsymbol{S}_i$, $\boldsymbol{D}_k^{B_0}$ is the local frame of the corresponding triangle in the character's neutral pose, $\boldsymbol{D}_k^{B_j}$ is the local frame offset of the corresponding triangle in the character's j-th blendshape, and $w_{i,j}$ is the j-th blendshape weight to approximate the i-th example pose. This fitting term measures the squared Frobenius norm of the difference between the local frame of the k-th triangle in the example pose $\boldsymbol{S}_i$ and the local frame of the reconstructed triangle using the unknown blendshapes. This fitting term alone is not sufficient to create appropriate blendshapes. Without regularization, the optimization overfits, resulting in blendshapes with artifacts, and this term does not require the blendshapes to have a semantic meaning. Therefore, we need to add a regularization term to prevent this behavior. Here, our template character's blendshapes come into play. We want to regularize the fitting, such that the deformation gradients for the k-th triangle of our target character's blendshapes $\boldsymbol{F}_k^{B_0 \to B_0 + B_j}$ are similar to the deformation gradients for the corresponding triangle of our template character's blendshapes $\boldsymbol{F}_k^{T_0 \to T_0 + T_j}$. Thus, we define the regularization term $L_{\text{reg}}^{\text{geo},k}$:

$$
\begin{aligned}
L_{\text{reg}}^{\text{geo},k} &= \sum_{j=1}^{m_B} \omega_k^j \left\| \boldsymbol{D}_k^{B_j} - \boldsymbol{D}_k^{T_j^*} \right\|^2, \\
\boldsymbol{D}_k^{T_j^*} &:= \boldsymbol{F}_k^{T_0 \to T_0 + T_j} \cdot \boldsymbol{D}_k^{B_0} - \boldsymbol{D}_k^{B_0},
\end{aligned}
\tag{8.4}
$$

where $\boldsymbol{D}_k^{T_j^*}$ denotes the local frame offset of the k-th triangle, obtained by applying the deformation gradient $\boldsymbol{F}_k^{T_0 \to T_0 + T_j}$ of the j-th template blendshape to the target character's neutral local frame $\boldsymbol{D}_k^{B_0}$ and subtracting the original local frame, $\boldsymbol{D}_k^{B_j}$ is the local frame offset of the k-th triangle of the j-th unknown blendshape, and ω_k^j is a per-triangle weight, used to allow or restrict deviation from the template blendshapes. By penalizing differences between the local frame offsets of the resulting blendshapes and those of the template blendshapes, the optimization is encouraged to compute blendshapes with a similar meaning to the template blendshapes. However, the per-triangle weight ω_k^j is computed such that strong deformation in the template blendshapes allows more deviation from them. The idea behind this strategy is to enforce similar regions to deform in the same blendshape but capture individual features, for example, more expressive smiling expressions. This is achieved by evaluating ω_k^j as:

$$
\omega_k^j = \left(\frac{\left(1 + \left\| \boldsymbol{D}_k^{T_j} \right\|\right)}{\left(\kappa + \left\| \boldsymbol{D}_k^{T_j} \right\|\right)} \right)^{\phi},
$$

where $\kappa = 0.1$ and $\phi \geq 1$ are constants.

The complete minimization loss L^k_{geo} for the optimization of the blendshapes is then obtained by adding $L^{\text{geo},k}_{\text{fit}}$ (Equation (8.3)) and $L^{\text{geo},k}_{\text{reg}}$ (Equation (8.4)):

$$L^k_{\text{geo}} = L^{\text{geo},k}_{\text{fit}} + \tau^{\text{geo}}_{\text{reg}} L^{\text{geo},k}_{\text{reg}},$$

where $\tau^{\text{geo}}_{\text{reg}}$ is a weight to balance both terms. This optimization step then results in new local frame offsets $\boldsymbol{D}^{B_j}_k$ and is performed separately for each triangle. The resulting blendshapes' vertex positions can then be obtained with the local frames of the character's mesh in neutral pose and Equations (8.1) and (8.2).

Optimizing Weights

In the previous step, we computed the best blendshape representation given the manually defined blendshape weights from the beginning. As these do not necessarily yield the ideal combination of the blendshapes to recreate the example poses, in this step, we keep the personalized blendshapes fixed and optimize for a new blendshape weight configuration. After that, the new blendshape weights are used as input for the blendshape optimization step again.

Similar to the blendshape optimization, the blendshape weight optimization loss consists of two terms, a fitting term and a regularization term. The fitting term $L^{\text{weight},i}_{\text{fit}}$ measures the sum of squared per-vertex distances between the reconstructed expression using our personalized blendshapes and the i-th example pose $\boldsymbol{S}_i$, and the regularization term $L^{\text{weight},i}_{\text{reg}}$ ensures the optimized blendshape weights stay close to the initially specified blendshape weights, as we want to iteratively improve the blendshape weight estimation. The full optimization loss is evaluated as follows:

$$\begin{aligned} L^i_{\text{weight}} &= L^{\text{weight},i}_{\text{fit}} + \tau^{\text{weight}}_{\text{reg}} L^{\text{weight},i}_{\text{reg}}, \\ L^{\text{weight},i}_{\text{fit}} &= \sum_{k=1}^{N_B} \left\| \boldsymbol{v}^{S_i}_k - \left(\boldsymbol{v}^{B_0}_k + \sum_{j=1}^{m_B} w_{i,j} \boldsymbol{v}^{B_j}_k \right) \right\|^2, \\ L^{\text{weight},i}_{\text{reg}} &= \sum_{j=1}^{m_B} (w_{i,j} - w^*_{i,j})^2, \end{aligned} \tag{8.5}$$

where $\boldsymbol{v}^{S_i}_k \in \mathbb{R}^3$ is the position of the k-th vertex of the target pose $\boldsymbol{S}_i$, $\boldsymbol{v}^{B_0}_k \in \mathbb{R}^3$ is the position of the k-th vertex of the character's neutral pose, $\boldsymbol{v}^{B_j}_k \in \mathbb{R}^3$ is the displacement vector of the k-th vertex of the character's j-th blendshape, $w_{i,j}$ are the unknown blendshape weights, $w^*_{i,j}$ are the initially specified blendshape weights for the template blendshapes, and $\tau^{\text{weight}}_{\text{reg}}$ is a weight to balance fitting and regularization.

This system is similar to the face inverse kinematics system proposed by Lewis et al. [LA10]. In this formulation, the unconstrained minimization yields a linear least-squares problem. However, blendshape weights are defined in the range $[0, 1]$ (although they are often exceeded in practice). This is not ensured by this loss function. To restrict the optimization, it is possible to add inequality constraints. Then, however, the minimization no longer results in a linear system and requires quadratic programming to solve. Similarly, additional constraints can be introduced to ensure specific combinations of blendshapes are impossible (e.g., eyes wide open and eyes closed). Depending on hard or soft constraints, the solving strategy has to be adjusted.

The weight optimization step concludes the optimization strategies in EBFR. Because both steps are performed in an alternating manner, in the end, a set of personalized blendshapes is computed with a corresponding set of blendshape weights that reconstruct the example poses.

9 RELATED WORK

Corresponding Publications *This chapter is based on the related work sections of the following publications:*

1. T. Menzel, M. Botsch, and M. E. Latoschik. "Automated Blendshape Personalization for Faithful Face Animations Using Commodity Smartphones". In: *Proceedings of the 28th ACM Symposium on Virtual Reality Software and Technology*. 2022. DOI: 10.1145/3562939.3565622
2. P. Kullmann, T. Menzel, M. Botsch, and M. E. Latoschik. "An Evaluation of Other-Avatar Facial Animation Methods for Social VR". In: *Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems*. 2023. DOI: 10.1145/3544549.3585617

In this chapter, we review methods for faithful facial animation of virtual avatars. We start by describing approaches to personalize facial blendshapes, which capture the unique facial characteristics of individuals that are necessary to reflect the avatar's identity. After that, we give an overview of blendshape weight generation techniques to drive those personalized blendshapes for plausible and natural facial animation.

9.1 Blendshape Personalization

While many different approaches for face animation have been proposed, such as skeleton and joint models [VS09] or physics-based muscle models [Wat87], linear blendshape models [Lew+14] are still the most widely used technique – in particular in interactive extended reality (XR) applications. The individual blendshapes (or morph targets) are typically based on the facial action coding system (FACS, [EF78]), giving them an anatomical as well as semantic meaning. The FLAME model of Li et al. [Li+17] replaces the over-complete, linearly dependent FACS blendshape basis by an orthogonal PCA basis. While this is indeed better suited for tracking and reconstruction, their basis lacks semantic meaning and is therefore not suitable for several applications. DECA [Fen+21] extends the FLAME model by learning person- and expression-specific geometric details. While the FLAME model provides only a low-dimensional linear expression basis, DECA adds expression-specific details to represent fine wrinkles and folds not contained in this approximation. SoftDECA [WBS23] further builds on this idea by replacing the FLAME model with a learned layered head model to approximate physics-based soft-tissue behavior. However, although both approaches are compatible with facial animation pipelines, neither provides a semantically meaningful basis. To be as widely applicable as possible, our method employs standard linear facial blendshapes.

Generating the required blendshapes by scanning an actor in all these expressions is not possible in most situations. Hence, the blendshapes of a generic face rig are typically transferred to the target avatar, using, for instance, RBF warps [OZS08; Rib+17; Seo+11], non-rigid registration [Gar+13], or variants of deformation transfer [Rob+21; SP04; Thi+18]. The blendshapes generated this way match the anatomical dimensions of the target avatar, but they typically do not faithfully reconstruct the captured person's unique facial expressions.

Higher-quality approaches therefore personalize blendshapes by capturing an actor not only in the neutral pose, but also in a couple of example expressions [Cao+16; Han+21; LWP10; Wei+11]. The optimization involved in the underlying example-based facial rigging process [LWP10] is ill-posed, since both the blendshape meshes of the actor and the blendshape weights of the captured expressions are unknown. The method, therefore, depends on good initial guesses of the blendshape weights, which restricts the poses the actor can (or has to) perform. Li et al. [LWP10] proposed a subset of 15 facial expressions that the actor should perform. These expressions consist of more than one single activated blendshape; they are instead a combination of several activated blendshapes. Carrigan et al. [Car+20] use an even smaller set of facial expressions by packing more basic blendshapes into the training expressions – making them harder to perform, though.

In contrast, Ichim et al. [IBP15] let the actor perform a sequence of dynamic facial expressions and video-record this performance using a smartphone. They detect facial feature points in each frame and non-rigidly deform their blendshapes to better fit these landmarks. However, the user has to check and correct the detected landmarks in about 25 of 1500 frames to account for errors of the facial feature detector. We also use a smartphone to record a facial performance of the actor, but leverage an iPhone's video and depth sensor and Apple's ARKit to capture blendshape weights and 3D geometry for each frame of this performance in a robust and fully automatic manner. Since our capturing process is easy and intuitive, we currently let the actor mimic all 52 ARKit blendshapes to get a maximum of personalization.

Han et al. [Han+21] evaluated two methods to extract personalized blendshapes from expression scans. They recorded expressions and blendshape weights and fitted an autoencoder and a linear regression to obtain the blendshapes. However, while they achieve good results with respect to the reconstruction error, they do not compute semantically meaningful blendshapes, which disqualifies their approach for many applications. Li et al. [LWP10] and Seol et al. [SML16] employ optimizations based on deformation gradients to extract personalized blendshapes from a set of training examples. Li et al. [LWP10] fit the deformation gradients of the unknown blendshapes to the deformation gradients of the example expressions, regularized by (the deformation gradients of) the blendshapes of a generic rig to ensure semantically meaningful blendshapes. Blendshape weights and blendshape meshes are solved for in an alternating optimization, requiring careful initialization for convergence (see Section 8.3). Seol et al. [SML16] first fit a mesh to the scanned

expressions and then separate the fitted expressions into the different blendshapes based on the vertex displacements in the generic rig. In our experiments, their blendshape separation method leads to strongly damped blendshapes, though. Our approach leverages ARKit's face tracking and therefore avoids the alternating optimization for blendshape weights and blendshape meshes. As shown in Section 10.2, our fitting also leads to considerably more accurate results compared to [LWP10].

Ming et al. [Min+25] compute a personalized blendshape rig from sparse multi-view videos. They combine geometric losses and neural inverse rendering to jointly optimize the facial rig, per-frame expression weights, and a neural rendering MLP for realistic appearance. The blendshapes are initialized from the ICT face model [Li+20b], effectively ensuring semantic consistency and compatibility with standard animation pipelines. However, they do rely on calibrated multi-view videos, which require lab setups.

Several approaches further increase the reconstruction accuracy by using corrective blendshapes or corrective deformation fields. The facial animation is then computed by the initial (generic) blendshape set, but it is refined with additional shapes, which add idiosyncrasies that cannot be represented by the initial blendshapes [BWP13; Cha+20; Gar+16; IBP15; Li+13].

Other approaches extend geometric blendshape rigs with additional textures [Cas+16; CKU23; Li+20a]. Casas et al. [Cas+16] generate additional expression blendshape textures, which are blended during animation. This way, individual features (e.g., facial wrinkles) that are not contained in the blendshape meshes are visualized. However, storing these textures and evaluating them during rendering increases memory consumption and limits scalability. Therefore, Li et al. [Li+20a] and Cai et al. [CKU23] propose to compress the texture into a small number of textures, storing information for multiple blendshapes. These textures are then blended using precomputed weight maps during animation, significantly reducing the size and number of auxiliary textures.

Our work focuses on computing personalized blendshapes without additional corrective shapes or auxiliary textures, to be compatible with standard blendshape pipelines in XR engines, but corrective fields or textures could easily be added in future work. In contrast to many previous works that use a face/head avatar only, we implant the resulting personalized facial blendshapes into a full-body avatar using a modification of deformation transfer [Bot+06; SP04].

Similar to the developments in full-body avatar reconstruction (see Section 4.2), many new approaches focus on the integration of Neural Radiance Fields (NeRF, [Mil+21]) and 3D Gaussian Splatting (3DGS, [Ker+23]) for facial animation [Ane+25; Gaf+21; Gao+22; Ma+24; Ma+25; Qia+24; Teo+24; Xu+24]. Both representations have the advantage of not being restricted to the mesh's topology, unlike standard blendshapes rigs. Therefore, they are better suited for displaying small dynamic details (e.g., facial wrinkles) that cannot be represented with blendshapes. However, these methods require sufficient training views to generate high-quality animations. When the facial

expression is not contained in the training data or the viewpoint differs too much, these methods can produce blurry results or artifacts, which must be handled carefully [Qia+24].

In this thesis, we restrict ourselves to the personalization of standard blendshape rigs as these are compatible with existing animation pipelines. Neural or appearance-based models can improve facial animation; however, they typically replace blendshape rigs and are therefore outside the scope of this work.

9.2 Blendshape Weight Generation for Facial Animation

Several approaches have been used to animate virtual faces, which we roughly divide into either expression tracking or expression synthesis. Murcia-López et al. [Mur+20] let participants select animation parameters for stylistic characters that are included in the Oculus Avatar SDK. The SDK activates blendshapes with FACS-like semantics as follows: lip-sync is created by retrieving phonemes in a temporal convolutional network, eye behavior is based on dynamic saliency-based gaze targeting with blinks about every six seconds or more often during speech and gaze, and ambient micro-expressions are linked to lip sync and eye gaze events. The Oculus Avatar SDK is also widely used in commercial applications, including PokerStars VR, Epic Roller Coasters, and Tribe XR. Other, more data-driven approaches created realistic facial animation from audio but are not readily integrated into game engines common in XR research [Cud+19; Ric+21; Vil+22]. Newer speech-driven facial animation approaches aim to improve the estimation from audio input by using additional reference images to create more user-specific facial expressions [Han+25b; Liu+25]. Others, in contrast, focus on the disentanglement of emotion and content [Han+25a; Pen+23]. The idea is to adjust blendshape weights using emotional information contained in acoustic frequencies, resulting in more personalized facial expressions instead of just creating phoneme-compatible animations.

Tracking facial expressions in XR setups is challenging because large areas of the face are occluded by the HMDs. Before headsets with built-in facial expression tracking became available (e.g., Meta Quest Pro or HTC Vive Pro Eye with Facial Tracker), previous works have deployed custom-built hardware or composed existing sensors, commonly fusing lower face tracking sensors with eye-tracking headsets [Li+15; Ols+16; Rot+19; Vol+22].

Apart from different expression outputs, facial animation methods can differ in processing duration, introducing audiovisual misalignment. Since light travels faster than sound, we are not used to quicker audio and are sensitive to even small synchronization errors: perceptual thresholds have been reported to range from 80 ms [Ste96] to 180 ms [YC08]. Hence, we can synchronize animations with recorded audio to isolate the effect of expression output irrespective of processing duration.

We decided to compare facial expressions synthesized with the Oculus Avatar SDK to facial expressions tracked with ARKit. Both use FACS-like action unit semantics and provide plugins to forward expression data to game engines.

AUTOMATED BLENDSHAPE PERSONALIZATION FOR FAITHFUL FACE ANIMATIONS USING COMMODITY SMARTPHONES

10

Part I introduced our avatar reconstruction system, enabling users to create avatars with simply their smartphones. The system allows for a fast avatar creation, and the resulting avatars achieve similar perception results compared to avatars created with expensive expert-operated scanner setups. However, the system only performs a static personalization, i.e., the avatar is created from a single neutral pose. To make the avatar animatable, we transferred a skeleton rig and a set of generic blendshapes. While a generic skeleton rig works well for body animation, generic blendshapes can produce incongruent facial expressions and social cues [LW22; Rot+16].

In this chapter, we address this limitation by introducing an approach for dynamic facial personalization. We developed (i) an iOS application that leverages Apple's ARKit [App26] to capture example expressions, (ii) a position-based formulation of EBFR based on [LWP10] to create personalized blendshape masks that outperform the blendshapes created with the original method in terms of accuracy, and (iii) a partial deformation transfer method to seamlessly implant the personalized blendshape masks to a full-body avatar.

Since this method was published before the avatar reconstruction approaches in Part I, we used avatars created with the method of Wenninger et al. [Wen+20], which utilize a different template model. However, a master's student I co-supervised adjusted the approach from this chapter, making it compatible with our new template model.

Individual Contribution *I implemented the automated blendshape personalization approach, presented in this chapter. The process was supervised by Mario Botsch and Marc Erich Latoschik. The main contributions in this chapter are:*

- *an iOS application guiding the user to record example facial expressions,*
- *a position-based formulation of EBFR based on [LWP10] to extract personalized blendshapes from the recorded expression,*
- *a seamless partial deformation transfer to transfer personalized blendshape masks to a full-body avatar.*

Corresponding Publication *This chapter is based on the following publication:*

> T. Menzel, M. Botsch, and M. E. Latoschik. "Automated Blendshape Personalization for Faithful Face Animations Using Commodity Smartphones". In: *Proceedings of the 28th ACM Symposium on Virtual Reality Software and Technology*. 2022. DOI: 10.1145/3562939.3565622

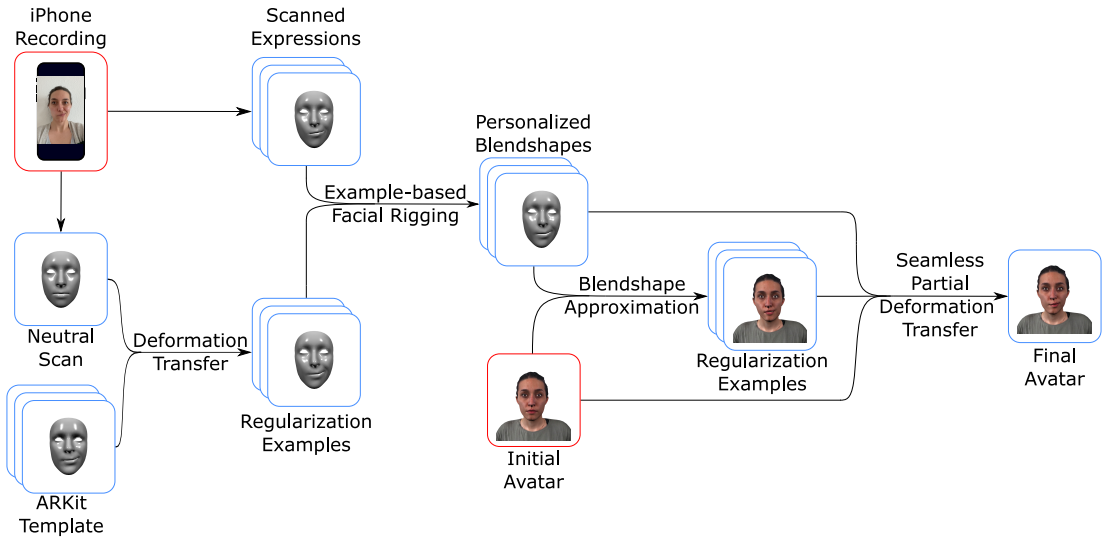


Figure 10.1: Our pipeline from recording example expressions to the final avatar. Red boxes indicate user input.

10.1 Method

In this section, we present our approach for extracting personalized blendshapes from a set of scanned facial expressions and how to transfer these blendshapes to an existing avatar. Figure 10.1 shows an overview of our pipeline. We begin by specifying the capturing process and the resulting input data (Section 10.1.1). From this training data, we extract personalized blendshapes (Section 10.1.2), which are then transferred to the full-body avatar using our seamless partial deformation transfer (Section 10.1.3).

10.1.1 Input Data

Our goal is to create personalized blendshapes for an existing full-body avatar. While any method could be used to generate that avatar, we employ the smartphone-based method of Wenninger et al. [Wen+20]. As a consequence, the complete avatar generation and personalization only requires a smartphone for capturing the person, which is in stark contrast to recent approaches based on complex photogrammetry rigs [Ach+17; FRS17] and makes the reconstruction of personalized avatars more widely available.

We capture the training data for the blendshape personalization using a custom application running on an iPhone 12 Pro. Although any recent ARKit-capable iOS device could be used as well, we found the depth measurements of the iPhone 12 Pro to be more accurate. Thus, we use this device to record our training data. The application guides the user through the recording session, making the whole capturing process easy and intuitive. All recording is performed by the front-facing depth and color cameras, such that the user can watch instructions and receive feedback while recording.

The user first scans their own face in a neutral expression, and is then asked to perform the facial expressions corresponding to the $m_B = 52$ ARKit blendshapes. For each of these blendshapes, the application shows a textual and pictorial explanation of the expression to be performed. The application continuously tracks the user's face using the ARKit framework and automatically captures the facial expression when the requested expression is performed to a sufficient extent. In particular, when capturing blendshape i (for $i \in \{1, \dots, m_B\}$), we observe the blendshape weight $w_{i,i}$ corresponding to that blendshape. Once this weight exceeds a certain threshold and reaches a maximum over time (i.e., starts decreasing), we record both the current set of blendshape weights $(w_{i,1}, \dots, w_{i,52})$ and the current geometry $\boldsymbol{S}_i$ of the ARKit face mesh, where the latter consists of $N_A = 1220$ vertices and 2304 triangles. While we expect the blendshape weight $w_{i,i}$ to be dominant when capturing blendshape i, other non-vanishing blendshape weights $w_{i,j}$ do not pose a problem for our reconstruction (see next section), such that the user does not have to strictly perform the requested expression in isolation (which is hardly possible).

With this automated recording procedure, the 52 requested facial expressions can be scanned in approximately two minutes.

10.1.2 Blendshape Personalization

While ARKit provides us with blendshape weights $(w_{i,1}, \dots, w_{i,m_B})$ and face meshes $\boldsymbol{S}_i$ for each expression $i \in \{1, \dots, m_B\}$ of the $m_B = 52$ ARKit blendshapes, it does not give access to the internal personalized blendshapes of the user. We therefore have to extract the personalized version of the ARKit blendshapes using a modification of EBFR [LWP10].

The original method of Li et al. [LWP10] fits the per-triangle deformation gradients of the unknown blendshapes to the deformation gradients of the scanned expressions in a first step, and then solves a linear least-squares system to extract the vertex positions best-matching the fitted deformation gradients [Bot+06; SP04] (see Chapter 8 for details). This two-step procedure has the disadvantage that it does not directly penalize the deviation of the reconstructed vertex positions from the scanned expressions; it only *indirectly* encourages the vertex positions to match the target expressions. In contrast, we *directly* optimize for vertex positions that best-match the observed expressions $\boldsymbol{S}_i$ in the least-squares sense (see Equation (10.1) below), which – as we will show in Section 10.2.1 – leads to more accurate results.

Given the blendshape weights $(w_{i,1}, \dots, w_{i,m_B})$ and face meshes $\boldsymbol{S}_i$ for the recorded expressions $i \in \{1, \dots, m_B\}$, as well as the neutral face scan $\boldsymbol{B}_0$, we have to compute the corresponding personalized delta-blendshapes $\boldsymbol{B}_1, \dots, \boldsymbol{B}_{m_B}$, following the formulation introduced in Section 8.1. Our model then produces a facial expression from blendshape weights $w_1, \dots, w_{m_B}$ as

$$\boldsymbol{B}_0 + \sum_{j=1}^{m_B} w_j \boldsymbol{B}_j.$$

Here, the matrix $\boldsymbol{B}_0 \in \mathbb{R}^{N_A \times 3}$ contains the $N_A = 1220$ vertex positions of the neutral face and the matrices $\boldsymbol{B}_1, \ldots, \boldsymbol{B}_{m_B} \in \mathbb{R}^{N_A \times 3}$ contain the N_A displacement vectors of the blendshapes, respectively.

We compute the blendshapes $\boldsymbol{B}_j$ by penalizing the distance from the observed training expressions $\boldsymbol{S}_i$, formulated as the cost function

$$L_{\text{fit}}(\boldsymbol{B}_1, \ldots, \boldsymbol{B}_{m_B}) = \sum_{i=1}^{m_B} \left\| \boldsymbol{S}_i - \left(\boldsymbol{B}_0 + \sum_{j=1}^{m_B} w_{i,j} \boldsymbol{B}_j \right) \right\|^2. \tag{10.1}$$

Since the blendshape weights $w_{i,j}$ were captured by ARKit and hence are known, minimizing Equation (10.1) only requires solving a sparse linear least-squares system.

However, in order to produce semantically meaningful blendshapes, the optimization must be regularized. To this end, we transfer the blendshapes of the (non-personalized) ARKit template to the recorded neutral expression $\boldsymbol{B}_0$ using deformation transfer [SP04], resulting in the generic blendshapes $\boldsymbol{T}_1, \ldots, \boldsymbol{T}_{m_B}$. Similar to Saito [Sai13], we add virtual triangles between upper and lower eyelids to ensure that the eyes are completely closed in the transferred eye-blink blendshapes. Our regularization energy then penalizes the deviation of the personalized blendshapes $\boldsymbol{B}_j$ from the generic blendshapes $\boldsymbol{T}_j$:

$$L_{\text{reg}}(\boldsymbol{B}_1, \ldots, \boldsymbol{B}_{m_B}) = \sum_{j=1}^{m_B} \left\| \boldsymbol{\Omega}_j \left(\boldsymbol{B}_j - \boldsymbol{T}_j \right) \right\|^2. \tag{10.2}$$

Here, $\boldsymbol{\Omega}_j$ are diagonal $(N_A \times N_A)$ matrices containing per-vertex regularization weights. These weights ensure that vertices that do not move in the template blendshape $\boldsymbol{T}_j$ also do not move in the personalized blendshape $\boldsymbol{B}_j$. They are computed as

$$\left(\boldsymbol{\Omega}_j\right)_{i,i} = \frac{\max_{k=1,\ldots,N_A} \left\| \boldsymbol{t}_{k,j} - \boldsymbol{t}_{k,0} \right\|}{\left\| \boldsymbol{t}_{i,j} - \boldsymbol{t}_{i,0} \right\|},$$

where $\boldsymbol{t}_{i,j} \in \mathbb{R}^3$ is the position of the i-th vertex in the template blendshape $\boldsymbol{T}_j$. This results in higher regularization weights for vertices with smaller displacement magnitude in the ARKit template blendshapes. We clamp the regularization weight $(\boldsymbol{\Omega}_j)_{i,i}$ to 10^5 if $\left\| \boldsymbol{t}_{i,j} - \boldsymbol{t}_{i,0} \right\| < \epsilon$ to avoid division by zero.

The personalized blendshapes are finally computed by minimizing the cost function

$$L_{\text{bs}}(\boldsymbol{B}_1, \ldots, \boldsymbol{B}_{m_B}) = L_{\text{fit}}(\boldsymbol{B}_1, \ldots, \boldsymbol{B}_{m_B}) + L_{\text{reg}}(\boldsymbol{B}_1, \ldots, \boldsymbol{B}_{m_B})$$

that combines the fitting term (Equation (10.1)) and the regularization term (Equation (10.2)), which again only involves solving a least-squares linear system.

Li et al. [LWP10] stated that an optimization based on vertex positions leads to visible artifacts if done naively. However, our results demonstrate

quantitatively and qualitatively that this is not the case for our regularization. In fact, optimizing vertex positions leads to more accurate results, as we show in Section 10.2. Without our regularization, though, we would get clearly noticeable artifacts.

10.1.3 Blendshape Transfer

Having computed the personalized ARKit blendshapes (grey face masks in Figure 10.1), we now transfer them to the provided full-body avatar using an extension of deformation transfer [SP04]. Note that our personalized ARKit blendshapes $\boldsymbol{B}_j$ only specify the deformation within the face area. However, this is only a part of the deformation due to face animation, since some blendshapes (e.g., jaw open) also affect the area adjacent to the face (e.g., the neck area). Therefore, we have to adjust the adjacent area accordingly.

In a first step, we approximate the personalized ARKit blendshapes with the pre-existing non-personalized blendshapes of the full-body avatar. To this end, we define a correspondence map $\boldsymbol{K}$ between the avatar's face and the ARKit face mesh. This is achieved by fitting the avatar template to the ARKit template using non-rigid registration [Ach+17] and selecting the closest point on the ARKit mesh for each vertex of the avatar's face. Since each full-body avatar shares the topology of the full-body template model (from [Wen+20] in our case), this mapping has to be computed only once.

To obtain the optimal blendshape weights for the approximation, we use an approach similar to Lewis et al. [LA10]. First, we use the iterative closest point algorithm (ICP) with scaling [ZSN05] to find the optimal translation, rotation, and scaling to register the ARKit face mesh to the avatar's face. Second, we compute the weights of the initial avatar blendshapes by minimizing the energy

$$L^j_{\text{approx}}(\hat{w}_1, \ldots, \hat{w}_{m_T}) = \left\| \boldsymbol{K}\boldsymbol{B}_j - \sum_{i=1}^{m_T} \hat{w}_i \hat{\boldsymbol{B}}_i \right\|^2 + \tau \sum_{i=1}^{m_T} \hat{w}_i^2, \tag{10.3}$$

where $\boldsymbol{K}$ is the pre-computed correspondence matrix that maps vertices of the ARKit face meshes to the full-body avatar. $\hat{\boldsymbol{B}}_i$ denote the initial avatar's blendshapes and $\hat{w}_i$ are the (unknown) blendshape weights. The second term penalizes large weights to avoid extreme poses. Solving a linear least-squares system results in the blendshape weights $\hat{w}_1, \ldots, \hat{w}_{m_T}$ for approximating the ARKit blendshape $\boldsymbol{B}_j$ using the initial avatar blendshapes $\hat{\boldsymbol{B}}_1, \ldots, \hat{\boldsymbol{B}}_{m_T}$.

The resulting approximation, which can be considered an *automatic facial retargeting* and is denoted by $\boldsymbol{Z}_j$, is already quite close to the desired personalized blendshape $\boldsymbol{B}_j$ (see Figure 10.2). Using the initial avatar's blendshapes, it provides the missing information on how to deform the avatar's face in the area not covered by the ARKit face mask (e.g., the neck area under the chin). We therefore use the approximation $\boldsymbol{Z}_j$ as regularization when transferring the personalized blendshape $\boldsymbol{B}_j$ to the avatar.

This transfer proceeds in two steps (see Figure 10.3): First, we apply deformation transfer to only the avatar's face using the pre-computed corre-



Figure 10.2: The personalized blendshape (left) and our approximation with generic blendshapes (right).

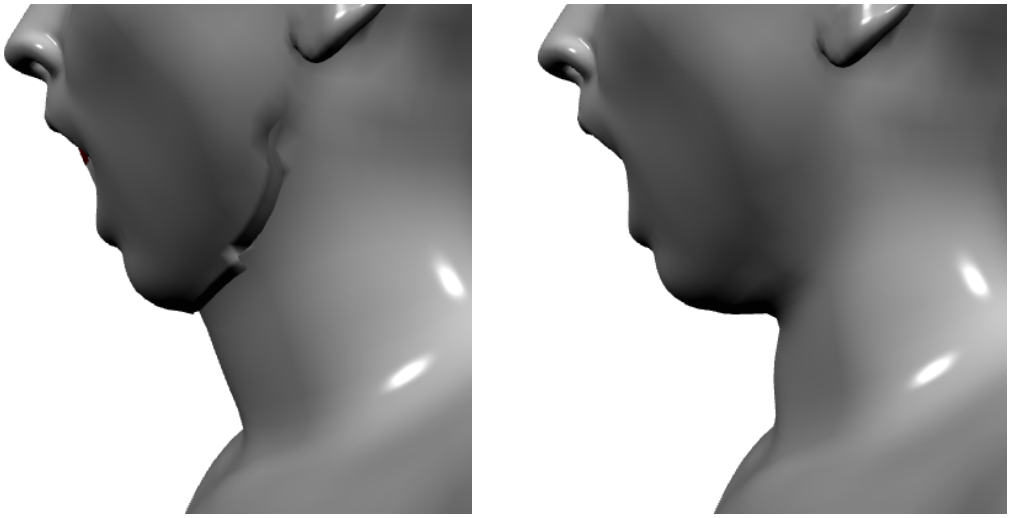


Figure 10.3: Seamless partial deformation transfer of the jaw open blendshape: The first step transfers the deformation of the face region only (left), the second step adjusts the adjacent neck region (right).

spondence mapping. Second, the vertices in the face area are fixed, and the vertices in the adjacent area (vertices that move in $\boldsymbol{Z}_j$ but do not belong to the face area) are non-rigidly deformed from $\boldsymbol{Z}_j$ [Ach+17]. This process seamlessly implants the personalized ARKit blendshapes to the target avatar.

The approximations $\boldsymbol{Z}_j$ also bring facial details not included in the ARKit mesh (eyeballs and teeth) to their desired position in the j-th personalized

avatar blendshape. However, since the ARKit mesh does not include eyeballs, the eyelid blendshapes might intersect them. We eventually repair those artifacts by moving the eyelid vertices to their closest point on the eyeball surface [AZB15].

10.2 Results

In the following, we present quantitative and qualitative comparisons between our blendshape personalization approach (Section 10.1.2), *EBFR* [LWP10], and deformation transfer (*DT*, [SP04]). After that, we show comparisons of our personalized blendshapes to automatic facial retargeting and to manual facial retargeting. *Automatic facial retargeting* refers to the optimization of Equation (10.3). The resulting blendshape weights are the best-fitting combinations of the avatar's initial blendshapes to approximate the personalized ARKit blendshapes. *Manual facial retargeting* refers to a manually optimized mapping of the ARKit blendshape set to the avatar's initial blendshape set. This mapping was hand-crafted by two PhD students in Computer Graphics with sufficient expertise in facial blendshape animation (although not being blendshape artists) and represents our best (manual) effort for retargeting the ARKit blendshapes to the initial avatar's blendshapes.

10.2.1 Face Mask Personalization

In order to evaluate the accuracy of the different sets of personalized ARKit blendshapes, we asked our subjects to record not only the 52 training face expressions (see Section 10.1.1), but also 20–30 additional test expressions, for which we record the ARKit face mesh $\boldsymbol{Y}_i$ and the corresponding blendshape weights $w_1, \ldots, w_{m_B}$. Using these weights and the blendshapes $\boldsymbol{B}_1, \ldots, \boldsymbol{B}_{m_B}$ to be evaluated, we compute the root mean square error (RMSE) over the N_A vertices w.r.t. $\boldsymbol{Y}_i$ as

$$\sqrt{\frac{1}{N_A} \left\| \boldsymbol{Y}_i - \left(\boldsymbol{B}_0 + \sum_{j=1}^{m_B} w_j \boldsymbol{B}_j \right) \right\|^2}. \tag{10.4}$$

Our implementation of EBFR only performs the blendshape optimization step from Li et al. [LWP10], since we already know the correct blendshape weights from ARKit. Figure 10.4 compares the RMSE (Equation (10.4)) of different sets of personalized blendshapes produced with DT (blue), EBFR (orange), and our approach (green) for Subject 1. In almost all cases (except expression 19), our method yields the lowest errors. When averaging the RMSEs over all expressions and all subjects, the RMSE of our method is 58% of the RMSE of EBFR and 45% of the RMSE of DT (see Appendix C).

We also evaluate and compare the maximum error computed over all N_A vertices, since this measure allows to compare the worst parts of the

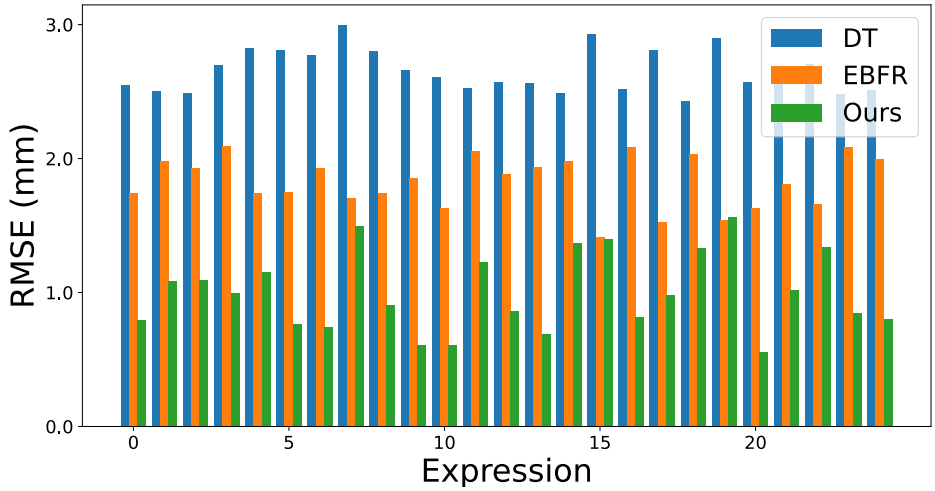


Figure 10.4: Root-mean-square reconstruction error of deformation transfer (DT, blue), example-based facial rigging (EBFR, orange), and our method (green) for Subject 1. Our method produces the lowest RMSE in almost all cases; EBFR's error falls between ours and DT's, whose error is consistently the highest.

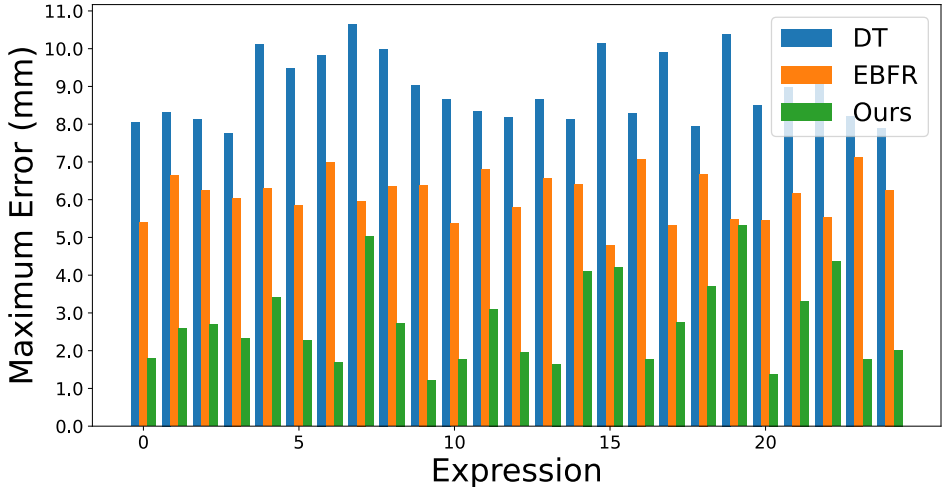


Figure 10.5: Maximum reconstruction error of deformation transfer (DT, blue), example-based facial rigging (EBFR, orange), and our method (green) for Subject 1. Our method produces the lowest maximum error in all cases; EBFR's error falls between ours and DT's, whose error is consistently the highest.

reconstructed expressions. If only a particular region of the face moves in a test expression, the RMSE would be artificially reduced due to averaging over mostly unchanged vertex positions. Figure 10.5 shows the maximum reconstruction errors of the different approaches for Subject 1. It can be seen that DT gives the worst results in all cases. Both EBFR and our method show significant improvements, with our method consistently yielding the lowest error of all three methods. Averaging the maximum error per expression over all test expressions and all subjects, our method reduces the maximum reconstruction error of EBFR and DT down to 50% and 38%, respectively. The color-coding of reconstruction errors in Figure 10.6 visualizes how the three reconstruction methods differ and shows that our reconstructions are the most accurate. Results for other test subjects are shown in Appendix C.

10.2.2 Avatar Blendshape Personalization

The previous section evaluated different approaches for personalizing the ARKit face mask blendshapes. In this section, we evaluate the final avatars produced by implanting the personalized ARKit blendshapes through our seamless partial deformation transfer.

Figure 10.7 compares the approximations $\boldsymbol{Z}_j$ (computed through automatic facial retargeting using the initial avatar blendshapes) to our final avatars with personalized blendshapes. While the approximation $\boldsymbol{Z}_j$ give reasonable results, the expressions are more accurately reproduced by our personalized blendshapes. This is most noticeable in the mouth region, where the retargeted version does not move the mouth corners far enough or cannot properly reproduce the lip shape of the target expression.

The automatic regargeting is computed by minimizing the approximation error (Equation (10.3)) w.r.t. the blendshape weights. Upon inspection, the resulting (optimal) weights exhibit many non-zero weights, with several weights even exceeding the range $[0, 1]$. This explains why a manual retargeting is rather unlikely to reproduce the optimal results. Still, the manual mapping is the default method for connecting face rigs and face tracking in game engines (see, e.g., the Unity Live Capture Plugin or the Faceware Live Client for Unity).

Figure 10.8 compares how well different approaches can reproduce some test expressions that have been tracked through ARKit. The manual retargeting using the initial avatar blendshapes yields the worst results, with clearly visible deviations in the mouth region. Automatic retargeting produces more accurate expressions, but the most accurate results are obtained with the personalized blendshapes. In the top row, the automatic retargeting does not move the mouth corners far enough to the side, visualized through the blue cross and the green line. In the bottom row, the manual and automatic retargeting do not properly close the mouth and eyes, respectively, while the personalized blendshapes do. Comparisons on the full test sequences can be seen in the accompanying video of the paper publication[1].

1 https://doi.org/10.1145/3562939.3565622

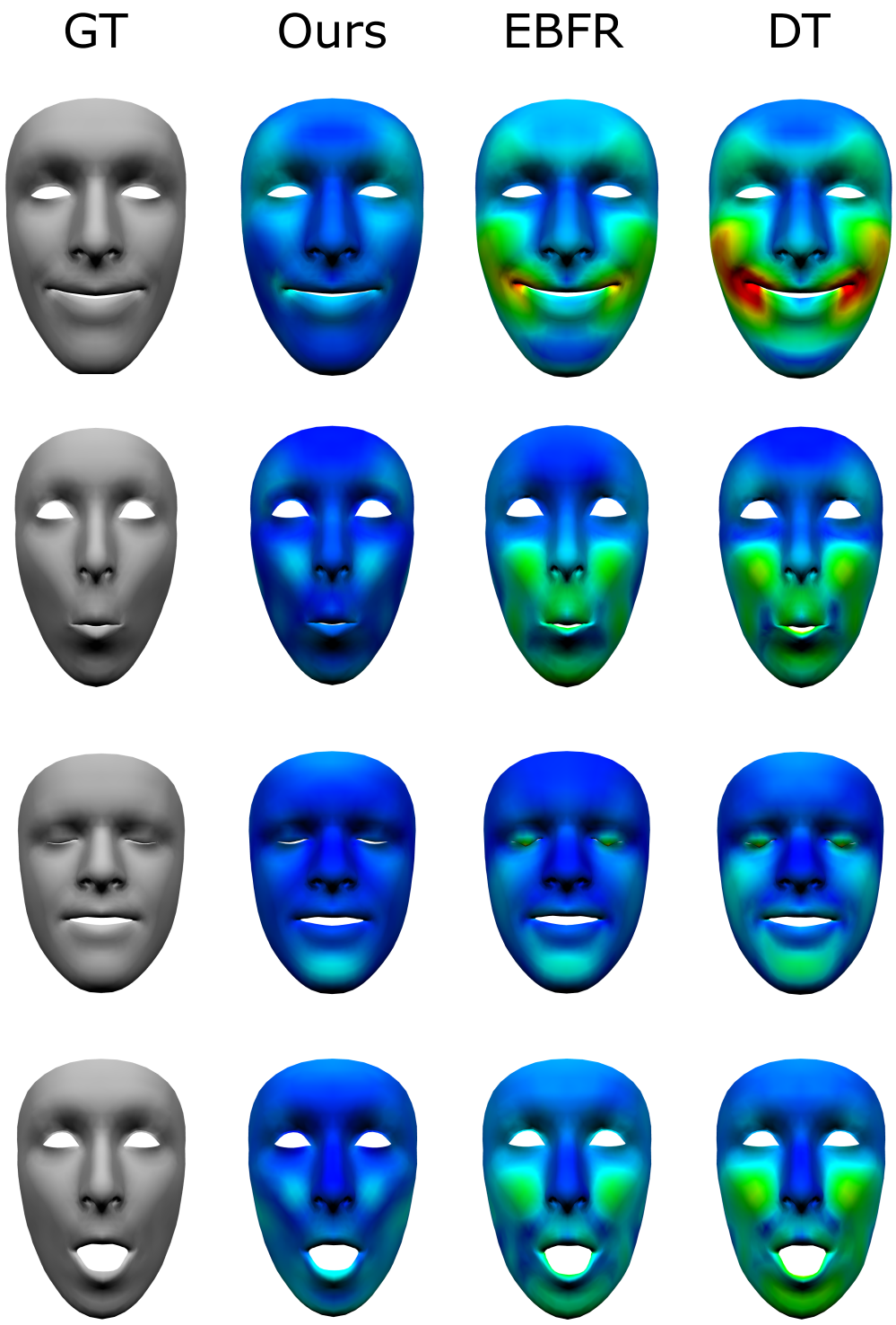


Figure 10.6: Color-coded reconstruction errors. From left to right: ground truth (GT), our method, example-based facial rigging (EBFR), and deformation transfer (DT) (blue = 0mm, red > 5mm). Our method produces the most accurate reconstructions.

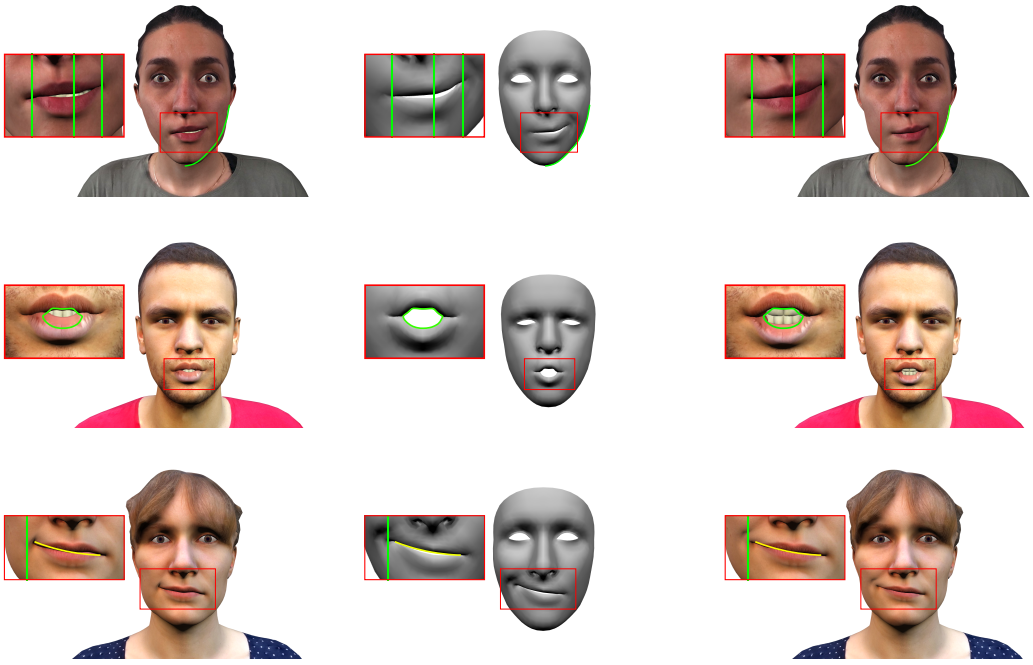


Figure 10.7: Comparison of the personalized ARKit blendshapes (middle) to an approximation using automatic retargeting using the initial avatar blendshapes (left) and to our final personalized blendshapes (right), where the latter yield more accurate results.

10.2.3 Computation Time

Our automatic, app-guided recording of training expressions takes about 2 minutes. Afterward, all computations of our pipeline take approximately 35 seconds per avatar. The ARKit face masks consist of 1220 vertices and 2304 triangles; the avatar meshes consist of 21k vertices and 42k triangles.

Table 10.1 shows the computation times for EBFR and our method, measured on a desktop PC with a 10-core 3.6 GHz CPU and an NVIDIA RTX 3070 GPU. These timings include setting up the linear systems and solving them through sparse Cholesky factorizations. For EBFR, it also includes converting the local frames back to the new blendshape basis, and, for our method, it includes the computation of the regularization blendshapes using deformation transfer. On average, our method is about 3× faster than EBFR. This is mainly due to the fact that EBFR performs the optimization per triangle, and hence has to solve 2304 linear least-squares problems.

10.3 Discussion

As described in Section 10.2.1, personalized blendshapes produce more accurate reconstructions of facial expressions than generic blendshapes transferred from a template rig. Our improved approach further reduces the maximum

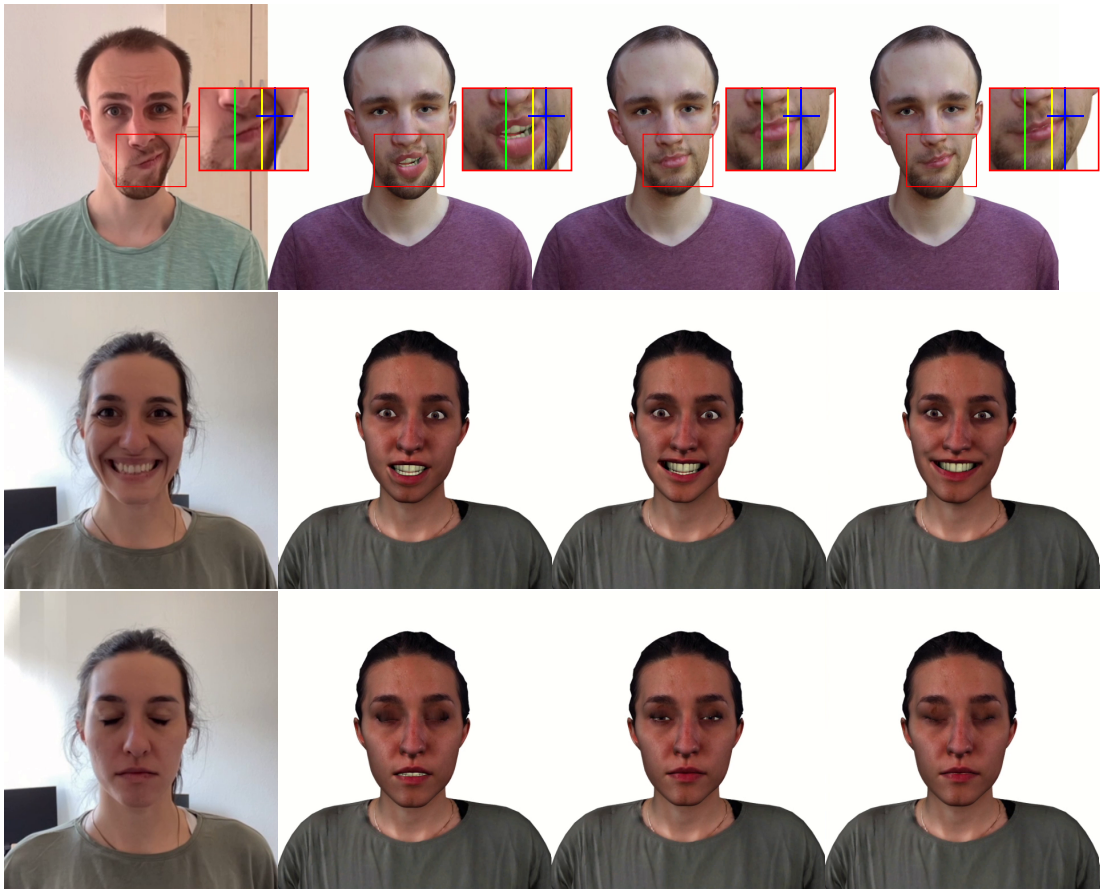


Figure 10.8: Comparison of the original tracked facial expression with the animated avatars. From left to right: captured image, manual retargeting, automatic retargeting, and personalized blendshapes. The most accurate results are obtained with personalized blendshapes, followed by automatic retargeting, while manual retargeting shows clearly visible deviations in the mouth region.

Method	Subject 1	Subject 2	Subject 3	Subject 4	Avg.
EBFR	1.585 s	1.594 s.	1.602 s	1.594 s	1.594 s
Ours	0.500 s	0.499 s	0.499 s	0.497 s.	**0.499 s**

Table 10.1: Computation times of example-based facial rigging (EBFR) and our method, including setting up and solving all required linear systems. On average, our method is about 3× faster than EBFR.

reconstruction error compared to deformation transfer and example-based facial rigging. Considering Figure 10.6, the mouth corners of Subject 1 (top row) are closer to the ground truth for our method, while both DT and EBFR lead to a stronger grin expression. Even these slight differences can be problematic, since they might lead to incongruent social cues [Rot+16] or create unnatural-appearing facial expressions that do not correctly convey a tracked person's actual look and feelings (Figure 10.8).

Our approach also has some limitations. Both the blendshape approximation and the seamless partial deformation transfer rely on correspondence mappings, which have to be computed in a pre-process step once per template model. Moreover, our seamless partial deformation transfer relies on reasonable initial blendshapes of the avatar, which are used to estimate how the adjacent area of the face (e.g., the neck) deforms during facial expressions. As a consequence, our method cannot be applied to avatars without blendshapes.

10.4 Conclusion

Faithful digital reconstruction of humans has various interesting use cases. It may become even more prominent in future social XR encounters. Here, an accurate reconstruction of facial expressions is a necessity due to the prominent role of facial expressions in non-verbal behavior and social interaction. In this chapter, we presented an optimization-based approach to generating personalized blendshapes necessary for a faithful reconstruction of facial expressions and their animation. The proposed method combines a position-based optimization with a seamless partial deformation transfer. It outperforms existing solutions and overall results in a much lower reconstruction error. It also neatly integrates with recent smartphone-based reconstruction pipelines for mesh generation and automated rigging, further paving the way to a widespread application of personalized avatars and agents in various use cases.

In the future, we would like to use the iPhone's front-facing depth sensor to capture more accurate geometry during expression scanning. Furthermore, we would like to investigate how our improved personalized blendshapes influence the perceptibility of expression semantics.

11 AN EVALUATION OF OTHER-AVATAR FACIAL ANIMATION METHODS FOR SOCIAL VR



Figure 11.1: Virtual character addresses observer. Screenshots show, left to right, static face, synthesized expression, and tracked expression. In our study, animation was either in sync with audio or delayed by the animation system's inherent latency.

In the last chapter, we explored a method to create personalized facial blendshapes for our smartphone-based avatars. These blendshapes allow us to create more realistic and faithful facial expressions, which better convey the desired emotion during facial animation. This is especially helpful in multi-user scenarios, where interaction should feel natural and plausible. However, in VR, the main use case for our avatars, facial animation poses an additional challenge beyond creating personalized blendshapes. Most VR devices occlude large parts of the face, making it difficult to track facial expressions. While some devices support facial tracking (e.g., Meta Quest Pro or HTC Vive Pro Eye with Facial Tracker), faithful facial animation is required regardless of the device for natural interaction. To address this, methods have been developed to generate blendshape weights from audio input, enabling facial animation even when tracking is impossible.

In this chapter, we compare these different blendshape weight generation techniques to validate the usefulness of audio input. Specifically, we use Meta's Oculus Avatar SDK [Met23] to generate blendshape weights from audio input and ARKit [App26] to visually track a face animation performed by an actor. Users then evaluate the naturalness and plausibility of the face animation on the actor's avatar (audio-generated weights, tracked, or static) in a VR scenario.

Individual Contribution *My main contributions include adjusting and integrating the iOS facial expression capture application and the blendshape personalization pipeline, developing an iOS application for recording facial animation videos, and integrating the recorded animations into the Unity application used for the user study. Furthermore, I developed the mapping between the Oculus Avatar SDK and ARKit blendshape sets by fitting our avatar template to the Oculus Avatar, establishing correspondences between the face topologies, solving an (face) inverse kinematics problem*

to reproduce expressions on our template, and implementing a Unity script to drive avatar animation with the mapped blendshape weights.

The user study was designed, conducted, and evaluated by Peter Kullmann.

Corresponding Publication *This chapter is based on the following publication:*

> P. Kullmann, T. Menzel, M. Botsch, and M. E. Latoschik. "An Evaluation of Other-Avatar Facial Animation Methods for Social VR". In: *Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems*. 2023. DOI: 10.1145/3544549.3585617

11.1 Method

Our study followed a 2×3 design with between-groups factor *latency adjustment* (animations in sync with audio vs. delayed by the latency inherent to the animation system) and within-groups factor *facial animation method* (face tracking vs. expression synthesis vs. static face). Written approval for the study was obtained from the ethics committee of the Institute Human-Computer-Media (MCM) of the University of Würzburg[1].

We hypothesize two effects:

1. *Main effect of facial animation method*: static faces are perceived as less natural and less plausible than animated (synthesized and tracked) faces.

2. *Interaction effect of latency adjustment x facial animation method*: In original latency, the quicker method is perceived as more natural and plausible than the slower method. When latency is adjusted for, both synthesized and tracked facial expressions are rated equally natural and plausible.

11.1.1 Character Creation

We based our virtual human on a full-body scan using the approach from Achenbach et al. [Ach+17]: In a custom-built rig of 94 DSLR cameras, multi-view images of our scanning subject were captured to generate a dense point cloud. A template model's pose and shape parameters were then optimized to fit this point cloud. In a second step, we personalized blendshapes with our automated blendshape personalization pipeline (see Chapter 10).

Finally, minor scanning artifacts in texture and mesh were corrected manually, resulting in a skinned mesh resembling our actor with high visual detail and ready for real-time animation.

11.1.2 Performance Capture

To initiate unscripted natural behavior, we examined a set of questions intended to be used in a classroom for English lessons [2], similar to the conversa-

1 https://www.mcm.uni-wuerzburg.de/forschung/ethikkommission

2 http://teflpedia.com/Teflpedia:Conversation_questions



Figure 11.2: Performance capture setup. Actor facing a mannequin as target while equipped with head tracker, microphone, and hand controllers, with face tracker on table.

tion prompts from Lee et al. [Lee+19]. We then picked questions we thought to inspire spontaneous, casual answers about non-intimate topics that neither require too much background knowledge nor reveal personal information, such as "What makes someone a good driver?" or "Are holidays really relaxing? What stressful things are involved in taking a holiday?".

We instructed a trained actor to respond freely to our selected questions, as if asked by a recently met acquaintance. Their imagined conversation partner was embodied as a mannequin and sat opposite of him at a table (see Figure 11.2). We set a target of about 30 seconds per response, because longer observation times of expressive behavior have not been shown to predict interpersonal outcomes better [AR92].

To record the performer's behavior, we equipped him with Valve Index Knuckles to track hand motions and an HTC Vive tracker to track head movements. We recorded a video and facial expressions with an iPhone 12 Pro that ran a custom-built app. It used official Apple SDKs to record the phone's screen with camera feed and microphone input, and the detected facial expressions as 52 FACS-like coefficients together with 3D poses for head, both eyes, and the point currently looked at.

For our virtual reconstruction of the scene, we let the actor raise his arms to a T-pose and probed several landmark positions in the room with an additional HTC Vive Controller: the corners of the table, iPhone camera, the mannequin's chin, forehead, and eyes, and the actor's tip of the nose, nasion, corners of the mouth, and wrists.

11.1.3 Animation

This section presents the animation framework utilized in our study. We first describe the alignment of the coordinate systems between ARKit, the Oculus Avatar SDK, and the actor's personalized avatar. We then explain the adjustments of the avatar's body pose and the transfer of the synthesized or tracked facial expressions to the avatar's face, before outlining our steps to synchronize audio and facial expressions.

Coordinate System Alignment

To find the offset between the head tracker origin and the skinned mesh's skull joint, we registered virtual face mesh landmarks with corresponding face landmarks probed in the performance capture setup. This was further refined by manually matching the tracker placement on the virtual character in a juxtaposition of face sensor screen capture and the virtual character rendered in a matching perspective.

To match coordinate frames from ARKit and SteamVR, we first registered matching vertices from ARKit's face mask with vertices from the full-body mesh with corresponding blendshape weights applied. Secondly, we applied the previous offset from the head tracker to the virtual skull joint. Finally, we further adjusted the offset so that the gaze rays hit the iPhone camera at the time of recording when the actor truly looked at it.

Similarly, we offset head poses forwarded to the Oculus Avatar SDK to have the SDK template avatar match the actor avatar's eye level and direction.

Body Pose

To infer the actor's body pose, we used the VRIK solver from RootMotion's Final IK package[3]: hand and head end effector targets followed respective tracker trajectories with a fixed pelvis target at the seat and elbow bend goals at the armrests. We retrieved offsets between trackers and end effectors from the recorded T-pose.

Facial Expression

Data for both tracked and synthesized facial animation was brought into a shared animation parameter space to drive the virtual actor mesh (see Figure 11.3). We transferred ARKit expressions directly since the mesh had exactly matching blendshapes. Mapping expressions from the Oculus Avatar SDK directly to their semantically matching ARKit blendshapes resulted in different expressiveness. To get more realistic facial expressions, we retargeted the Oculus Avatar SDK template avatar to the virtual actor avatar by computing a per-expression mapping from the Oculus blendshape basis to the personalized ARKit blendshapes (see Algorithm 11.1). Although the algorithm iterates over the Oculus blendshapes, the retargeting is performed in three conceptual stages, independently for each Oculus blendshape:

3 https://assetstore.unity.com/packages/tools/animation/final-ik-14290

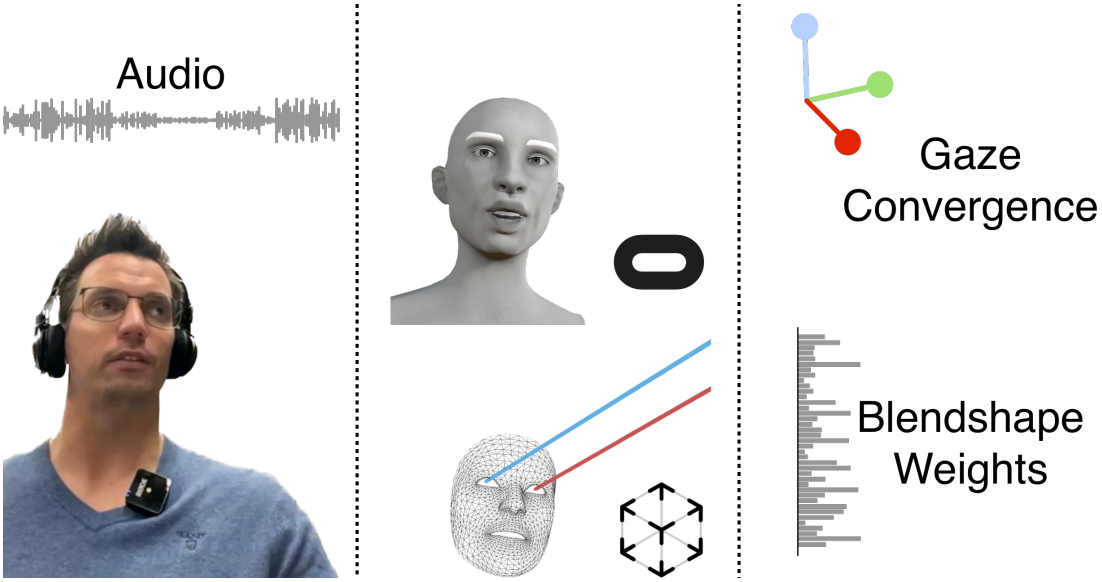


Figure 11.3: Facial Animation Pipeline. Performance (left) is turned into corresponding animation parameters (right) from either audio and tagged gaze targets (Meta Oculus Avatar SDK, top center), or RGBD tracking (Apple ARKit, bottom center). Resulting avatar renders shown in Figure 11.1.

Stage 1: Topology Transfer To resemble the Oculus avatar with the topology of our virtual actor's avatar, we first fit the template mesh from the avatar reconstruction method [Ach+17] to the Oculus avatar using non-rigid registration (see Section 3.2), yielding the shape of the Oculus avatar in the topology of our template mesh (see Algorithm 11.1, Line 1). Since the fitted mesh differs geometrically from the template mesh, we transfer the generic template blendshapes to the fitted model using deformation transfer, obtaining a compatible generic blendshape basis (Line 2). From the fitted model and the Oculus avatar, we then compute a correspondence mapping between both topologies (Line 3). Using this mapping, we transfer the Oculus avatar's blendshape displacements to the fitted model (Line 5).

Stage 2: Oculus-to-Template Expression Mapping In the next stage, we compute the combination of generic template blendshapes (from [Ach+17]) that best approximate the transferred Oculus blendshape deformation on the fitted mesh (Line 6). The optimization is similar to our blendshape weight approximation from the previous personalization pipeline (see Equation (10.3)). This gives us one combination of generic template blendshape weights for each Oculus blendshape.

Stage 3: Oculus-to-ARKit Expression Mapping Since the blendshape sets of our template model and our personalized avatar's model do not share the same blendshape basis (see Section 10.1.3), we repeat the previous step for our personalized avatar. More precisely, for each Oculus blendshape, we first use the found generic blendshape configuration to pose the personalized avatar

Input:

Oculus avatar $\boldsymbol{M}_O$ with blendshapes $\mathcal{B}_O = \{\boldsymbol{B}_O^1, \ldots, \boldsymbol{B}_O^{m_O}\}$
Template mesh $\boldsymbol{M}_T$ with stacked generic blendshapes $\boldsymbol{B}_T \in \mathbb{R}^{3N_T \times m_T}$
Personalized avatar mesh $\boldsymbol{M}_P$ with:
- Transferred stacked generic blendshapes $\boldsymbol{B}_T^P \in \mathbb{R}^{3N_T \times m_T}$
- Stacked personalized blendshapes $\boldsymbol{B}_P \in \mathbb{R}^{3N_T \times m_B}$

Output:

Mapping $\mathcal{H}_{O \to P}$ from Oculus to personalized blendshapes

Procedure:

1: Fit template mesh $\boldsymbol{M}_T$ to Oculus avatar $\boldsymbol{M}_O$ using non-rigid registration, yielding fitted model $\widetilde{\boldsymbol{M}}_T$
2: Transfer $\boldsymbol{B}_T$ to $\widetilde{\boldsymbol{M}}_T$ using deformation transfer, yielding transferred generic blendshapes $\widetilde{\boldsymbol{B}}_T$
3: Compute correspondence mapping $\mathcal{R}$ from $\widetilde{\boldsymbol{M}}_T$ to $\boldsymbol{M}_O$
4: **for** Oculus blendshape $\boldsymbol{B}_O^i \in \mathcal{B}_O$ **do**
5: Transfer $\boldsymbol{B}_O^i$ to $\widetilde{\boldsymbol{M}}_T$ using $\mathcal{R}$, yielding target deformation $\widetilde{\boldsymbol{M}}_T^{B_O^i} - \widetilde{\boldsymbol{M}}_T$
6: Compute template blendshape weights $\boldsymbol{w}_T$ by solving:

$$\min_{\boldsymbol{w}_T} \left\| \left(\widetilde{\boldsymbol{M}}_T^{B_O^i} - \widetilde{\boldsymbol{M}}_T \right) - \widetilde{\boldsymbol{B}}_T \boldsymbol{w}_T \right\|^2 + \tau \| \boldsymbol{w}_T \|^2$$

7: Pose $\boldsymbol{M}_P$ using $\boldsymbol{B}_T^P$ and $\boldsymbol{w}_T$, yielding personalized target $\boldsymbol{M}_P^{B_O^i}$
8: Compute personalized blendshape weights $\boldsymbol{w}_P$ by solving:

$$\min_{\boldsymbol{w}_P} \left\| \left(\boldsymbol{M}_P^{B_O^i} - \boldsymbol{M}_P \right) - \boldsymbol{B}_P \boldsymbol{w}_P \right\|^2 + \tau \| \boldsymbol{w}_P \|^2$$

9: Store the mapping $(i, \boldsymbol{w}_P)$ to $\mathcal{H}_{O \to P}$

Algorithm 11.1: Algorithm for retargeting Oculus blendshapes to personalized ARKit blendshapes. First, we fit our template mesh to the Oculus avatar (Line 1) and compute a correspondence mapping (Line 3). For each Oculus blendshape, we transfer it to the fitted mesh (Line 5) and solve a face inverse kinematics problem to compute the best combination of template blendshapes (Line 6). We then apply the deformation to the personalized avatar (Line 7) and solve a second face inverse kinematics problem to compute the combination of personalized ARKit blendshapes that best approximate the transferred Oculus blendshape (Line 8).

of our virtual actor (Line 7). This posed version of our virtual actor's avatar then serves as the target in the same optimization as in the previous stage (Line 8). Only this time, we compute the best combination of personalized ARKit blendshapes.

Altogether, this results in a direct mapping from the Oculus Avatar SDK's blendshape basis to our personalized ARKit blendshapes. After that, we baked

the computed combinations of personalized ARKit blendshapes as new target shapes that can be driven directly using the Oculus Avatar SDK.

To transfer gaze direction, we rotated the actor avatar's eyes so that their visual axes pointed towards the look-at point. For tracked expressions, we used the reported look-at point; for synthesized expressions, we derived the look-at point as the middle of where the Oculus Avatar SDK's avatar gaze rays are closest.

Latency Adjustment

For synthesized expressions, we pre-calculated phoneme weights for the audio track to synchronize them. The Oculus Avatar SDK used them with an offset of 40 ms. To get them in their original latency, we live-fed the actor audio into the SDK.

To synchronize tracked expressions with the audio track, we offset the animation track by our measured motion-to-photon latency. We recorded a person repeatedly opening their mouth with a bilabial plosive and manually counted the number of frames until their mirrored ARKit mesh opened its mouth. This yielded an offset of 152.5 ms. For original latency, we used the tracked expressions without delay.

The body pose was synchronized to audio by slowly swinging a controller in the face tracker's camera frustum and delaying the audio track so that movement peaks co-occur. Stauffert et al. [SNL20] report a motion-to-photon latency for an HTC Vive tracker of 56.14 ms, which we used as delay for body pose tracking data.

11.1.4 Immersive Observation

Observation in VR occurred seated from the perspective of the mannequin facing the actor during performance capture. Since we focus on the effects of facial animation, we occluded the character's lower body and forearms by placing a table between the observer and the actor in the otherwise empty virtual environment (see the presentation video of the paper publication[4]). We used a Reverb G2 Omnicept as headset (resolution of 2160×2160 pixels per eye at 90 Hz refresh rate).

11.1.5 Procedure

Our study procedure, depicted in Figure 11.4, took about 60 minutes. We welcomed participants and let them read the study briefing. After we answered questions about the procedure, participants gave informed written consent to their participation and the use of their data. Then they filled out digital questionnaires about previous XR and gaming experience, demographics, and symptoms related to simulator-sickness using the Simulator Sickness

4 Presentation video (1:41 min): https://doi.org/10.1145/3544549.3585617

Questionnaire (SSQ, [Ken+93]) on a dedicated workstation. The experimenter performed quasi-random group assignment using covariate-adaptive randomization [JPH21]. Accordingly, participants were evenly assigned to the latency adjustment conditions (all animations either in sync with audio or delayed by the animation system's latency) across biological sex and previous XR and gaming experience. Each participant then observed and rated the virtual character in four different performance blocks.

In each block, the performance was shown in its three facial animation variations, first rated with a static face, then with synthesized and tracked facial expression in randomized order. Thus, every participant rated twelve observations. To conclude, participants reported SSQ scores, familiarity with the shown actor before the experiment, and gave optional study feedback in an open text field.

Before the first trial, the experimenter instructed them on how to don the VR headset, including adjusting lens spacing, strap fit, and a sound test. When ready, participants were instructed to sit relaxed and the virtual camera was calibrated. Once calibrated, the previously set-up black screen overlay was removed to reveal the virtual scene. In it, a text panel informed participants about the virtual character about to be shown and instructed them to observe him attentively. Then, the panel showed the prompt he would respond to. Performances ended by hiding the virtual character and displaying a prompt to remove the headset to continue with the questionnaire on the computer. Here, we investigated *behavior naturalness* by asking for agreement to three statements on a Likert-scale (7 points from "completely disagree" to "completely agree"): 1. The verbal behavior of the virtual character seemed natural, 2. The nonverbal behavior of the virtual character seemed natural, 3. The verbal and nonverbal behavior of the virtual character fit together. To assess *appearance and behavior plausibility*, we asked for agreements to six statements from the same-named dimension of the Virtual Human Plausibility Questionnaire (VHPQ, [Mal+22]) on a Likert-scale (7 points from "completely disagree" to "completely agree"): 1. The behavior of the virtual character seemed plausible, 2. The appearance of the virtual character seemed plausible, 3. The virtual character's behavior matched its appearance, 4. The behavior and appearance of the virtual character were coherent, 5. The virtual character behaved as I would expect it to behave, 6. I could predict how the virtual character would behave by its appearance.

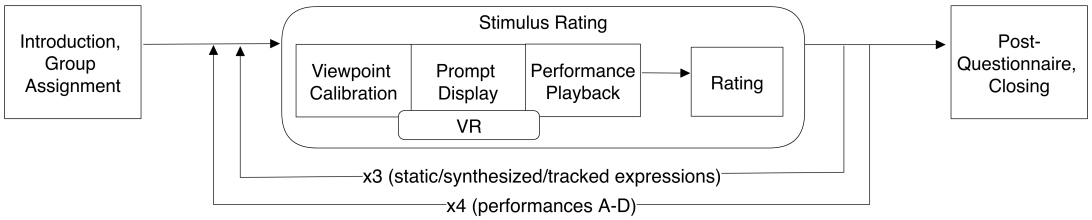


Figure 11.4: Experiment Procedure

11.1.6 Participants

We recruited 48 participants through the University of Würzburg's participation management system. They were free to pick a time slot provided on weekdays during normal working hours and compensated with €10. We excluded four from analysis – two due to technical issues with the setup, one because of language comprehension issues, and one because they reported being familiar with the actor.

The 44 participants we included for analysis (29 of them female) had a mean age of 26.1 years ($SD = 6.1$) and mostly had a higher education entrance qualification (23) or completed studies (19).

11.2 Results

We used R v4.2.2 [R C22] for analysis, aggregated Likert-scale ratings as interval-level data [Nor10], and evaluated effects with linear mixed modeling using nlme [PBR22]. We used random intercepts for all variables and modeled factors and interactions as fixed effects, as proposed by Field et al. [FMF12]. All effects are reported as significant at $p < .05$ using the log-likelihood ratio. Descriptive values are shown in Table 11.1. To test our hypotheses, we used two planned orthogonal contrasts throughout our analysis: first, comparing observation of static faces to observation of either animated faces (tracked or synthesized); second, comparing synthesized to tracked expressions. Since post-exposure SSQ scores were lower than pre-exposure scores in both conditions, we did not analyze simulator sickness further.

11.2.1 Behavior Naturalness

Rating of *nonverbal behavior naturalness* was significantly affected by facial animation method ($\chi^2(2) = 93.24$, $p < .001$), but not by latency adjustment ($\chi^2(1) = .25$, $p = .62$). There was no significant interaction between latency adjustment and facial animation method ($\chi^2(2) = .50$, $p = .78$). Contrasting ratings of static faces with ratings of synthesized and tracked expressions revealed a significant difference ($b = .68$, $t(86) = 11.8$, $p \leq .001$). Likewise, contrasting synthesized with tracked animations proved significant ($b = -.5$, $t(86) = -5$, $p \leq .001$).

Verbal behavior naturalness rating was also significantly affected by facial animation method ($\chi^2(2) = 68.36$, $p < .001$), but not by the method's latency ($\chi^2(1) = .37$, $p = .54$). There was no significant interaction between latency adjustment and facial animation method ($\chi^2(2) = 2.62$, $p = .27$). Contrasting the static face with the two dynamic methods (synthesized and tracked) revealed a significant difference ($b = .65$, $t(86) = 9.74$, $p = .006$). Likewise, the contrast comparing synthesized to tracked animations proved significant ($b = -.5$, $t(86) = -5$, $p \leq .001$).

Match between verbal and nonverbal behavior was significantly affected by facial animation method ($\chi^2(2) = 167.53$, $p < .001$), but not by latency adjustment ($\chi^2(1) = 1.19$, $p = .28$). Ratings show no significant interaction between latency adjustment and facial animation method ($\chi^2(2) = 2.24$, $p = .33$). Contrasting animated with non-animated faces revealed significant differences ($b = 1.01$, $t(86) = 18.9$, $p < .001$), as did contrasting both animated face versions ($b = -.56$, $t(86) = -6.08$, $p < .001$).

11.2.2 Appearance and Behavior Plausibility

Rating of *appearance and behavior plausibility* was significantly affected by the facial animation method ($\chi^2(2) = 104.69$, $p < .001$), but not by the method's latency ($\chi^2(1) = .53$, $p = .47$). There was no significant interaction between latency adjustment and facial animation method ($\chi^2(2) = .34$, $p = .84$).

Contrasting ratings of static faces with the two dynamic methods (synthesized and tracked) revealed a significant difference ($b = .55$, $t(86) = 13.0$, $p \leq .001$). Likewise, the contrast comparing synthesized to tracked animations proved significant ($b = -.36$, $t(86) = -5.45$, $p \leq .001$).

11.2.3 Qualitative Feedback

Several participants mentioned they felt directly addressed by the avatar. A few participants speculated about what changed between observations within a performance block. Some guessed that faces continuously moved more realistically from trial to trial within a block, although we randomized the condition order. Multiple participants mentioned they had difficulty differentiating the two animated conditions.

Latency	Face	Appearance	Behavior Naturalness		
		+ Behavior Plausibility	Verbal	Nonverbal	Match
Original	Static	3.56 (1.26)	4.39 (2.36)	3.26 (1.91)	2.28 (1.45)
	Synthesized	5.67 (0.98)	6.36 (0.85)	5.85 (1.20)	6.14 (0.90)
	Tracked	4.83 (1.28)	5.68 (1.50)	4.72 (1.55)	4.89 (1.60)
Adjusted	Static	3.80 (1.38)	3.76 (2.38)	3.36 (0.50)	2.77 (1.51)
	Synthesized	5.75 (1.00)	6.35 (0.83)	5.88 (1.21)	6.08 (1.20)
	Tracked	5.02 (1.20)	5.73 (1.28)	5.01 (1.47)	5.09 (1.51)

Table 11.1: Descriptive statistics. Means of independent variables with standard deviations in brackets.

11.3 Discussion

Static faces were, on average, rated as less natural and less plausible than animated faces. Notably, this effect was also shown for verbal naturalness. While we had formulated our hypothesis H1 towards the overall difference in naturalness, we highlight this difference because verbal behavior was never manipulated and always equal per performance block.

Against our hypothesis H2, the performance ratings did not reveal an interaction effect between the facial animation method and its synchronization to the speaker audio. This might be due to the relatively small audiovisual skew we used between groups. Also, rating differences in verbal behavior naturalness hint at participants not necessarily focusing on the two channels (verbal/nonverbal) distinctly, while not paying close attention to their temporal alignment.

While our planned contrasts did show significant differences, we expected the comparison between the two animated performances to show inverted differences: All plausibility and naturalness ratings were, on average, higher for synthesized expressions than for tracked ones. In other words, participants found non-personalized, "generic" expressions fit the actor's avatar better than ones from the actor himself. This might stem from artifacts in the tracked facial expressions or the synthesized expressions, including more prosocial cues.

As a first suspicion, we retroactively re-watched the tracked performances with a focus on tracking artifacts. While common in live tracking data, for example, in the form of jittery lips, sudden jumps after tracking loss, and eyes and/or mouth not closing completely, these issues usually do not occur in facial expression synthesis. Synthesized expressions are smoother by design. However, we did not find such artifacts in the captured facial performance.

In a further exploratory analysis, we compared the two animated face variations. In social settings, gaze behavior includes directing gaze at another one's face (face-gaze) or eyes (eye-gaze), simultaneously looking at each other's face (mutual gaze) or eyes (eye contact), and intentionally not looking at another person (gaze avoidance). Since we did not record observer's eye gaze, we looked at how often the virtual human's eyes were directed toward the observer. We calculated how long in total the virtual human's gaze rays hit the mannequin's head. More specifically, we checked whether a sphere moving from the eye origin along the gaze direction hit a capsule collider placed to fit the mannequin head. The tracked expressions included more gaze towards the mannequin, but with shorter dwell times.

We suggest this rating difference in favor of synthesized expressions should be utilized for conversational agents or playing back monologues. Therefore, more natural and more plausible animations might be achieved without the need for facial expression tracking. For truly interactive settings with dynamic turn-taking, our findings might not generalize. Since facial expression tracking also contains personal facial expression dynamics and veridical gaze points,

showing synthesized expressions instead might contribute to misunderstandings. Still, achieving natural and plausible facial animations with tracking input from consumer devices is a valuable insight for XR researchers and developers.

11.4 Limitations and Future Work

We presented one character throughout all observations. This might have rendered the experimental setting less realistic, since people usually do not repeat themselves word-for-word. We opted for this approach because it allowed direct comparisons between the conditions, but suggest also comparing characters that differ in factors like gender, ethnicity, and voice. Similarly, we suggest exploring the effects of facial animation depending on character familiarity.

Also, our approach could be extended to truly interactive setups. However, latency might be difficult to treat as an independent variable, because audio and nonverbal behavior can then only be synchronized by delaying audio signals by the facial animation's processing duration. This amplifies the difference between conversation partners' "non-mutual realities" [RJ01] and results in misunderstandings, for example, in the form of overlapping talk [Seu+21].

Furthermore, future work should also address the effects of nonverbal behavior on the perception of verbal behavior in more detail.

11.5 Conclusion

We explored how two facial animation methods (tracking or synthesis of facial expression) compared to each other and a baseline of a static face when used on a personalized, photorealistic virtual human. In a mixed-design observation study, 44 participants observed four performances, each in its three facial animation variations, and subsequently rated their appearance and behavior plausibility, and behavior naturalness. The between-groups factor of latency adjustment (animation in-sync with audio or delayed by recording latency) showed not to influence ratings significantly. However, the within-factor facial animation method showed to significantly affect ratings: Overall, performances were rated more plausible and more natural when shown with animated faces (synthesized/tracked expressions), even more so for synthesized ones than for tracked ones. This implies that natural, plausible facial animations for avatars do not require facial expression tracking when showing another avatar's monologue. We suggest further work to address these implications for character variations and truly interactive settings.

12 SUMMARY

This thesis presented our progress in the field of affordable and accessible avatar reconstruction and personalization. This chapter summarizes our contributions and the comparisons from the previous chapters, then reviews the limitations, and gives an outlook on future implications.

Summary of Contributions

In the first part of this thesis, we presented an end-to-end avatar reconstruction system that addresses the challenges of reducing hardware requirements while producing high-fidelity full-body avatars. Similar to many current approaches, we leverage the ubiquity of smartphones and their integrated high-quality cameras to significantly reduce the costs for avatar scanners. Our end-to-end reconstruction approach enables even non-expert first-time users to create realistic virtual avatars with just their smartphones. A subsequent study validated the effectiveness and reliability of our method, indicating that our avatars are well-suited for self-representation in VR and, in some cases, even preferred over avatars computed with an expert-operated multi-camera rig.

However, despite the positive results, this avatar reconstruction system was prone to motion-induced artifacts when the scanned person (unintentionally) moves during the scanning procedure. We therefore presented measures to account for these micro-motions. By combining sub-scan fitting, silhouette constraints, and inverse rendering texture generation, we further improved the quality of the resulting avatars. These adaptations significantly reduced the effects of micro-motions in the avatars, making the duration of the scanning procedure less problematic for avatar reconstruction. Finally, we adjusted the computation of the avatar's texture. Instead of relying on the photogrammetry software's texture generation, we use inverse rendering to compute high-quality textures from the sub-scans. This sub-scan-adjusted generation ensures artifact-free textures, such that the reprojection of the final avatar onto the image is almost perfect.

Both of our presented avatar reconstruction approaches use mesh-based representations. Compared to NeRF- and 3DGS-based approaches, which are the subject of considerable current research, our avatars outperform them in novel pose animation, computation time, and inference time. Our avatars are therefore better suited for multi-user VR applications.

As the first part of the thesis dealt with reconstructing avatars from a static neutral pose (A-pose with neutral facial expression), and a set of generic blendshapes was transferred to the resulting avatars, facial expressions can only be reproduced approximately. Even when using motion-tracking techniques to transfer the real expression onto the avatar, the reproduction is subject to

subtle changes, potentially altering the information and emotional response. Therefore, the second part of this thesis introduced an optimization-based approach to generate personalized blendshapes in a fully automatic manner. Our method outperforms existing blendshape personalization pipelines in terms of reconstruction error, thereby ensuring a more faithful reconstruction of facial expressions. The use of a consumer smartphone perfectly complements the idea behind the previous reconstruction approaches, as no expensive scanner is needed for the full-body reconstruction or the blendshape reconstruction.

The recreation of faithful facial expressions not only requires a personalized avatar with personalized expressions, but also a realistic set of blendshape weights. Although the blendshapes are personalized, it is not guaranteed that every combination of blendshapes creates a realistic and valid facial expression. In the second part, we therefore also evaluated different methods to generate blendshape weights for facial animation. We compared audio-driven blendshape weight generation, facial tracking, and static faces for plausibility and naturalness of facial animation. The study revealed that animated faces were perceived as more natural and plausible than static faces. Contrary to the first intuition, animated faces with tracked facial expressions were perceived as less natural and plausible than faces animated with synthesized facial expressions. Although this finding does not generalize to all communication, it highlights the potential of using synthesized facial expressions for conversational agents or other scenarios, where tracking is not possible.

Limitations and Future Work

While the systems and approaches presented in this work demonstrated noticeable improvements over existing methods, some limitations remain that offer opportunities for future work.

Mesh-Based Representation One limitation is that both of our avatar reconstruction approaches (see Part I) use mesh-based representations. Although avatars consisting of meshes are useful for VR applications, because they can be rendered at much higher frame rates than other representations (NeRF/3DGS), they also limit our reconstruction possibilities, as clothing, skin, and hair are all represented in the same surface. Particularly during animation, this reduces the realism of our reconstructions, as physically plausible effects such as gravity are not possible. Recent progress in semantic segmentation (e.g., Meta's Sapiens model [Khi+25]) has significantly improved the quality of segmentation masks for skin, hair, and clothing. The segmentation masks, therefore, provide usable classifications of different parts of the point cloud. These classifications could enable separate fitting of clothing, hair, and skin, allowing separate models to be fit, resulting in more realistic animation possibilities.

Motion Another limitation is motion during the scanning procedure. Although we presented an approach to minimize the impact of motion-induced

errors in photogrammetry-based avatar reconstruction, our improvements apply only to micro-motion during the scanning procedure. Fast motion, that would be visible inside a sub-scan, still produces noise in the resulting point clouds, thus producing motion artifacts in the final avatars, such as incorrect body proportions. This could be improved by using semantic body-part segmentation. With semantic segmentations of rigid body parts, the generated point clouds would contain less noise. These noise-free or at least noise-reduced point clouds could be used to fit the model partially and thus incrementally create a motion artifact-free avatar. Addressing this motion limitation would further improve the accuracy of the avatars, making them overall more plausible.

Lighting Lighting during scanning also poses a problem. None of the proposed avatar reconstruction pipelines models the lighting in the scanning location when computing the final texture; thus, the lighting is baked into the texture. A way to reduce the effect of light occurring in the scene could be to use new normal estimation approaches (e.g., Sapiens normal estimation tool [Khi+25]). These normal estimation tools estimate the surface normal per pixel, and allow modeling of lighting in the scene using spherical harmonics, which model lighting on the spherical harmonics basis functions and a normal input parameter. The key challenge here is the consistency of estimated normals across multiple images, as inconsistent normals for the same point would prevent modeling the light in the scene.

Future Implications

Our proposed approaches enable the rapid reconstruction and personalization of avatars for user representation, regardless of scanning location or equipment. The resulting avatars can be used in a variety of applications, from entertainment (e.g., games) to social metaverse scenarios (e.g., social VR). Since realistic avatars increase users' perceived sense of embodiment and presence, our avatars are tailor-made for this task.

At the same time, realistic avatars raise some important considerations. Our applications create digital twins of individuals. Both the captured data and the resulting avatar can be considered biometric data. In applications where avatars are used as representations of users, this introduces potential risks, such as identity theft and impersonation. Addressing this requires clear usage policies and safety measures for the users.

However, when keeping these risks in mind, this work presented an automated way to create personalized, realistic avatars using only consumer devices, making it accessible to non-expert users. With responsible handling of privacy and identity, the presented methods open up new opportunities for future research and social VR applications.

BIBLIOGRAPHY

[Ach+17] J. Achenbach, T. Waltemate, M. E. Latoschik, and M. Botsch. "Fast generation of realistic virtual humans". In: *Proceedings of the 23rd ACM Symposium on Virtual Reality Software and Technology*. 2017. DOI: 10.1145/3139131.3139154.

[Agi25] Agisoft. *Agisoft Metashape*. 2025. URL: https://www.agisoft.com (visited on 01/04/2026).

[AI21] S. Aseeri and V. Interrante. "The Influence of Avatar Representation on Interpersonal Communication in Virtual Social Environments". In: *IEEE Transactions on Visualization and Computer Graphics* 27.5 (2021), pp. 2608–2617. DOI: 10.1109/TVCG.2021.3067783.

[Ali25] Alice Vision. *Meshroom*. 2025. URL: https://alicevision.org/#meshroom (visited on 01/04/2026).

[All+18a] T. Alldieck, M. Magnor, W. Xu, C. Theobalt, and G. Pons-Moll. "Detailed Human Avatars from Monocular Video". In: *International Conference on 3D Vision (3DV)*. 2018, pp. 98–109. DOI: 10.1109/3DV.2018.00022.

[All+18b] T. Alldieck, M. Magnor, W. Xu, C. Theobalt, and G. Pons-Moll. "Video Based Reconstruction of 3D People Models". In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018, pp. 8387–8397. DOI: 10.1109/CVPR.2018.00875.

[All+19] T. Alldieck, M. Magnor, B. L. Bhatnagar, C. Theobalt, and G. Pons-Moll. "Learning to Reconstruct People in Clothing From a Single RGB Camera". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 1175–1186. DOI: 10.1109/CVPR.2019.00127.

[Ane+25] S. Aneja, S. Weiss, I. Baeza, P. Chandran, G. Zoss, M. Nießner, and D. Bradley. "ScaffoldAvatar: High-Fidelity Gaussian Avatars with Patch Expressions". In: *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*. 2025. DOI: 10.1145/3721238.3730729.

[App22] Apple, Inc. *ARKit Framework - Blendshapes*. 2022. URL: https://developer.apple.com/documentation/arkit/arfaceanchor/2928251-blendshapes (visited on 01/04/2026).

[App23a] Apple Inc. *Vision Framework - VNFaceLandmarks2D*. 2023. URL: https://developer.apple.com/documentation/vision/vnfacelandmarks2d (visited on 01/04/2026).

[App23b] Apple, Inc. *RealityKit Framework - Object Capture - PhotogrammetrySession*. 2023. URL: https://developer.apple.com/documentation/realitykit/photogrammetrysession (visited on 01/04/2026).

[App23c] Apple, Inc. *Vision Framework - VNGeneratePersonSegmentationRequest*. 2023. URL: https://developer.apple.com/documentation/vision/vngeneratepersonsegmentationrequest (visited on 01/04/2026).

[App25] Apple, Inc. *Vision Framework - VNDetectHumanHandPosesRequest*. 2025. URL: https://developer.apple.com/documentation/vision/vndetecthumanhandposerequest (visited on 01/04/2026).

[App26] Apple, Inc. *ARKit Framework*. 2026. URL: https://developer.apple.com/augmented-reality/arkit/ (visited on 01/04/2026).

[AR92] N. Ambady and R. Rosenthal. "Thin Slices of Expressive Behavior as Predictors of Interpersonal Consequences: A Meta-Analysis." In: *Psychological bulletin* 111.2 (1992), pp. 256–274. DOI: 10.1037/0033-2909.111.2.256.

[ARV07] B. Amberg, S. Romdhani, and T. Vetter. "Optimal Step Nonrigid ICP Algorithms for Surface Registration". In: *2007 IEEE Conference on Computer Vision and Pattern Recognition*. 2007. DOI: 10.1109/CVPR.2007.383165.

[AZB15] J. Achenbach, E. Zell, and M. Botsch. "Accurate Face Reconstruction through Anisotropic Fitting and Eye Correction". In: *Vision, Modeling & Visualization*. 2015. DOI: 10.2312/vmv.20151251.

[Bar+21] A. Bartl, S. Wenninger, E. Wolf, M. Botsch, and M. E. Latoschik. "Affordable but not Cheap: A Case Study of the Effects of Two 3D-Reconstruction Methods of Virtual Humans". In: *Frontiers in Virtual Reality* 2 (2021). DOI: 10.3389/frvir.2021.694617.

[BC06] V. Braun and V. Clarke. "Using thematic analysis in psychology". In: *Qualitative Research in Psychology* 3.2 (2006), pp. 77–101. DOI: 10.1191/1478088706qp063oa.

[BKM09] A. Bangor, P. Kortum, and J. Miller. "Determining what individual SUS scores mean: Adding an adjective rating scale". In: *Journal of usability studies* 4.3 (2009), pp. 114–123.

[Bot+06] M. Botsch, R. Sumner, M. Pauly, and M. Gross. "Deformation Transfer for Detail-Preserving Surface Editing". In: *Proceedings of VMV*. 2006.

[Bot+10] M. Botsch, L. Kobbelt, M. Pauly, P. Alliez, and B. Lévy. *Polygon Mesh Processing*. AK Peters, 2010. ISBN: 978-1-56881-426-1.

[BR07] B. J. Brown and S. Rusinkiewicz. "Global non-rigid alignment of 3-D scans". In: *ACM SIGGRAPH 2007 Papers*. 2007, 21–es. DOI: 10.1145/1275808.1276404.

[Bra00] G. Bradski. "The OpenCV Library". In: *Dr. Dobb's Journal of Software Tools* (2000).

[Bro96] J. Brooke. "SUS: A quick and dirty usability scale". In: *Usability evaluation in industry*. Taylor & Francis, 1996, pp. 4–7.

[BS90] V. A. Bowers and H. L. Snyder. "Concurrent versus Retrospective Verbal Protocol for Comparing Window Usability". In: *Proceedings of the Human Factors Society Annual Meeting* 34.17 (1990), pp. 1270–1274. DOI: 10.1177/154193129003401720.

[Bus04] S. R. Buss. "Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods". In: *IEEE Journal of Robotics and Automation* 17 (2004), pp. 1–19.

[BV99] V. Blanz and T. Vetter. "A morphable model for the synthesis of 3D faces". In: *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*. 1999, pp. 187–194. ISBN: 0201485605. DOI: 10.1145/311535.311556.

[BWP13] S. Bouaziz, Y. Wang, and M. Pauly. "Online modeling for realtime facial animation". In: *ACM Transactions on Graphics* 32.4 (2013), pp. 1–10. DOI: 10.1145/2461912.2461976.

[Cao+16] C. Cao, H. Wu, Y. Weng, T. Shao, and K. Zhou. "Real-time facial animation with image-based dynamic avatars". In: *ACM Transactions on Graphics* 35.4 (2016), pp. 1–12. DOI: 10.1145/2897824.2925873.

[Cao+19] Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh. "OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2019). DOI: 10.1109/TPAMI.2019.2929257.

[Cap23a] Captury. *CapturyLive 259*. 2023. URL: https://captury.com/real-time-processing/ (visited on 02/26/2026).

[Cap23b] Captury. *Unity Plugin*. 2023. URL: https://captury.com/resources/ (visited on 02/26/2026).

[Car+20] E. Carrigan, E. Zell, C. Guiard, and R. McDonnell. "Expression Packing: As-Few-As-Possible Training Expressions for Blendshape Transfer". In: *Computer Graphics Forum* 39.2 (2020), pp. 219–233. DOI: 10.1111/cgf.13925.

[Cas+15] D. Casas, O. Alexander, A. W. Feng, G. Fyffe, R. Ichikari, P. Debevec, R. Wang, E. Suma, and A. Shapiro. "Rapid Photorealistic Blendshapes from Commodity RGB-D Sensors". In: *Proceedings of the 19th Symposium on Interactive 3D Graphics and Games*. 2015, pp. 134–134. DOI: 10.1145/2699276.2721398.

[Cas+16] D. Casas, A. Feng, O. Alexander, G. Fyffe, P. Debevec, R. Ichikari, H. Li, K. Olszewski, E. Suma, and A. Shapiro. "Rapid Photorealistic Blendshape Modeling from RGB-D Sensors". In: *Proceedings of the 29th International Conference on Computer Animation and Social Agents*. 2016, pp. 121–129. DOI: 10.1145/2915926.2915936.

[Cha+20] B. Chaudhuri, N. Vesdapunt, L. Shapiro, and B. Wang. "Personalized Face Modeling for Improved Face Reconstruction and Motion Retargeting". In: *Computer Vision – ECCV 2020*. Springer International Publishing, 2020, pp. 142–160. DOI: 10.1007/978-3-030-58558-7_9.

[Cha+94] P. Charbonnier, L. Blanc-Féraud, G. Aubert, and M. Barlaud. "Two deterministic half-quadratic regularization algorithms for computed imaging". In: *Proceedings of 1st International Conference on Image Processing*. Vol. 2. 1994, pp. 168–172. DOI: 10.1109/ICIP.1994.413553.

[Che+17] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam. *Rethinking Atrous Convolution for Semantic Image Segmentation*. 2017. arXiv: 1706.05587.

[Che+21] J. Chen, Y. Zhang, D. Kang, X. Zhe, L. Bao, X. Jia, and H. Lu. *Animatable Neural Radiance Fields from Monocular RGB Videos*. 2021. arXiv: 2106.13629.

[Che+25a] J. Chen, J. Hu, G. Wang, Z. Jiang, T. Zhou, Z. Chen, and C. Lv. "TaoAvatar: Real-Time Lifelike Full-Body Talking Avatars for Augmented Reality via 3D Gaussian Splatting". In: *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*. 2025, pp. 10723–10734. DOI: 10.1109/CVPR52734.2025.01002.

[Che+25b] J. Chen, C. Li, J. Zhang, L. Zhu, B. Huang, H. Chen, and G. H. Lee. "Generalizable Human Gaussians from Single-View Image". In: *Proceedings of the 13th International Conference on Learning Representations*. 2025.

[CKU23] Y. Cai, S. Kinuwaki, and N. Umetani. "FaceZip: Automatic Texture Compression for Facial Blendshapes". In: *Graphics Interface 2023*. 2023.

[Cud+19] D. Cudeiro, T. Bolkart, C. Laidlaw, A. Ranjan, and M. J. Black. "Capture, Learning, and Synthesis of 3D Speaking Styles". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 10093–10103. DOI: 10.1109/CVPR.2019.01034.

[DBH25] D. Deng, M. Bujić, and J. Hamari. "How Avatar User Visual Incongruities Impact the Sense of Embodiment in Virtual Reality: A Systematic Review". In: *Proceedings of the 31st ACM Symposium on Virtual Reality Software and Technology*. 2025. DOI: 10.1145/3756884.3766029.

[de 11] F. de Vignemont. “Embodiment, ownership and disownership”. In: *Consciousness and Cognition* 20.1 (2011), pp. 82–93. DOI: 10.1016/j.concog.2010.09.004.

[DIM24] T. D. Do, C. Isabella Protko, and R. P. McMahan. “Stepping into the Right Shoes: The Effects of User-Matched Avatar Ethnicity and Gender on Sense of Embodiment in Virtual Reality”. In: *IEEE Transactions on Visualization and Computer Graphics* 30.5 (2024), pp. 2434–2443. DOI: 10.1109/TVCG.2024.3372067.

[Dli22] Dlib. *Dlib C++ Library*. 2022. URL: https://dlib.net (visited on 02/27/2026).

[Do+23] T. D. Do, S. Zelenty, M. Gonzalez-Franco, and R. P. McMahan. “VALID: A perceptually validated Virtual Avatar Library for Inclusion and Diversity”. In: *Frontiers in Virtual Reality* 4 (2023). DOI: 10.3389/frvir.2023.1248915.

[Döl+22] N. Döllinger, E. Wolf, D. Mal, S. Wenninger, M. Botsch, M. E. Latoschik, and C. Wienrich. “Resize Me! Exploring the User Experience of Embodied Realistic Modulatable Avatars for Body Image Intervention in Virtual Reality”. In: *Frontiers in Virtual Reality* 3 (2022). DOI: 10.3389/frvir.2022.935449.

[Döl+23] N. Döllinger, M. Beck, E. Wolf, D. Mal, M. Botsch, M. E. Latoschik, and C. Wienrich. “"If It’s Not Me It Doesn’t Make a Difference" – The Impact of Avatar Customization and Personalization on User Experience and Body Awareness in Virtual Reality”. In: *2023 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 2023, pp. 483–492. DOI: 10.1109/ISMAR59233.2023.00063.

[Döl+24] N. Döllinger, D. Mal, S. Keppler, E. Wolf, M. Botsch, J. H. Israel, M. E. Latoschik, and C. Wienrich. “Virtual Body Swapping: A VR-Based Approach to Embodied Third-Person Self-Processing in Mind-Body Therapy”. In: *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*. 2024, pp. 1–18. DOI: 10.1145/3613904.3642328.

[Don+25] Z. Dong, L. Duan, J. Song, M. J. Black, and A. Geiger. “MoGA: 3D Generative Avatar Prior for Monocular Gaussian Avatar Reconstruction”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2025, pp. 13304–13314.

[EF69] P. Ekman and W. V. Friesen. “The repertoire of nonverbal behavior: Categories, origins, usage, and coding”. In: *Semiotica* 1.1 (1969), pp. 49–98. DOI: 10.1515/semi.1969.1.1.49.

[EF78] P. Ekman and W. V. Friesen. *Facial Action Coding System*. 1978. DOI: 10.1037/t27734-000.

[Fac] Faceware Technologies. *Faceware Technologies*. https://facewaretech.com. (Visited on 01/06/2026).

[Far03] G. Farnebäck. "Two-Frame Motion Estimation Based on Polynomial Expansion". In: *Image Analysis*. 2003, pp. 363–370. DOI: 10.1007/3-540-45103-X_50.

[Fen+21] Y. Feng, H. Feng, M. J. Black, and T. Bolkart. "Learning an animatable detailed 3D face model from in-the-wild images". In: *ACM Transactions on Graphics* 40.4 (2021). DOI: 10.1145/3450626.3459936.

[Fie+23] M. L. Fiedler, E. Wolf, N. Döllinger, M. Botsch, M. E. Latoschik, and C. Wienrich. "Embodiment and personalization for self-identification with virtual humans". In: *2023 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*. 2023, pp. 799–800. DOI: 10.1109/VRW58643.2023.00242.

[Fie+24] M. L. Fiedler, E. Wolf, N. Döllinger, D. Mal, M. Botsch, M. E. Latoschik, and C. Wienrich. "From Avatars to Agents: Self-Related Cues through Embodiment and Personalization Affect Body Perception in Virtual Reality". In: *IEEE Transactions on Visualization and Computer Graphics* 30.11 (2024), pp. 7386–7396. DOI: 10.1109/TVCG.2024.3456211.

[Fie+26] M. L. Fiedler, C. Merz, L. Schach, J. Tschanter, M. Botsch, C. Wienrich, and M. E. Latoschik. "Am I Still Me? Visual Congruence Across Reality-Virtuality and Avatar Appearance in Shaping Self-Perception and Behavior". In: *IEEE Transactions on Visualization and Computer Graphics* (2026), pp. 1–11. DOI: 10.1109/TVCG.2026.3679890.

[FMF12] A. P. Field, J. Miles, and Z. Field. *Discovering Statistics Using R*. London ; Thousand Oaks, Calif: Sage, 2012. ISBN: 978-1-4462-0046-9 978-1-4462-0045-2.

[FRS17] A. Feng, E. S. Rosenberg, and A. Shapiro. "Just-in-Time, Viable, 3D Avatars from Scans". In: *Computer Animation and Virtual Worlds* 28.3-4 (2017), e1769. DOI: 10.1002/cav.1769.

[Gaf+21] G. Gafni, J. Thies, M. Zollhofer, and M. Nießner. "Dynamic Neural Radiance Fields for Monocular 4D Facial Avatar Reconstruction". In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, pp. 8649–8658. DOI: 10.1109/CVPR46437.2021.00854.

[Gal+21] D. Gall, D. Roth, J.-P. Stauffert, J. Zarges, and M. E. Latoschik. "Embodiment in Virtual Reality Intensifies Emotional Responses to Virtual Stimuli". In: *Frontiers in Psychology* 12 (2021). DOI: 10.3389/fpsyg.2021.674179.

[Gao+22] X. Gao, C. Zhong, J. Xiang, Y. Hong, Y. Guo, and J. Zhang. "Reconstructing Personalized Semantic Facial NeRF Models from Monocular Video". In: *ACM Transactions on Graphics* 41.6 (2022). DOI: 10.1145/3550454.3555501.

[Gar+13] P. Garrido, L. Valgaert, C. Wu, and C. Theobalt. "Reconstructing detailed dynamic face geometry from monocular video". In: *ACM Transactions on Graphics* 32.6 (2013), pp. 1–10. DOI: 10.1145/2508363.2508380.

[Gar+16] P. Garrido, M. Zollhöfer, D. Casas, L. Valgaerts, K. Varanasi, P. Pérez, and C. Theobalt. "Reconstruction of Personalized 3D Face Rigs from Monocular Video". In: *ACM Transactions on Graphics* 35.3 (2016), pp. 1–15. DOI: 10.1145/2890493.

[Gol09] J. Golbeck. "Trust and nuanced profile similarity in online social networks". In: *ACM Transactions on the Web* 3.4 (2009). DOI: 10.1145/1594173.1594174.

[Gon+10] M. González-Franco, D. Pérez-Marcos, B. Spanlang, and M. Slater. "The contribution of real-time mirror reflections of motor actions on virtual body ownership in an immersive virtual environment". In: *2010 IEEE Virtual Reality Conference (VR)*. 2010, pp. 111–114. DOI: 10.1109/VR.2010.5444805.

[Guo+25] C. Guo, J. Li, Y. Kant, Y. Sheikh, S. Saito, and C. Cao. "Vid2Avatar-Pro: Authentic Avatar from Videos in the Wild via Universal Prior". In: *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2025, pp. 5559–5570. DOI: 10.1109/CVPR52734.2025.00522.

[GW08] R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. 3rd. Prentice Hall, 2008. ISBN: 9780131687288.

[Han+21] J. H. Han, J.-I. Kim, H. Kim, and J. W. Suh. "Generate Individually Optimized Blendshapes". In: *2021 IEEE International Conference on Big Data and Smart Computing (BigComp)*. IEEE, 2021, pp. 114–120. DOI: 10.1109/bigcomp51126.2021.00030.

[Han+25a] T. Han, B. Zhou, A. Liu, Y. Liang, D. Zhang, Z. Lei, and J. Wan. "PESTalk: Speech-Driven 3D Facial Animation with Personalized Emotional Styles". In: *Proceedings of the 33rd ACM International Conference on Multimedia*. 2025, pp. 7893–7901. DOI: 10.1145/3746027.3755190.

[Han+25b] T. Han et al. "PMMTalk: Speech-Driven 3D Facial Animation From Complementary Pseudo Multi-Modal Features". In: *IEEE Transactions on Multimedia* 27 (2025), pp. 2570–2581. DOI: 10.1109/TMM.2024.3521701.

[HHL24] S. Hu, T. Hu, and Z. Liu. "GauHuman: Articulated Gaussian Splatting from Monocular Human Videos". In: *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2024, pp. 20418–20431. DOI: 10.1109/CVPR52733.2024.01930.

[HM17] C.-C. Ho and K. F. MacDorman. "Measuring the uncanny valley effect". In: *International Journal of Social Robotics* 9.1 (2017), pp. 129–139. DOI: 10.1007/s12369-016-0380-9.

[Hor87] B. K. P. Horn. "Closed-Form Solution of Absolute Orientation Using Unit Quaternions". In: *Journal of the Optical Society of America A* 4.4 (1987), pp. 629–642. DOI: 10.1364/JOSAA.4.000629.

[Hu+24] L. Hu, H. Zhang, Y. Zhang, B. Zhou, B. Liu, S. Zhang, and L. Nie. "GaussianAvatar: Towards Realistic Human Avatar Modeling from a Single Video via Animatable 3D Gaussians". In: *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2024, pp. 634–644. DOI: 10.1109/CVPR52733.2024.00067.

[Ian+25] F. Iandola, S. Pidhorskyi, I. Santesteban, D. Gupta, A. Pahuja, N. Bartolovic, F. Yu, E. Garbin, T. Simon, and S. Saito. "SqueezeMe: Mobile-Ready Distillation of Gaussian Full-Body Avatars". In: *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*. 2025. DOI: 10.1145/3721238.3730599.

[IBM22] IBM. *SPSS Statistics*. https://www.ibm.com/products/spss-statistics. 2022. (Visited on 02/27/2026).

[IBP15] A. E. Ichim, S. Bouaziz, and M. Pauly. "Dynamic 3D Avatar Creation from Hand-Held Video Input". In: *ACM Transactions on Graphics* 34.4 (2015). DOI: 10.1145/2766974.

[JH16] S. Jung and C. E. Hughes. "The Effects of Indirect Real Body Cues of Irrelevant Parts on Virtual Body Ownership and Presence". In: *Proceedings of the 26th International Conference on Artificial Reality and Telexistence and the 21st Eurographics Symposium on Virtual Environments*. 2016, pp. 107–114.

[Jia+22] B. Jiang, Y. Hong, H. Bao, and J. Zhang. "SelfRecon: Self Reconstruction Your Digital Avatar from Monocular Video". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 5595–5605. DOI: 10.1109/CVPR52688.2022.00552.

[Jia+23] T. Jiang, X. Chen, J. Song, and O. Hilliges. "InstantAvatar: Learning Avatars From Monocular Video in 60 Seconds". In: *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2023, pp. 16922–16932. DOI: 10.1109/CVPR52729.2023.01623.

[Jia+25] T. Jiang, H.-I. Ho, M. Kaufmann, and J. Song. "PriorAvatar: Efficient and Robust Avatar Creation from Monocular Video Using Learned Priors". In: *Proceedings of the SIGGRAPH Asia 2025 Conference Papers*. 2025. DOI: 10.1145/3757377.3763978.

[JPH21] M. Jin, A. Polis, and J. Hartzel. "Algorithms for Minimization Randomization and the Implementation with an R Package". In: *Communications in Statistics - Simulation and Computation* 50.10 (2021), pp. 3077–3087. DOI: 10.1080/03610918.2019.1619765.

[JSW05] T. Ju, S. Schaefer, and J. Warren. "Mean value coordinates for closed triangular meshes". In: *ACM Transactions on Graphics* 24.3 (2005), pp. 561–566. DOI: 10.1145/1073204.1073229.

[KB14] D. P. Kingma and J. Ba. *Adam: A Method for Stochastic Optimization*. 2014. arXiv: 1412.6980.

[Ken+93] R. S. Kennedy, N. E. Lane, K. S. Berbaum, and M. G. Lilienthal. "Simulator Sickness Questionnaire: An Enhanced Method for Quantifying Simulator Sickness". In: *The international journal of aviation psychology* 3.3 (1993), pp. 203–220. DOI: 10.1207/s15327108ijap0303_3.

[Ker+23] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis. "3D Gaussian Splatting for Real-Time Radiance Field Rendering". In: *ACM Transactions on Graphics* 42.4 (2023). DOI: 10.1145/3592433.

[KGS12] K. Kilteni, R. Groten, and M. Slater. "The Sense of Embodiment in Virtual Reality". In: *Presence: Teleoperators and Virtual Environments* 21.4 (2012), pp. 373–387. DOI: 10.1162/PRES_a_00124.

[Khi+25] R. Khirodkar, T. Bagautdinov, J. Martinez, S. Zhaoen, A. James, P. Selednik, S. Anderson, and S. Saito. "Sapiens: Foundation for Human Vision Models". In: *Computer Vision – ECCV 2024*. Cham, 2025, pp. 206–228. DOI: 10.1007/978-3-031-73235-5_12.

[Kim+18] H. K. Kim, J. Park, Y. Choi, and M. Choe. "Virtual reality sickness questionnaire (VRSQ): Motion sickness measurement index in a virtual reality environment". In: *Applied Ergonomics* 69 (2018), pp. 66–73. DOI: 10.1016/j.apergo.2017.12.016.

[Kir+23] A. Kirillov et al. "Segment Anything". In: *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2023, pp. 3992–4003. DOI: 10.1109/ICCV51070.2023.00371.

[KLC23] D. Y. Kim, H. K. Lee, and K. Chung. "Avatar-mediated experience in the metaverse: The impact of avatar realism on user-avatar relationship". In: *Journal of Retailing and Consumer Services* 73 (2023), p. 103382. DOI: 10.1016/j.jretconser.2023.103382.

[KS15] P. Kortum and M. Sorber. "Measuring the Usability of Mobile Applications for Phones and Tablets". In: *International Journal of Human-Computer Interaction* 31.8 (2015), pp. 518–529. DOI: 10.1080/10447318.2015.1064658.

[Kwo+23] Y. Kwon, L. Liu, H. Fuchs, M. Habermann, and C. Theobalt. "DELIFFAS: Deformable Light Fields for Fast Avatar Synthesis". In: *Advances in Neural Information Processing Systems*. Vol. 36. 2023, pp. 40944–40962.

[LA10] J. Lewis and K.-i. Anjyo. "Direct Manipulation Blendshapes". In: *IEEE Computer Graphics and Applications* 30.4 (2010), pp. 42–50. DOI: 10.1109/mcg.2010.41.

[Lat+19] M. E. Latoschik, F. Kern, J.-P. Stauffert, A. Bartl, M. Botsch, and J.-L. Lugrin. "Not Alone Here?! Scalability and User Experience of Embodied Ambient Crowds in Distributed Social Virtual Reality". In: *IEEE Transactions on Visualization and Computer Graphics* 25.5 (2019), pp. 2134–2144. DOI: 10.1109/TVCG.2019.2899250.

[Lee+19] G. Lee, Z. Deng, S. Ma, T. Shiratori, S. S. Srinivasa, and Y. Sheikh. "Talking With Hands 16.2M: A Large-Scale Dataset of Synchronized Body-Finger Motion and Audio for Conversational Motion Analysis and Synthesis". In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, pp. 763–772. DOI: 10.1109/ICCV.2019.00085.

[Lei+24] J. Lei, Y. Wang, G. Pavlakos, L. Liu, and K. Daniilidis. "GART: Gaussian Articulated Template Models". In: *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2024, pp. 19876–19887. DOI: 10.1109/CVPR52733.2024.01879.

[Lew+14] J. P. Lewis, K. Anjyo, T. Rhee, M. Zhang, F. Pighin, and Z. Deng. "Practice and Theory of Blendshape Facial Models". In: *Eurographics 2014 - State of the Art Reports*. 2014. DOI: 10.2312/egst.20141042.

[Ley+11] M. Leyrer, S. A. Linkenauger, H. H. Bülthoff, U. Kloos, and B. Mohler. "The Influence of Eye Height and Avatars on Egocentric Distance Estimates in Immersive Virtual Environments". In: *Proceedings of the ACM SIGGRAPH Symposium on Applied Perception in Graphics and Visualization*. 2011, pp. 67–74. DOI: 10.1145/2077451.2077464.

[LHY25] S. Liu, K. I. Haque, and Z. Yumak. ""I don't like my avatar": Investigating Human Digital Doubles". In: *Proceedings of the 18th ACM SIGGRAPH Conference on Motion, Interaction, and Games*. 2025. DOI: 10.1145/3769047.3769061.

[Li+13] H. Li, J. Yu, Y. Ye, and C. Bregler. "Realtime facial animation with on-the-fly correctives". In: *ACM Transactions on Graphics* 32.4 (2013). DOI: 10.1145/2461912.2462019.

[Li+15] H. Li, L. Trutoiu, K. Olszewski, L. Wei, T. Trutna, P.-L. Hsieh, A. Nicholls, and C. Ma. "Facial Performance Sensing Head-Mounted Display". In: *ACM Transactions on Graphics* 34.4 (2015). DOI: 10.1145/2766939.

[Li+17] T. Li, T. Bolkart, M. J. Black, H. Li, and J. Romero. "Learning a model of facial shape and expression from 4D scans". In: *ACM Transactions on Graphics* 36.6 (2017), pp. 1–17. DOI: 10.1145/3130800.3130813.

[Li+20a] J. Li, Z. Kuang, Y. Zhao, M. He, K. Bladin, and H. Li. "Dynamic facial asset and rig generation from a single scan". In: *ACM Transactions on Graphics* 39.6 (2020). ISSN: 0730-0301. DOI: 10.1145/3414685.3417817.

[Li+20b] R. Li et al. "Learning Formation of Physically-Based Face Attributes". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 3407–3416. DOI: 10.1109/CVPR42600.2020.00347.

[Li+24] Z. Li, Z. Zheng, L. Wang, and Y. Liu. "Animatable Gaussians: Learning Pose-dependent Gaussian Maps for High-fidelity Human Avatar Modeling". In: *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2024, pp. 19711–19722. DOI: 10.1109/CVPR52733.2024.01864.

[Lim24] Limesurvey GmbH. *LimeSurvey: An Open Source survey tool*. LimeSurvey GmbH, Hamburg, Germany, 2024. URL: https://www.limesurvey.org (visited on 02/27/2026).

[Liu+21] L. Liu, M. Habermann, V. Rudnev, K. Sarkar, J. Gu, and C. Theobalt. "Neural actor: neural free-view synthesis of human actors with pose control". In: *ACM Transactions on Graphics* 40.6 (2021). DOI: 10.1145/3478513.3480528.

[Liu+25] C. Liu, Y. Pan, C. Ding, S. Rahardja, and X. Yang. "MEDTalk: Multimodal Controlled 3D Facial Animation with Dynamic Emotions by Disentangled Embedding". In: *Proceedings of the 33rd ACM International Conference on Multimedia*. 2025, pp. 7538–7547. DOI: 10.1145/3746027.3754933.

[LKJ24] I. Lee, B. Kim, and H. Joo. "Guess The Unseen: Dynamic 3D Scene Reconstruction from Partial 2D Glimpses". In: *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2024, pp. 1062–1071. DOI: 10.1109/CVPR52733.2024.00107.

[Lop+15] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black. "SMPL: a skinned multi-person linear model". In: *ACM Transactions on Graphics* 34.6 (2015). DOI: 10.1145/2816795.2818013.

[Lug+19] C. Lugaresi et al. *MediaPipe: A Framework for Building Perception Pipelines*. 2019. arXiv: 1906.08172.

[LW22] M. E. Latoschik and C. Wienrich. "Congruence and Plausibility, Not Presence: Pivotal Conditions for XR Experiences and Effects, a Novel Approach". In: *Frontiers in Virtual Reality* 3 (2022). DOI: 10.3389/frvir.2022.694433.

[LWP10] H. Li, T. Weise, and M. Pauly. "Example-based facial rigging". In: *ACM Transactions on Graphics* 29.4 (2010). DOI: 10.1145/1778765.1778769.

[Ma+21] S. Ma, T. Simon, J. Saragih, D. Wang, Y. Li, F. D. La Torre, and Y. Sheikh. "Pixel Codec Avatars". In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, pp. 64–73. DOI: 10.1109/CVPR46437.2021.00013.

[Ma+24] S. Ma, Y. Weng, T. Shao, and K. Zhou. "3D Gaussian Blendshapes for Head Avatar Animation". In: *ACM SIGGRAPH 2024 Conference Papers*. 2024. DOI: 10.1145/3641519.3657462.

[Ma+25] S. Ma, Y. Zheng, Y. Weng, and K. Zhou. "Relightable Gaussian blendshapes for head avatar animation". In: *Computers & Graphics* 132 (2025), p. 104324. DOI: 10.1016/j.cag.2025.104324.

[Mal+22] D. Mal, E. Wolf, N. Döllinger, M. Botsch, C. Wienrich, and M. E. Latoschik. "Virtual Human Coherence and Plausibility – Towards a Validated Scale". In: *2022 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*. 2022, pp. 788–789. DOI: 10.1109/VRW55335.2022.00245.

[Mal+24] D. Mal, E. Wolf, N. Döllinger, M. Botsch, C. Wienrich, and M. E. Latoschik. "From 2D-Screens to VR: Exploring the Effect of Immersion on the Plausibility of Virtual Humans". In: *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*. 2024. DOI: 10.1145/3613905.3650773.

[Met23] Meta. *Oculus Avatar SDK*. 2023. URL: https://developers.meta.com/horizon/documentation/native/pc/as-avatars-sdk-intro/ (visited on 04/02/2026).

[MFH12] D. Matsumoto, M. G. Frank, and H. S. Hwang. "Reading People: Introduction to the World of Nonverbal Behavior". In: *Nonverbal Communication: Science and Applications*. SAGE Publications, Inc., 2012, pp. 3–14. ISBN: 9781452244037. DOI: 10.4135/9781452244037.n1.

[Mil+21] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. "NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis". In: *Communications of the ACM* 65.1 (2021), pp. 99–106. DOI: 10.1145/3503250.

[Min+25] X. Ming, J. Li, J. Ling, L. Zhang, and F. Xu. "High-Quality Mesh Blendshape Generation from Face Videos via Neural Inverse Rendering". In: *Computer Vision – ECCV 2024*. 2025, pp. 106–125. DOI: 10.1007/978-3-031-72897-6_7.

[MLT89] N. Magnenat-Thalmann, R. Laperrière, and D. Thalmann. "Joint-dependent local deformations for hand animation and object grasping". In: *Proceedings on Graphics Interface '88*. 1989, pp. 26–33.

[MMK12] M. Mori, K. F. MacDorman, and N. Kageki. "The Uncanny Valley [From the Field]". In: *IEEE Robotics Automation Magazine* 19.2 (2012), pp. 98–100. DOI: 10.1109/MRA.2012.2192811.

[Moh+10] B. J. Mohler, S. H. Creem-Regehr, W. B. Thompson, and H. H. Bülthoff. "The Effect of Viewing a Self-Avatar on Distance Judgments in an HMD-Based Virtual Environment". In: *Presence: Teleoperators and Virtual Environments* 19.3 (2010), pp. 230–242. DOI: 10.1162/pres.19.3.230.

[Mor+24a] A. Moreau, J. Song, H. Dhamo, R. Shaw, Y. Zhou, and E. Pérez-Pellitero. "Human Gaussian Splatting: Real-time Rendering of Animatable Avatars". In: *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2024, pp. 788–798. DOI: 10.1109/CVPR52733.2024.00081.

[Mor+24b] W. Morgenstern, M. T. Bagdasarian, A. Hilsmann, and P. Eisert. "Animatable Virtual Humans: Learning Pose-Dependent Human Representations in UV Space for Interactive Performance Synthesis". In: *IEEE Transactions on Visualization and Computer Graphics* 30.5 (2024), pp. 2644–2650. DOI: 10.1109/TVCG.2024.3372117.

[Mot+23] A. Mottelson, A. Muresan, K. Hornbæk, and G. Makransky. "A Systematic Review and Meta-Analysis of the Effectiveness of Body Ownership Illusions in Virtual Reality". In: *ACM Transactions on Computer-Human Interaction* 30.5 (2023). DOI: 10.1145/3590767.

[MSS25] G. Moon, T. Shiratori, and S. Saito. "Expressive Whole-Body 3D Gaussian Avatar". In: *Computer Vision – ECCV 2024*. 2025, pp. 19–35. DOI: 10.1007/978-3-031-72940-9_2.

[Mül+22] T. Müller, A. Evans, C. Schied, and A. Keller. "Instant Neural Graphics Primitives with a Multiresolution Hash Encoding". In: *ACM Transactions on Graphics* 41.4 (2022). DOI: 10.1145/3528223.3530127.

[Mur+20] M. Murcia-López, T. Collingwoode-Williams, W. Steptoe, R. Schwartz, T. J. Loving, and M. Slater. "Evaluating Virtual Reality Experiences Through Participant Choices". In: *2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. 2020, pp. 747–755. DOI: 10.1109/VR46266.2020.00098.

[Mys22] S. Mystakidis. "Metaverse". In: *Encyclopedia* 2.1 (2022), pp. 486–497. DOI: 10.3390/encyclopedia2010031.

[Nag17] E. van der Nagel. "From usernames to profiles: the development of pseudonymity in Internet communication". In: *Internet Histories* 1.4 (2017), pp. 312–331. DOI: 10.1080/24701475.2017.1389548.

[Nat25] NaturalPoint, Inc. *OptiTrack: Motion Capture Systems & Software*. 2025. URL: https://optitrack.com (visited on 01/06/2026).

[Nor10] G. Norman. "Likert Scales, Levels of Measurement and the "Laws" of Statistics". In: *Advances in Health Sciences Education* 15.5 (2010), pp. 625–632. DOI: 10.1007/s10459-010-9222-y.

[OBB20] A. A. A. Osman, T. Bolkart, and M. J. Black. "STAR: Sparse Trained Articulated Human Body Regressor". In: *European Conference on Computer Vision (ECCV)*. 2020, pp. 598–613. DOI: 10.1007/978-3-030-58539-6_36.

[Ols+16] K. Olszewski, J. J. Lim, S. Saito, and H. Li. "High-Fidelity Facial and Speech Animation for VR HMDs". In: *ACM Transactions on Graphics* 35.6 (2016). ISSN: 0730-0301, 1557-7368. DOI: 10.1145/2980179.2980252.

[OZS08] V. C. Orvalho, E. Zacur, and A. Susin. "Transferring the Rig and Animations from a Character to Different Face Models". In: *Computer Graphics Forum* 27.8 (2008), pp. 1997–2012. DOI: 10.1111/j.1467-8659.2008.01187.x.

[Pan+24] H. Pang, H. Zhu, A. Kortylewski, C. Theobalt, and M. Habermann. "ASH: Animatable Gaussian Splats for Efficient and Photoreal Human Rendering". In: *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2024, pp. 1165–1175. DOI: 10.1109/CVPR52733.2024.00117.

[Pan+26] J. Pan, X. Li, J. Bai, and J. Dai. "LiteNeRFAvatar: A lightweight NeRF with local feature learning for dynamic human avatar". In: *Pattern Recognition* 170 (2026), p. 112008. DOI: 10.1016/j.patcog.2025.112008.

[Pas+20] S. Pastel, C.-H. Chen, K. Petri, and K. Witte. "Effects of body visualization on performance in head-mounted display virtual reality". In: *PLOS ONE* 15.9 (2020). DOI: 10.1371/journal.pone.0239226.

[Pau+25] P. Paudel, A. Khanal, D. P. Paudel, J. Tandukar, and A. Chhatkuli. "iHuman: Instant Animatable Digital Humans From Monocular Videos". In: *Computer Vision – ECCV 2024*. 2025, pp. 304–323. DOI: 10.1007/978-3-031-73226-3_18.

[PBR22] J. Pinheiro, D. Bates, and R Core Team. *nlme: Linear and Nonlinear Mixed Effects Models*. R package version 3.1-161. 2022. URL: https://CRAN.R-project.org/package=nlme (visited on 02/27/2026).

[Pen+21] S. Peng, J. Dong, Q. Wang, S. Zhang, Q. Shuai, X. Zhou, and H. Bao. "Animatable Neural Radiance Fields for Modeling Dynamic Human Bodies". In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021, pp. 14294–14303. DOI: 10.1109/ICCV48922.2021.01405.

[Pen+23] Z. Peng, H. Wu, Z. Song, H. Xu, X. Zhu, J. He, H. Liu, and Z. Fan. "EmoTalk: Speech-Driven Emotional Disentanglement for 3D Face Animation". In: *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2023, pp. 20630–20640. DOI: 10.1109/ICCV51070.2023.01891.

[Pen+25] S. Peng, W. Xie, Z. Wang, X. Guo, Z. Chen, B. Yang, and X. Dong. "RMAvatar: Photorealistic human avatar reconstruction from monocular video based on rectified mesh-embedded Gaussians". In: *Graphical Models* 139 (2025), p. 101266. DOI: 10.1016/j.gmod.2025.101266.

[PGB03] P. Pérez, M. Gangnet, and A. Blake. "Poisson image editing". In: *ACM Transactions on Graphics* 22.3 (2003), pp. 313–318. DOI: 10.1145/882262.882269.

[Qia+24] S. Qian, T. Kirschstein, L. Schoneveld, D. Davoli, S. Giebenhain, and M. Nießner. "GaussianAvatars: Photorealistic Head Avatars with Rigged 3D Gaussians". In: *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2024, pp. 20299–20309. DOI: 10.1109/CVPR52733.2024.01919.

[R C22] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria, 2022. URL: https://www.R-project.org/ (visited on 02/27/2026).

[Rad+23] R. Radiah, D. Roth, F. Alt, and Y. Abdelrahman. "The Influence of Avatar Personalization on Emotions in VR". In: *Multimodal Technologies and Interaction* 7.4 (2023). DOI: 10.3390/mti7040038.

[Rav+20] N. Ravi, J. Reizenstein, D. Novotny, T. Gordon, W.-Y. Lo, J. Johnson, and G. Gkioxari. "Accelerating 3D Deep Learning with PyTorch3D". In: (2020). arXiv: 2007.08501.

[RDP99] K. Robinette, H. Daanen, and E. Paquet. "The CAESAR project: a 3-D surface anthropometry survey". In: *Second International Conference on 3-D Digital Imaging and Modeling (Cat. No.PR00062)*. 1999, pp. 380–386. DOI: 10.1109/IM.1999.805368.

[RE20] E. L. Rosenberg and P. Ekman. *What the face reveals: Basic and applied studies of spontaneous expression using the Facial Action Coding System (FACS)*. Oxford University Press, 2020.

[Rib+17] R. B. i. Ribera, E. Zell, J. P. Lewis, J. Noh, and M. Botsch. "Facial retargeting with automatic range of motion alignment". In: *ACM Transactions on Graphics* 36.4 (2017). DOI: 10.1145/3072959.3073674.

[Rib+20] E. Riba, D. Mishkin, J. Shi, D. Ponsa, F. Moreno-Noguer, and G. Bradski. "A survey on Kornia: an Open Source Differentiable Computer Vision Library for PyTorch". In: 2020. arXiv: 2009.10521.

[Ric+21] A. Richard, C. Lea, S. Ma, J. Gall, F. de la Torre, and Y. Sheikh. "Audio- and Gaze-Driven Facial Animation of Codec Avatars". In: *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*. 2021, pp. 41–50. DOI: 10.1109/WACV48630.2021.00009.

[RJ01] K. Ruhleder and B. Jordan. "Co-constructing non-mutual realities: Delay-generated trouble in distributed interaction". In: *Computer Supported Cooperative Work (CSCW)* 10.1 (2001), pp. 113–138. DOI: 10.1023/A:1011243905593.

[RL20] D. Roth and M. E. Latoschik. "Construction of the virtual embodiment questionnaire (VEQ)". In: *IEEE Transactions on Visualization and Computer Graphics* 26.12 (2020), pp. 3546–3556. DOI: 10.1109/TVCG.2020.3023603.

[RMN21] S. k. Renganayagalu, S. C. Mallam, and S. Nazir. "Effectiveness of VR Head Mounted Displays in Professional Training: A Systematic Review". In: *Technology, Knowledge and Learning* 26.4 (2021), pp. 999–1041. DOI: 10.1007/s10758-020-09489-9.

[Rob+21] R. A. Roberts, R. K. dos Anjos, A. Maejima, and K. Anjyo. "Deformation transfer survey". In: *Computers & Graphics* 94 (2021), pp. 52–61. DOI: 10.1016/j.cag.2020.10.004.

[Rot+16] D. Roth, C. Bloch, A.-K. Wilbers, M. E. Latoschik, K. Kaspar, and G. Bente. "What You See is What You Get: Channel Dominance in the Decoding of Affective Nonverbal Behavior Displayed by Avatars". In: *Presentation at the 66th Annual Conference of the International Communication Association (ICA)*. 2016.

[Rot+19] D. Roth, G. Bente, P. Kullmann, D. Mal, C. F. Purps, K. Vogeley, and M. E. Latoschik. "Technologies for Social Augmentations in User-Embodied Virtual Reality". In: *25th ACM Symposium on Virtual Reality Software and Technology*. 2019. DOI: 10.1145/3359996.3364269.

[Sai13] J. Saito. "Smooth contact-aware facial blendshapes transfer". In: *Proceedings of the 2013 Digital Production Symposium*. 2013, pp. 7–12. DOI: 10.1145/2491832.2491836.

[Sal+23] A. Salagean, E. Crellin, M. Parsons, D. Cosker, and D. Stanton Fraser. "Meeting Your Virtual Twin: Effects of Photorealism and Personalization on Embodiment, Self-Identification and Perception of Self-Avatars in Virtual Reality". In: *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. 2023. DOI: 10.1145/3544548.3581182.

[Sam+21] M. Sampaio, M. V. Navarro Haro, B. De Sousa, W. Vieira Melo, and H. G. Hoffman. "Therapists Make the Switch to Telepsychology to Safely Continue Treating Their Patients During the COVID-19 Pandemic. Virtual Reality Telepsychology May Be Next". In: *Frontiers in Virtual Reality* 1 (2021). DOI: 10.3389/frvir.2020.576421.

[SE93] H. A. Simon and K. A. Ericsson. *Protocol analysis: Verbal reports as data*. The MIT Press, 1993. ISBN: 978-0262550239.

[Seo+11] Y. Seol, J. Seo, P. H. Kim, J. P. Lewis, and J. Noh. "Artist friendly facial animation retargeting". In: *Proceedings of the 2011 SIGGRAPH Asia Conference*. 2011, pp. 1–10. DOI: 10.1145/2024156.2024196.

[Seu+21] L. M. Seuren, J. Wherton, T. Greenhalgh, and S. E. Shaw. "Whose Turn Is It Anyway? Latency and the Organization of Turn-Taking in Video-Mediated Interaction". In: *Journal of Pragmatics* 172 (2021), pp. 63–78. DOI: 10.1016/j.pragma.2020.11.005.

[SF16] J. L. Schönberger and J.-M. Frahm. "Structure-from-Motion Revisited". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 4104–4113. DOI: 10.1109/CVPR.2016.445.

[SFW17] R. Skarbez, J. Frederick P. Brooks, and M. C. Whitton. "A Survey of Presence and Related Concepts". In: *ACM Computing Surveys* 50.6 (2017), p. 96. DOI: 10.1145/3134301.

[Sha+24] Z. Shao, Z. Wang, Z. Li, D. Wang, X. Lin, Y. Zhang, M. Fan, and Z. Wang. "SplattingAvatar: Realistic Real-Time Human Avatars with Mesh-Embedded Gaussian Splatting". In: *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2024, pp. 1606–1616. DOI: 10.1109/CVPR52733.2024.00159.

[She+24] A. Shetty, M. Habermann, G. Sun, D. Luvizon, V. Golyanik, and C. Theobalt. "Holoported Characters: Real-time Free-viewpoint Rendering of Humans from Sparse RGB Cameras". In: *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2024, pp. 1206–1215. DOI: 10.1109/CVPR52733.2024.00121.

[Shi+25] C. Shi, S. Jia, J. Liu, Y. Zhang, L. Zhu, Z. Yang, J. Ma, C. Niu, and C. Lv. "HRM^{2}Avatar: High-Fidelity Real-Time Mobile Avatars from Monocular Phone Scans". In: *Proceedings of the SIGGRAPH Asia 2025 Conference Papers*. 2025. DOI: 10.1145/3757377.3763894.

[SL16] J. Sauro and J. R. Lewis. *Quantifying the user experience: Practical statistics for user research*. Morgan Kaufmann, 2016.

[Sla+10] M. Slater, B. Spanlang, M. V. Sanchez-Vives, and O. Blanke. "First Person Experience of Body Transfer in Virtual Reality". In: *PLOS ONE* 5.5 (2010). DOI: 10.1371/journal.pone.0010564.

[SM25] G. Sim and G. Moon. "PERSONA: Personalized Whole-Body 3D Avatar with Pose-Driven Deformations from a Single Image". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2025, pp. 12670–12680.

[SML16] Y. Seol, W.-C. Ma, and J. P. Lewis. "Creating an actor-specific facial rig from performance capture". In: *Proceedings of the 2016 Digital Production Symposium*. 2016, pp. 13–17. DOI: 10.1145/2947688.2947693.

[SNL20] J.-P. Stauffert, F. Niebling, and M. E. Latoschik. "Simultaneous Run-Time Measurement of Motion-to-Photon Latency and Latency Jitter". In: *2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. 2020, pp. 636–644. DOI: 10.1109/VR46266.2020.00086.

[Son+26] H. Song, B. Yoon, S. Yang, S. Kang, H. Kim, H. Metzmacher, and W. Woo. "VRGaussianAvatar: Integrating 3D Gaussian Avatars into VR". In: *IEEE Transactions on Visualization and Computer Graphics* (2026), pp. 1–11. DOI: 10.1109/TVCG.2026.3680708.

[SP04] R. W. Sumner and J. Popović. "Deformation transfer for triangle meshes". In: *ACM Transactions on Graphics* 23.3 (2004), pp. 399–405. DOI: 10.1145/1015706.1015736.

[Ste+16] A. Steed, Y. Pan, F. Zisch, and W. Steptoe. "The impact of a self-avatar on cognitive load in immersive virtual reality". In: *2016 IEEE Virtual Reality (VR)*. 2016, pp. 67–76. DOI: 10.1109/VR.2016.7504689.

[Ste96] R. Steinmetz. "Human Perception of Jitter and Media Synchronization". In: *IEEE Journal on Selected Areas in Communications* 14.1 (1996), pp. 61–72. ISSN: 07338716. DOI: 10.1109/49.481694.

[Tan+25a] J. Tang, D. Davoli, T. Kirschstein, L. Schoneveld, and M. Nießner. "GAF: Gaussian Avatar Reconstruction from Monocular Videos via Multi-view Diffusion". In: *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2025, pp. 5546–5558. DOI: 10.1109/CVPR52734.2025.00521.

[Tan+25b] Z. Tang, H. Yang, H. Zhang, J. Chen, and D. Huang. "3D²-Actor: Learning Pose-Conditioned 3D-Aware Denoiser for Realistic Gaussian Avatar Modeling". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 39.7 (2025), pp. 7338–7346. DOI: 10.1609/aaai.v39i7.32789.

[Teo+24] K. Teotia, H. Kim, P. Garrido, M. Habermann, M. Elgharib, and C. Theobalt. "GaussianHeads: End-to-End Learning of Drivable Gaussian Head Avatars from Coarse-to-fine Representations". In: *ACM Transactions on Graphics* 43.6 (2024). DOI: 10.1145/3687927.

[Thi+18] J. Thies, M. Zollhöfer, M. Stamminger, C. Theobalt, and M. Nießner. "Face2Face: real-time face capture and reenactment of RGB videos". In: *ACM Communications of the ACM* 62.1 (2018), pp. 96–104. DOI: 10.1145/3292039.

[Tur+21] C. Turbyne, A. Goedhart, P. de Koning, F. Schirmbeck, and D. Denys. "Systematic Review and Meta-Analysis of Virtual Reality in Mental Healthcare: Effects of Full Body Illusions on Body Image Disturbance". In: *Frontiers in Virtual Reality* 2 (2021). DOI: 10.3389/frvir.2021.657638.

[Uni20] Unity Technologies. *Unity 2020.3.25f1*. 2020. URL: https://unity.com/ (visited on 03/03/2026).

[Val24a] Valve Corporation. *Steam VR 2.3*. 2024. URL: https://store.steampowered.com/app/250820/SteamVR/ (visited on 03/03/2026).

[Val24b] Valve Corporation. *Steam VR Plugin 2.7.3*. 2024. URL: https://assetstore.unity.com/packages/tools/integration/steamvr-plugin-32647 (visited on 03/03/2026).

[Vil+22] M. Villanueva Aylagas, H. Anadon Leon, M. Teye, and K. Tollmar. "Voice2Face: Audio-driven Facial and Tongue Rig Animations with cVAEs". In: *Computer Graphics Forum* (2022). DOI: 10.1111/cgf.14640.

[Vol+22] M. Volonte, E. Ofek, K. Jakubzak, S. Bruner, and M. Gonzalez-Franco. "HeadBox: A Facial Blendshape Animation Toolkit for the Microsoft Rocketbox Library". In: *2022 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*. 2022, pp. 39–42. DOI: 10.1109/VRW55335.2022.00015.

[VS09] S. Villagrasa and A. Susín Sánchez. "FACe! 3D Facial Animation System based on FACS". In: *IV Iberoamerican Symposium in Computer Graphics*. 2009, pp. 203–209.

[Wal+18] T. Waltemate, D. Gall, D. Roth, M. Botsch, and M. E. Latoschik. "The Impact of Avatar Personalization and Immersion on Virtual Body Ownership, Presence, and Emotional Response". In: *IEEE Transactions on Visualization and Computer Graphics* 24.4 (2018), pp. 1643–1652. DOI: 10.1109/TVCG.2018.2794629.

[Wan+04] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli. "Image quality assessment: from error visibility to structural similarity". In: *IEEE Transactions on Image Processing* 13.4 (2004), pp. 600–612. DOI: 10.1109/TIP.2003.819861.

[Wan+23] T. Wang et al. "RODIN: A Generative Model for Sculpting 3D Digital Avatars Using Diffusion". In: *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2023, pp. 4563–4573. DOI: 10.1109/CVPR52729.2023.00443.

[Wan+24] S. Wang, B. Antic, A. Geiger, and S. Tang. "IntrinsicAvatar: Physically Based Inverse Rendering of Dynamic Humans from Monocular Videos via Explicit Ray Tracing". In: *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2024, pp. 1877–1888. DOI: 10.1109/CVPR52733.2024.00184.

[Wan+25] R. Wang et al. "FRESA: Feedforward Reconstruction of Personalized Skinned Avatars from Few Images". In: *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2025, pp. 281–291. DOI: 10.1109/CVPR52734.2025.00035.

[Wat87] K. Waters. "A muscle model for animation three-dimensional facial expression". In: *ACM SIGGRAPH Computer Graphics* 21.4 (1987), pp. 17–24. DOI: 10.1145/37402.37405.

[WBS23] N. Wagner, M. Botsch, and U. Schwanecke. "SoftDECA: Computationally Efficient Physics-Based Facial Animations". In: *Proceedings of the 16th ACM SIGGRAPH Conference on Motion, Interaction and Games*. 2023. DOI: 10.1145/3623264.3624439.

[Wei+11] T. Weise, S. Bouaziz, H. Li, and M. Pauly. "Realtime performance-based facial animation". In: *ACM Transactions on Graphics* 30.4 (2011). DOI: 10.1145/2010324.1964972.

[Wen+20] S. Wenninger, J. Achenbach, A. Bartl, M. E. Latoschik, and M. Botsch. "Realistic Virtual Humans from Smartphone Videos". In: *Proceedings of the 26th ACM Symposium on Virtual Reality Software and Technology*. 2020. DOI: 10.1145/3385956.3418940.

[Wen+22] C.-Y. Weng, B. Curless, P. P. Srinivasan, J. T. Barron, and I. Kemelmacher-Shlizerman. "HumanNeRF: Free-Viewpoint Rendering of Moving People From Monocular Video". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 16189–16199. DOI: 10.1109/CVPR52688.2022.01573.

[Wen+24] J. Wen, X. Zhao, Z. Ren, A. G. Schwing, and S. Wang. "GoMAvatar: Efficient Animatable Human Modeling from Monocular Video Using Gaussians-on-Mesh". In: *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2024, pp. 2059–2069. DOI: 10.1109/CVPR52733.2024.00201.

[Wie+03] T. Wiegand, G. Sullivan, G. Bjontegaard, and A. Luthra. "Overview of the H.264/AVC video coding standard". In: *IEEE Transactions on Circuits and Systems for Video Technology* 13.7 (2003), pp. 560–576. DOI: 10.1109/TCSVT.2003.815165.

[Wil22] R. R. Wilcox. *Introduction to robust estimation and hypothesis testing*. Academic Press, 2022. ISBN: 978-0128200988.

[WMK14] P. Wagner, Z. Malisz, and S. Kopp. "Gesture and Speech in Interaction: An Overview". In: *Speech Communication* 57 (2014), pp. 209–232. DOI: 10.1016/j.specom.2013.09.008.

[Wol+21] E. Wolf, N. Merdan, N. Döllinger, D. Mal, C. Wienrich, M. Botsch, and M. E. Latoschik. "The Embodiment of Photorealistic Avatars Influences Female Body Weight Perception in Virtual Reality". In: *2021 IEEE Virtual Reality and 3D User Interfaces (VR)*. 2021, pp. 65–74. DOI: 10.1109/VR50410.2021.00027.

[Wol+22a] E. Wolf, N. Döllinger, D. Mal, S. Wenninger, B. Andrea, M. Botsch, M. E. Latoschik, and C. Wienrich. "Does Distance Matter? Embodiment and Perception of Personalized Avatars in Relation to the Self-Observation Distance in Virtual Reality". In: *Frontiers in Virtual Reality* 3 (2022). DOI: 10.3389/frvir.2022.1031093.

[Wol+22b] E. Wolf, M. L. Fiedler, N. Döllinger, C. Wienrich, and M. E. Latoschik. "Exploring Presence, Avatar Embodiment, and Body Perception with a Holographic Augmented Reality Mirror". In: *2022 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. 2022, pp. 350–359. DOI: 10.1109/VR51125.2022.00054.

[Xu+24] Y. Xu, B. Chen, Z. Li, H. Zhang, L. Wang, Z. Zheng, and Y. Liu. "Gaussian Head Avatar: Ultra High-fidelity Head Avatar via Dynamic Gaussians". In: *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2024, pp. 1931–1941. DOI: 10.1109/CVPR52733.2024.00189.

[Xu+25] W. Xu, Y. Zhan, Z. Zhong, and X. Sun. "Sequential Gaussian Avatars with Hierarchical Motion Context". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2025, pp. 13592–13603.

[Yan+24] X. Yang, X. Chen, D. Gao, S. Wang, X. Han, and B. Wang. "HAVE-FUN: Human Avatar Reconstruction from Few-Shot Unconstrained Images". In: *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2024, pp. 742–752. DOI: 10.1109/CVPR52733.2024.00077.

[YC08] A. C. Younkin and P. J. Corriveau. "Determining the Amount of Audio-Video Synchronization Errors Perceptible to the Average End-User". In: *IEEE Transactions on Broadcasting* 54.3 (2008), pp. 623–627. DOI: 10.1109/TBC.2008.2002102.

[Yoo+19] B. Yoon, H.-i. Kim, G. A. Lee, M. Billinghurst, and W. Woo. "The Effect of Avatar Appearance on Social Presence in an Augmented Reality Remote Collaboration". In: *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. 2019, pp. 547–556. DOI: 10.1109/VR.2019.8797719.

[Yu+23] W. Yu et al. "NOFA: NeRF-Based One-Shot Facial Avatar Reconstruction". In: *ACM SIGGRAPH 2023 Conference Proceedings*. 2023. DOI: 10.1145/3588432.3591555.

[Zha+25] Z. Zhai, G. Chen, W. Wang, D. Zheng, and J. Xiao. "TAGA: Self-supervised Learning for Template-free Animatable Gaussian Articulated Model". In: *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2025, pp. 21159–21169. DOI: 10.1109/CVPR52734.2025.01971.

[Zha+26] J. Zhang, X. Li, H. Jia, J. Li, Z. Su, G. Wang, and K. Li. "LoGAvatar: Local Gaussian Splatting for human avatar modeling from monocular video". In: *Computer-Aided Design* 190 (2026), p. 103973. DOI: 10.1016/j.cad.2025.103973.

[Zhe+22] Z. Zheng, H. Huang, T. Yu, H. Zhang, Y. Guo, and Y. Liu. "Structured Local Radiance Fields for Human Avatar Modeling". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 15872–15882. DOI: 10.1109/CVPR52688.2022.01543.

[Zhe+23] Z. Zheng, X. Zhao, H. Zhang, B. Liu, and Y. Liu. "AvatarReX: Real-time Expressive Full-body Avatars". In: *ACM Transactions on Graphics* 42.4 (2023). DOI: 10.1145/3592101.

[ZSN05] T. Zinßer, J. Schmidt, and H. Niemann. "Point Set Registration with Integrated Scale Estimation". In: *International Conference on Pattern Recognition and Image Processing (PRIP 2005)*. 2005.

APPENDIX

A.1 GART Configuration

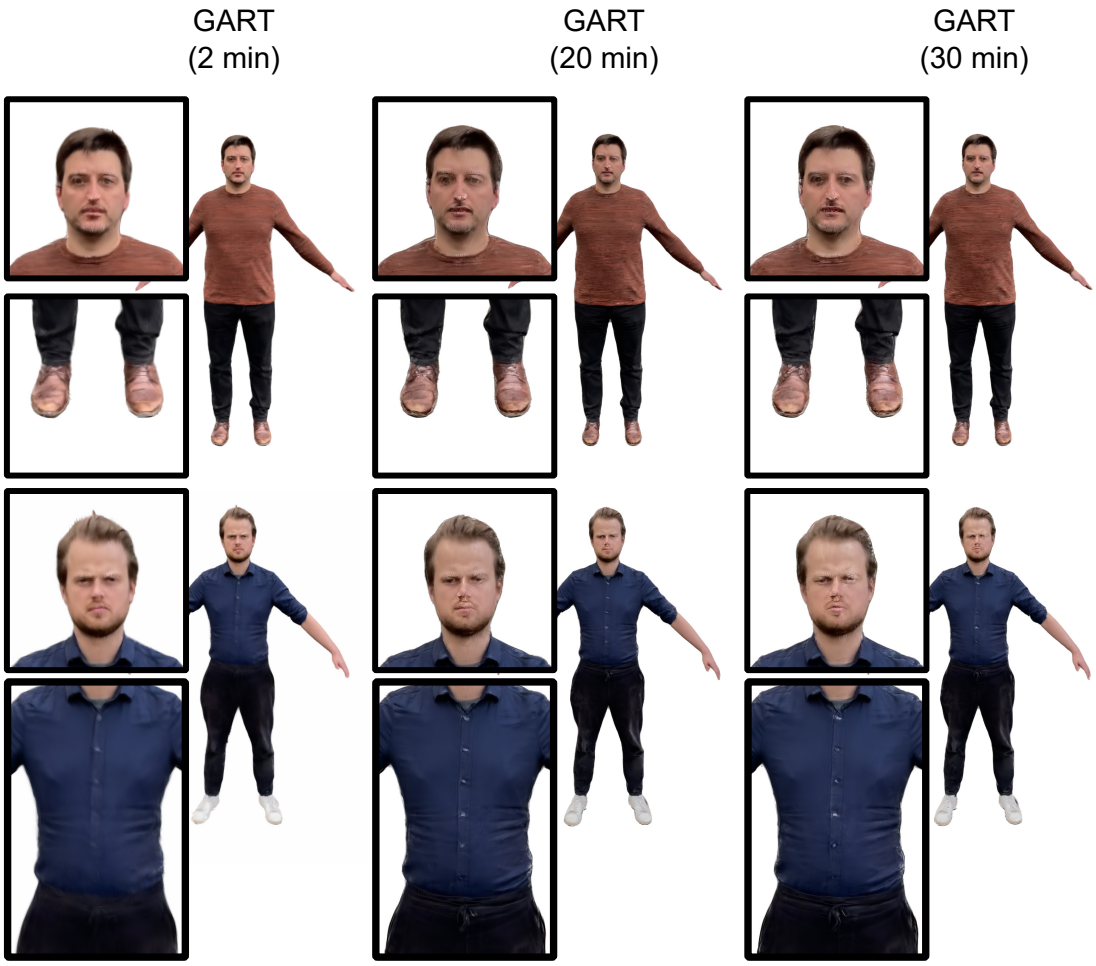


Figure A.1: Reconstructions of GART [Lei+24] with different training times.

We tested GART [Lei+24] with different training times. As Figure A.1 shows, longer training times ensure sharper output (e.g., shoes, buttons on the blue shirt), but result in more artifacts. But longer training times also degraded the quality of the faces, e.g., artifacts on lips and noses. As faces are important for the quality of an avatar, we have, therefore, opted for the two-minute training.

A.2 InstantAvatar Configuration

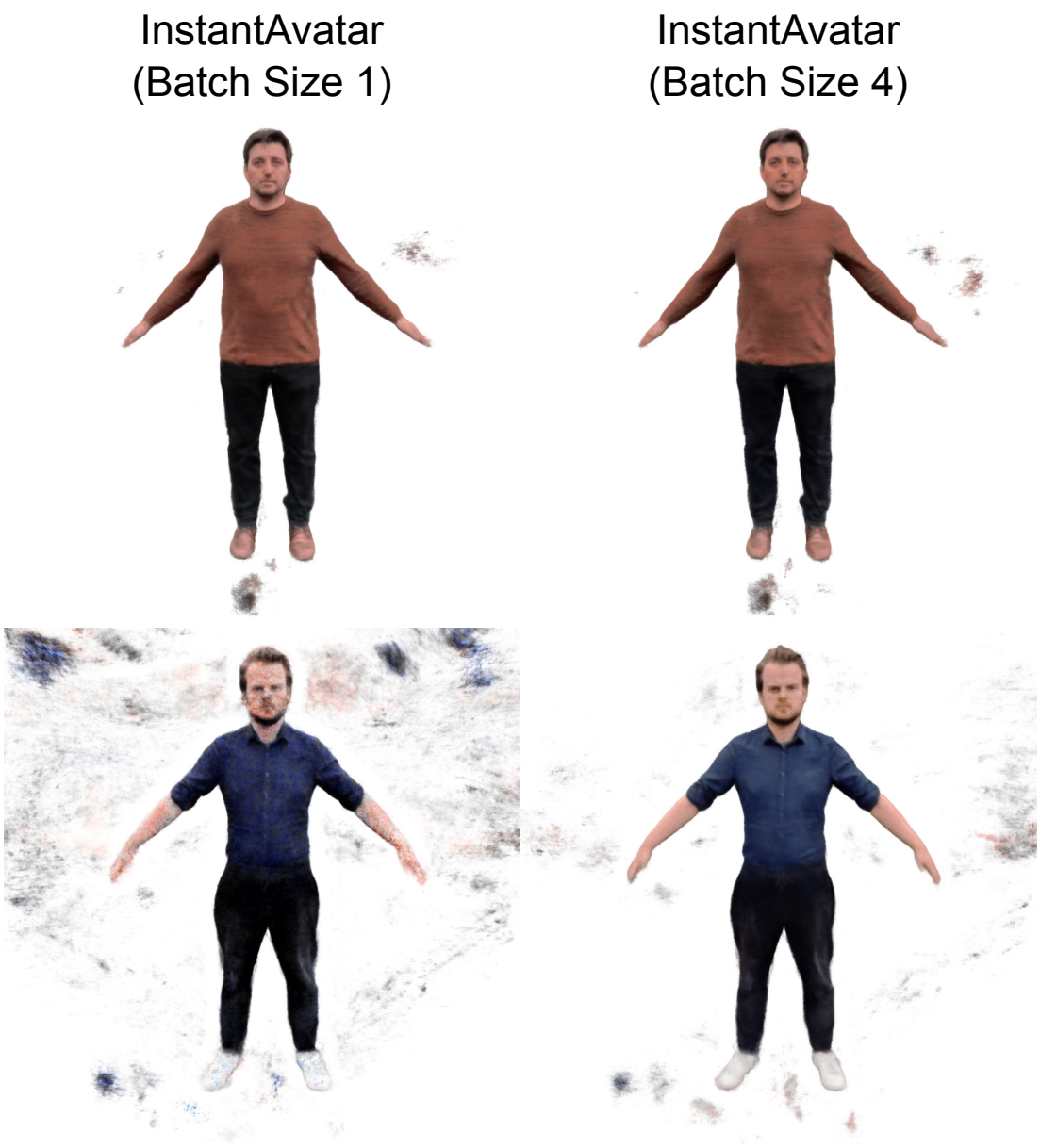


Figure A.2: Reconstructions of InstantAvatar [Jia+23] with different batch sizes.

We computed the InstantAvatar [Jia+23] results with two different batch sizes (1 and 4). In our tests, a batch size of four led to significantly fewer artifacts (see Figure A.2). Therefore, we opted for the training configuration with batch size four.

A.3 Body Movement Tasks

The body movement tasks were instructed in German and have only been translated into English for publication.

Instructions:

1. Please stand relaxed and allow your arms to hang down loosely. In the following, you shall perform various movement tasks. Please perform only the movements specified in the audio instructions and try not to get distracted during this phase. In case you have any important questions or feel unwell, you can always reach out to the experimenter. We will now start with the tasks.
2. Hold now your dominant hand in front of you as if you want to wave at your reflection. The palm of your hand should be facing the mirror.
3. Look at the back of your outstretched hand while waving towards your reflection.
4. Please continue to wave towards your reflection while looking at the palm of the same hand in the mirror.
5. Lower your hand to allow both arms to hang loosely down and look at your reflection in the mirror.
6. Hold now your non-dominant hand in front of you as if you want to wave towards your reflection. The palm of your hand should be facing the mirror.
7. Look at the back of your outstretched hand while waving towards your reflection.
8. Please continue to wave at your reflection while looking at the palm of the same hand in the mirror.
9. Lower your hand to allow both arms to hang loosely down and look at your reflection in the mirror.
10. Please turn now 90 degrees to the left and look over your right shoulder at the reflection.
11. Turn back and take a relaxed look at your reflection in the mirror.
12. Please turn now 90 degrees to the right and look over your left shoulder at the reflection.
13. Turn back and take a relaxed look at your reflection in the mirror.
14. Please look down to your feet.
15. Look now at your feet in the mirror.
16. Please slowly raise both feet one after another so that your knees are alternating at hip level. Look at your legs from above.
17. Repeat the movement now for both feet while looking at your legs in the mirror.
18. Please keep standing on the marker and take a relaxed look at your reflection in the mirror.
19. Move now both arms in front of you so that your palms are facing downwards.
20. Move your arms in a circle in front of your body so that it is not too exhausting. Compare the movements of the arms in front of you with the movements in the mirror.

21. Lower your hand to allow both arms to hang loosely down and look at your reflection in the mirror.
22. Please stretch out your arms to the left and right so that your palms are facing downwards. Move your hips alternately to the left and right while looking at your reflection in the mirror.
23. Lower your arms again and look at your reflection in a relaxed manner. The movement tasks have been finished now.

A.4 Preference Questions

The preference questions on the similarity, fidelity, plausibility, and suitability of the avatars for self-presentation were asked in German and have only been translated into English for publication.

Questions:

1. Similarity: Which of the three avatars looks most similar to you?
2. Fidelity: Which of the three avatars feels to you as if it best represents your real body in the virtual environment?
3. Plausibility: Which of the three avatars seems most plausible to you to represent you in a virtual environment?
4. Suitability: Which of the three avatars would you choose to represent you in a virtual environment?

A.5 Interview Protocols

The interviews were conducted in German and the protocols have only been translated into English for publication.

Interview protocol for scanning participants:

1. In the following, I would like to ask you a few more in-depth questions about your impressions and experiences when carrying out the body scan with the smartphone app. Our primary aim is to gain a better understanding of the usability of the app and to improve it.
2. Did you know the person you scanned with the smartphone app before?
   a) If yes: How did you know the person?
3. I would now like to ask you to reflect on the various steps you went through when using the smartphone app and to express your thoughts and impressions out loud. Please go through the process again and comment out loud on what you thought. Describe whether there were any problems during the process and how you felt about using it. Try to describe your thoughts and impressions during use in as much detail as possible.
4. I have a few more specific questions now.
5. How did you feel while using the smartphone app?

6. Have there any moments in the scan process that were particularly pleasant or unpleasant for you?
    a) If yes: What kind of moments and how did you feel?
7. Did you understand how the smartphone app works and what its purpose is?
    a) If no: What exactly was not understandable?
8. Were you able to understand the smartphone scan process and was it clear at every point what needed to be done?
    a) If no: What exactly was unclear?
9. Which features of the smartphone app did you find particularly useful?
10. Were there certain moments in the scan process that you particularly liked?
    a) If yes: What kind of moments?
11. Were there any situations while using the smartphone app that were particularly difficult or did not work as expected?
    a) If yes: What kind of situations?
12. Were there any processes in the scan process that were difficult for you?
    a) If yes: Which kind of processes?
13. Are there any functions where you think the app could be improved?
    a) If yes: Which functions exactly and what could be improved?
14. Have you identified certain aspects of the scan process that could be improved?
    a) If yes: Which processes exactly and what could be improved?
15. Are there any additional comments you would like to add about the smartphone app and its use?

Interview protocol for scanned participants:

1. In the following, I would like to ask you a few more in-depth questions about your impressions and experiences of the body scans. Our primary aim is to gain a better understanding of how the scan process is perceived and to improve it.
2. Have you ever had a body scan before?
    a) If yes: What type of body scan have you had before?
3. Did you know all the time during the body scans what you had to do?
    a) If no: At what point in the respective scan method were there ambiguities, and how did these manifest themselves?
4. How did you perceive the smartphone app scan process compared to the scan process of the expert system?
    a) What was the reason for this?
    b) Was the smartphone app scan process more pleasant or less pleasant than the expert scan process?
5. Did you know the person who scanned you with the smartphone app before?
    a) If yes: How did you know the person?

6. Did the person performing the scan have any influence on how comfortable or uncomfortable you felt during the scan?
    a) If yes: For what reason?
    b) If yes: Was the gender of the person performing the scan decisive for your feelings?
7. Would you have felt differently about the app scan process if it had been carried out by the person performing the expert system or by another expert?
    a) If yes: Why would it have felt different?
8. What could the person performing the scan have done to make you feel more comfortable during the app scan process?
9. Would you choose to have a body scan again?
    a) If yes: Which scan process/method would you prefer and why?
    b) If no: Why not?
10. If you could change something in the respective scan processes, what would it be?
11. Which scanning method did you prefer today?

B APPENDIX

In Figures B.1 – B.11, we show results from iHuman [Pau+25], RMAvatar [Pen+25], our initial reconstruction system Avatars for the Masses (A4M, see Chapter 5), and our improved approach (see Chapter 6) in training poses. This means the subject is rendered as it was captured. Subjects 2, 8, and 11 asked to remain anonymous.

Figures B.12 – B.22 show results from iHuman, RMAvatar, our initial reconstruction system A4M, and our improved method in novel poses, that were not seen during capture/training.

Figures B.23 and B.24 show a female subject with loose, curly hair and a long skirt, and a male subject with long hair. These are examples of worst-case conditions for the reconstruction with our approach. Loose hair in windy conditions is difficult to reconstruct, as can be seen in Figure B.25 (see also the corresponding avatar in Figure B.42). While the point cloud captures the face perfectly, the hair is almost not present. In contrast to Gaussian-based approaches, the training image reconstruction appears considerably worse. In novel pose, iHuman and RMAvatar lose all detail and become blurry, while A4M and the improved approach keep a consistent quality. Animation without information about the clothing itself leads to artifacts, as the template is modeled wearing minimal clothing.

In Figures B.26 – B.44 we show more generated avatars from four directions, including previous subjects and additional ones.

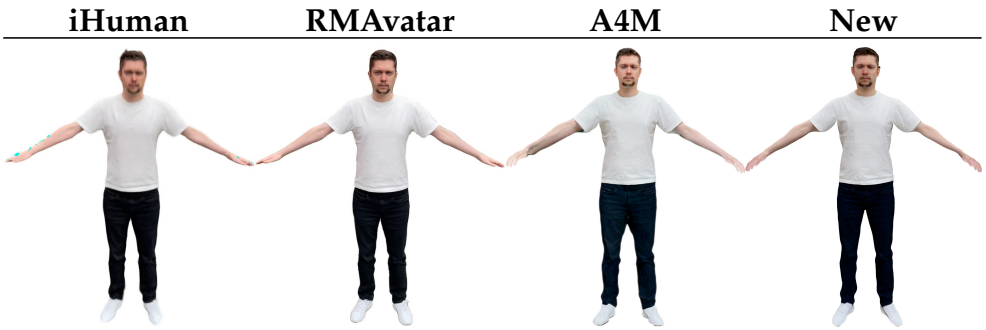


Figure B.1: Subject 1 in training pose

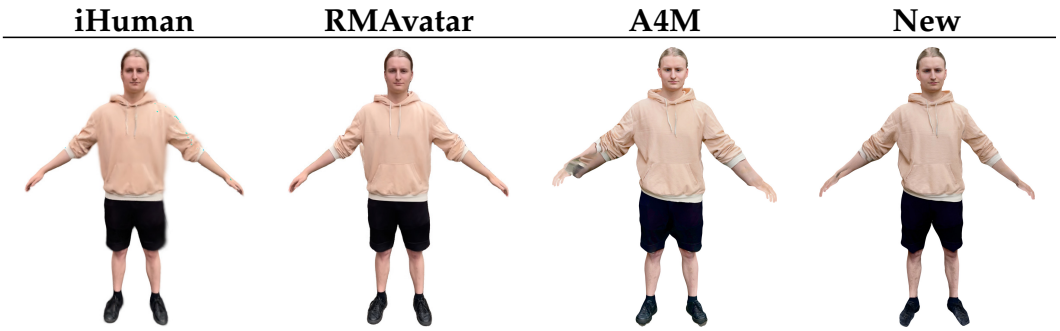


Figure B.2: Subject 2 in training pose

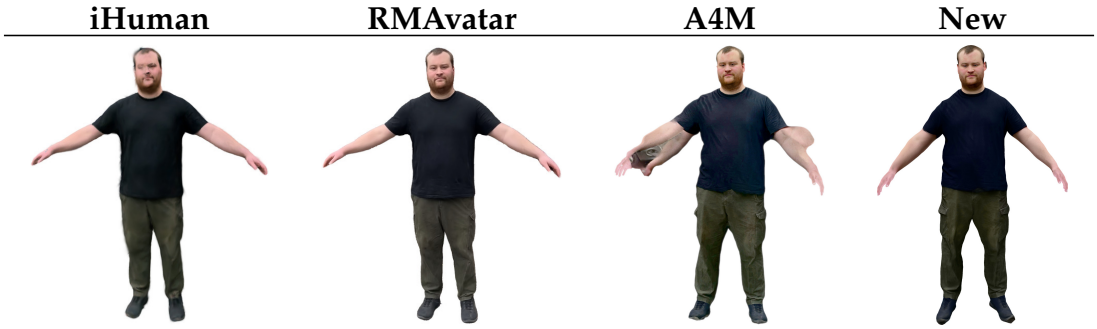


Figure B.3: Subject 3 in training pose

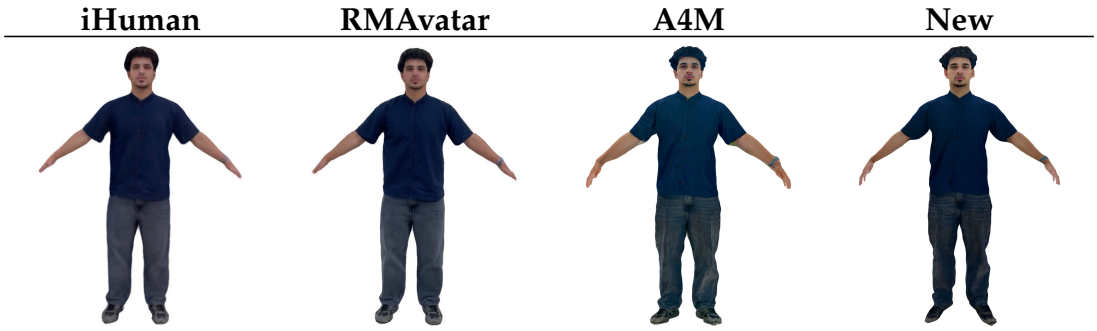


Figure B.4: Subject 4 in training pose

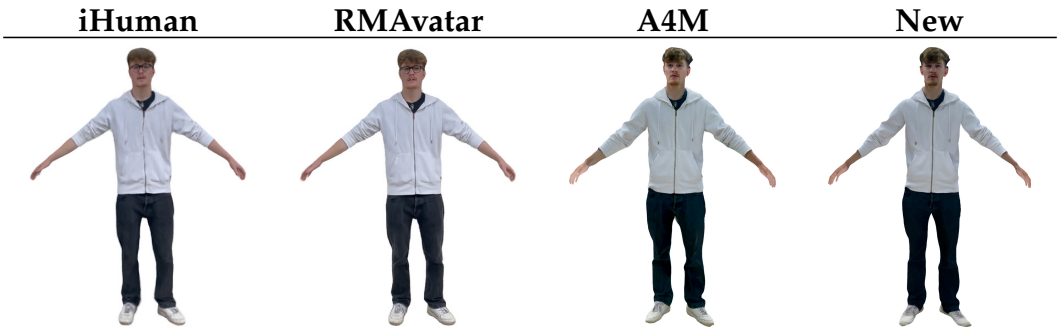


Figure B.5: Subject 5 in training pose

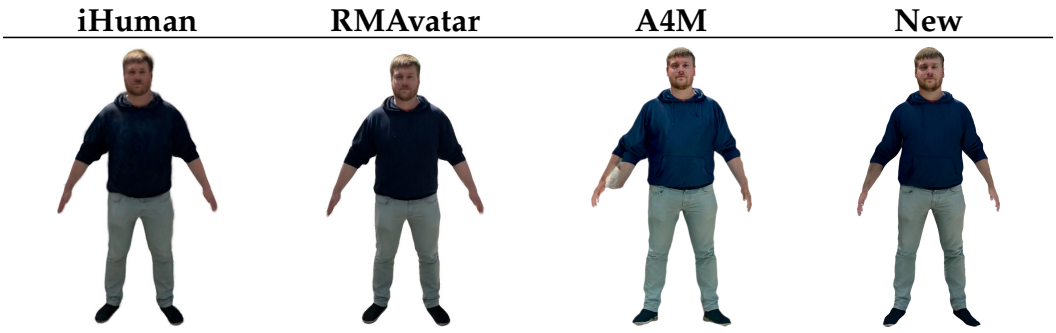


Figure B.6: Subject 6 in training pose

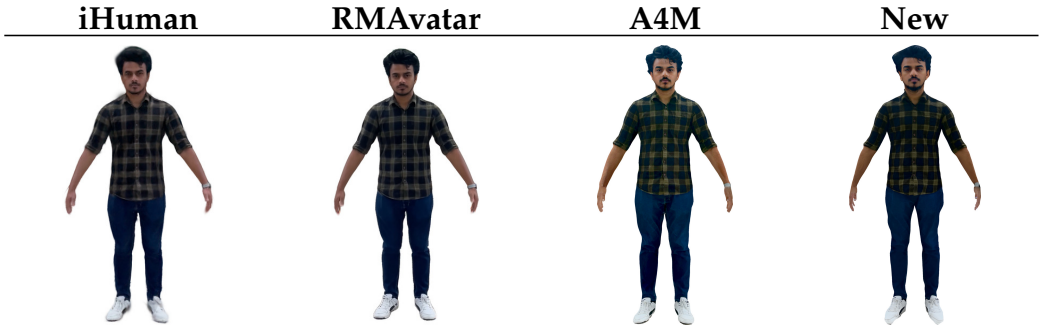


Figure B.7: Subject 7 in training pose

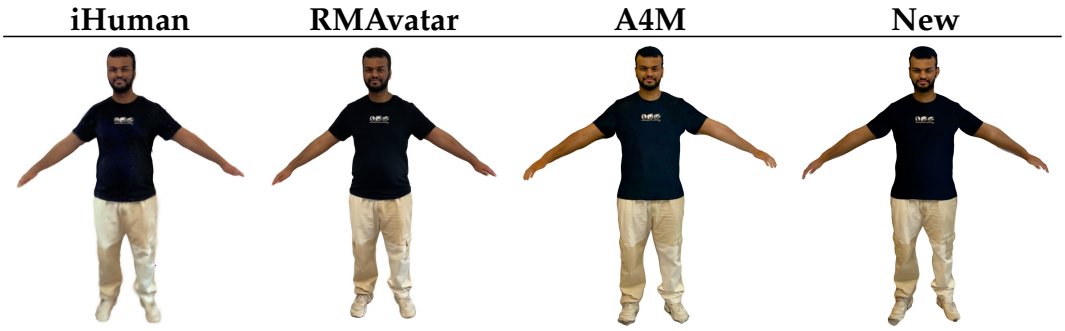


Figure B.8: Subject 8 in training pose

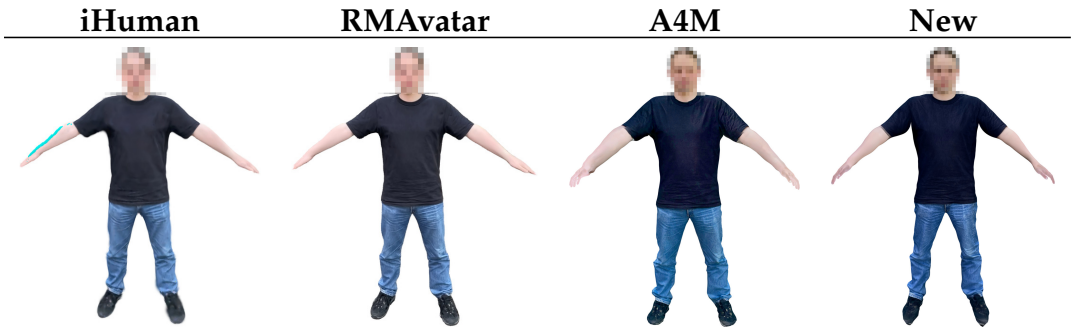


Figure B.9: Subject 9 in training pose

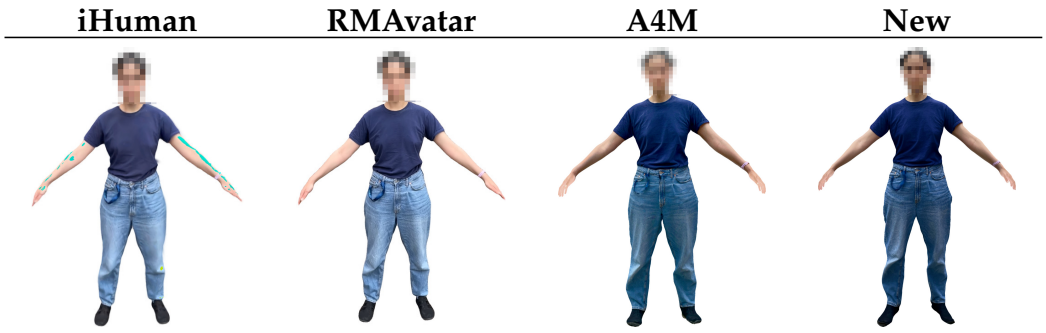


Figure B.10: Subject 10 in training pose

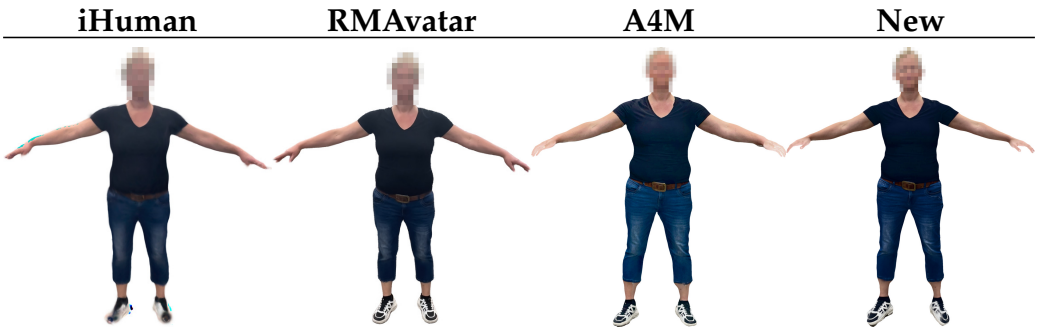


Figure B.11: Subject 11 in training pose

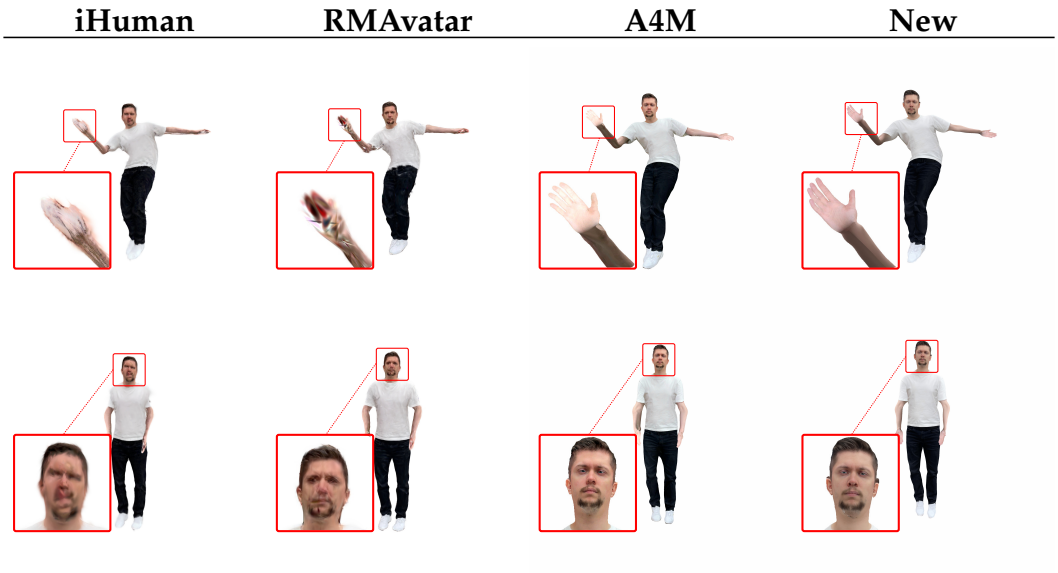


Figure B.12: Subject 1 in novel pose

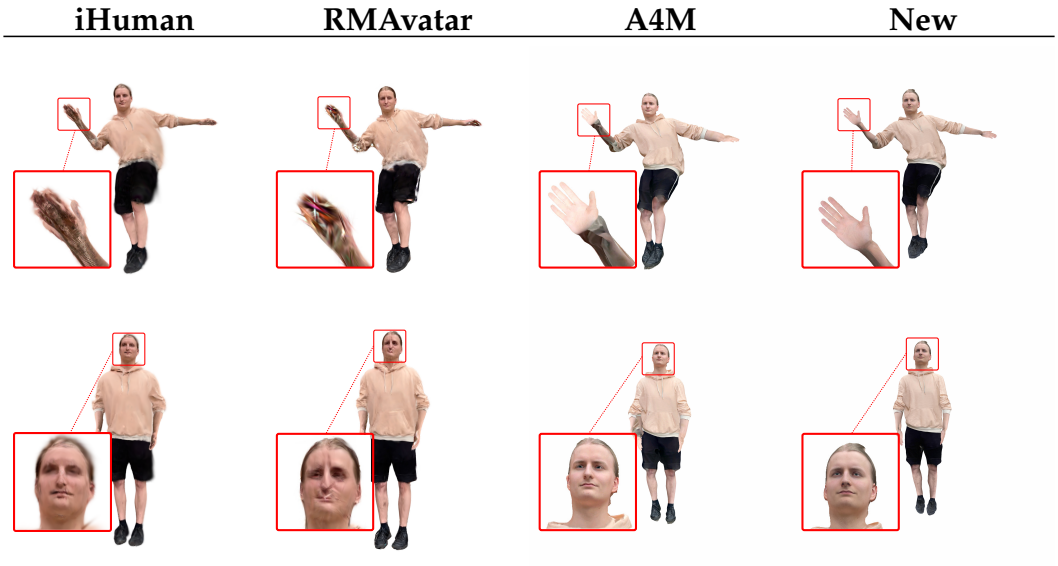


Figure B.13: Subject 2 in novel pose

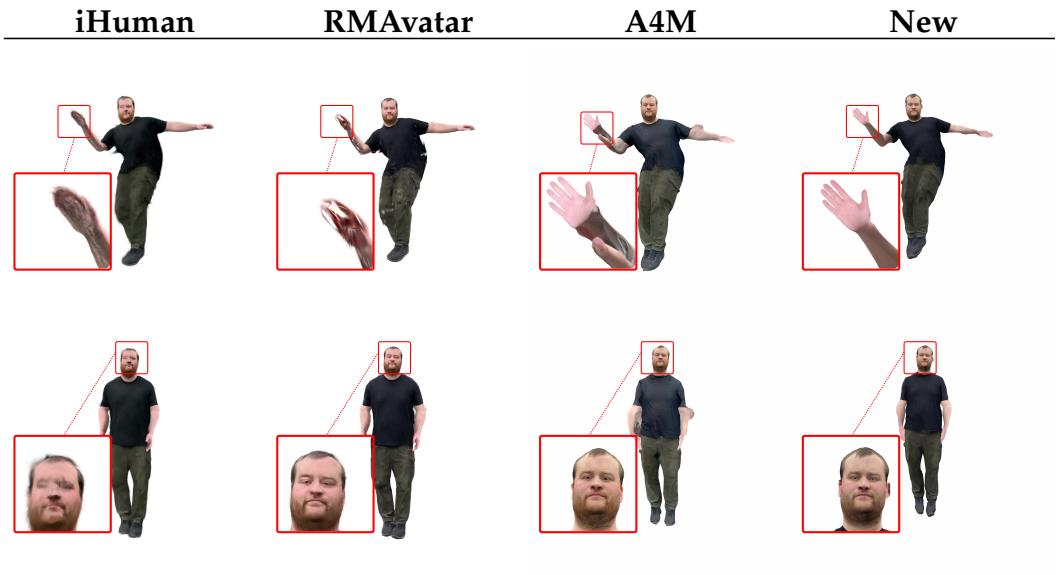


Figure B.14: Subject 3 in novel pose

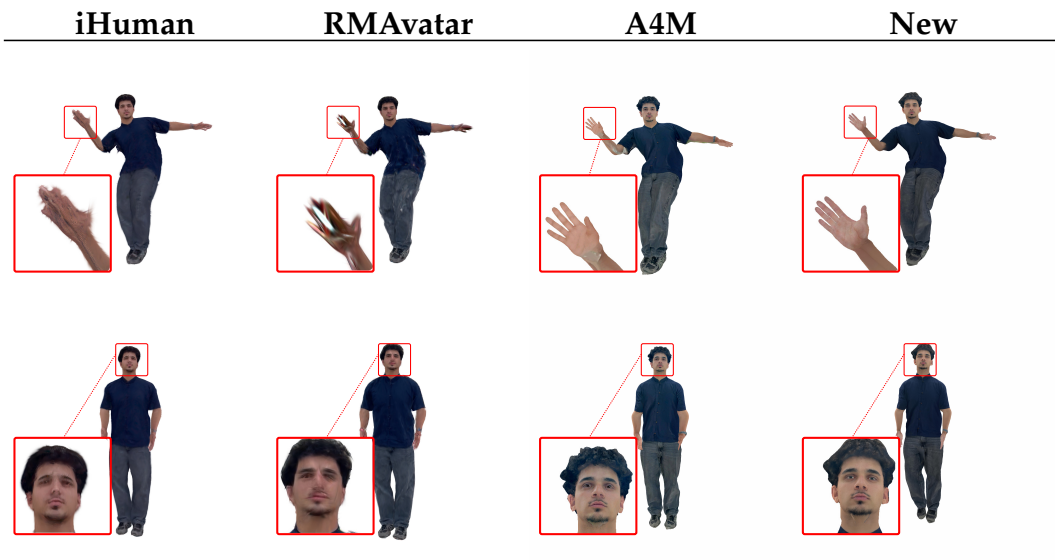


Figure B.15: Subject 4 in novel pose

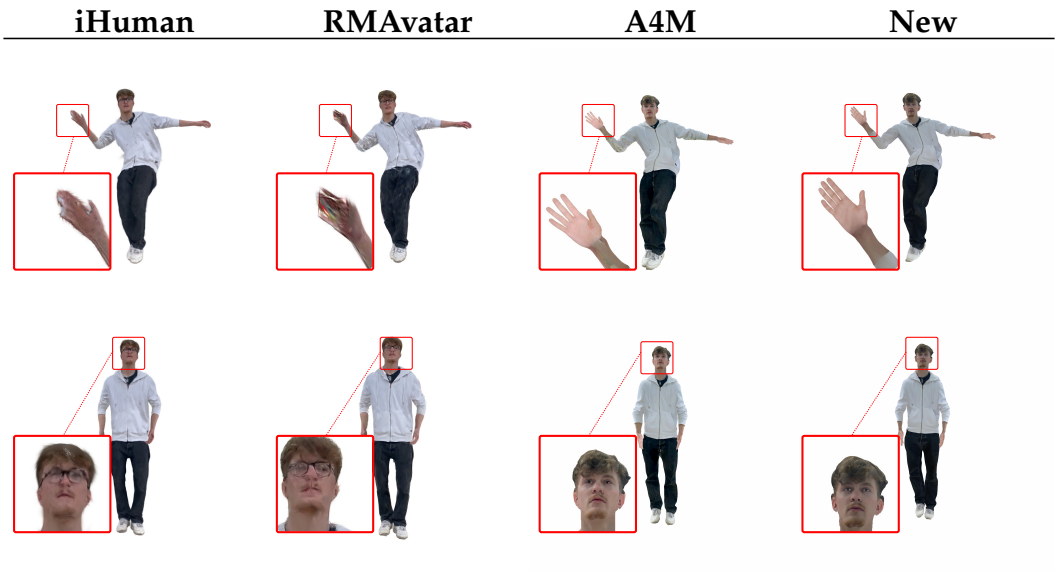


Figure B.16: Subject 5 in novel pose

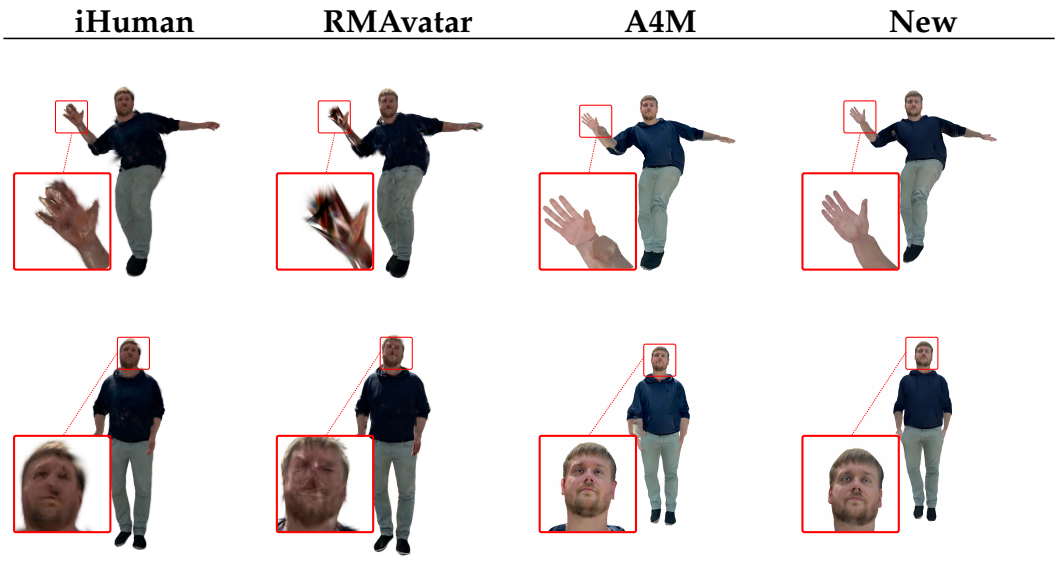


Figure B.17: Subject 6 in novel pose

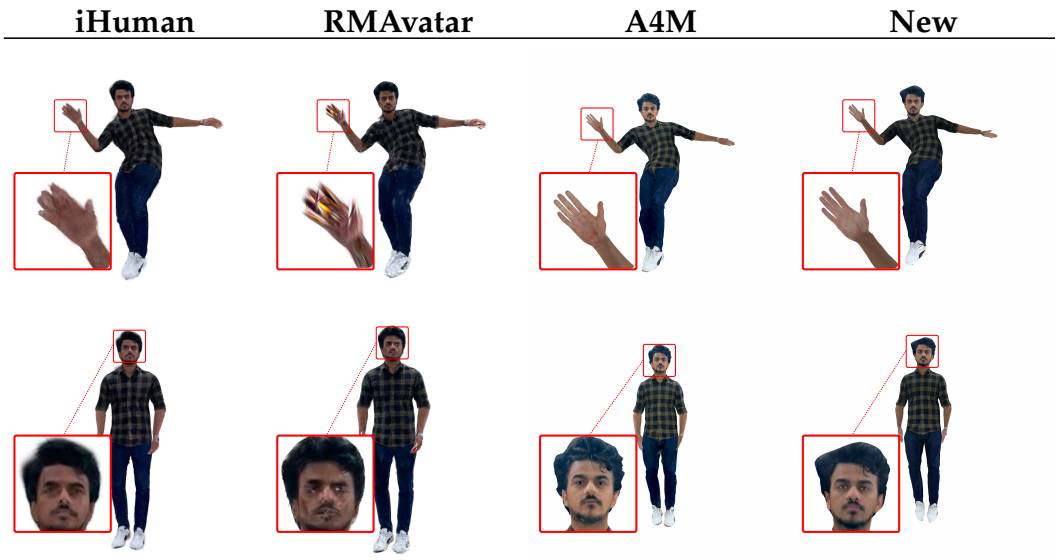


Figure B.18: Subject 7 in novel pose

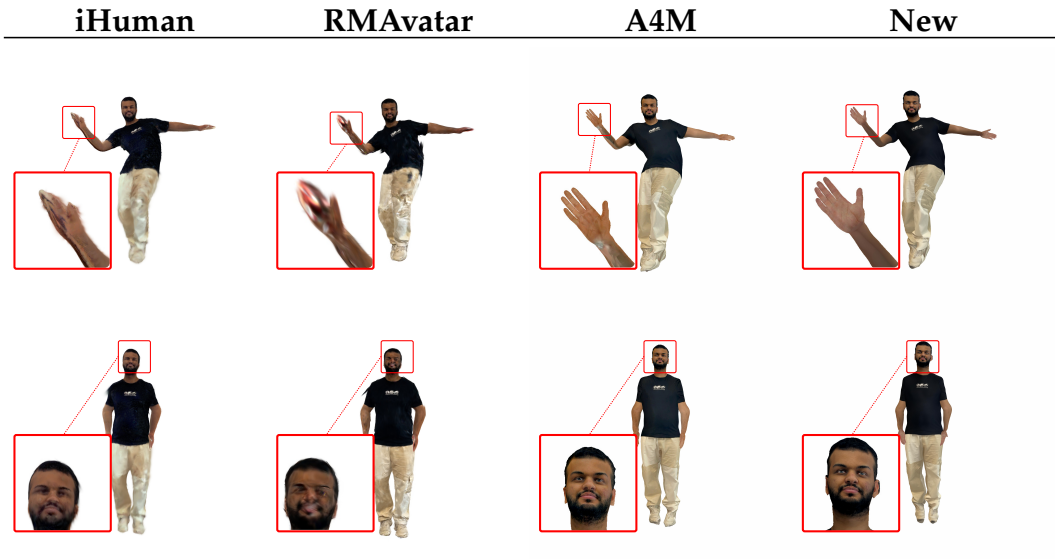


Figure B.19: Subject 8 in novel pose

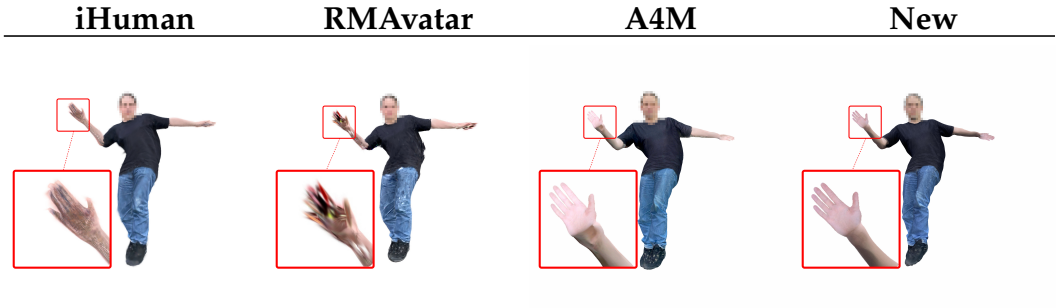


Figure B.20: Subject 9 in novel pose

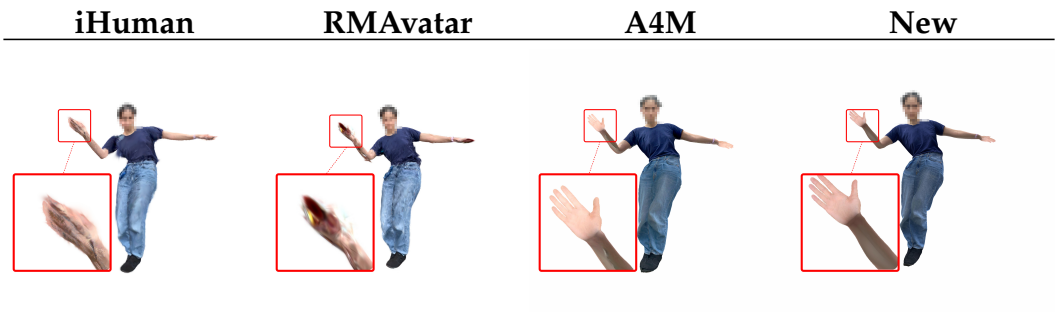


Figure B.21: Subject 10 in novel pose

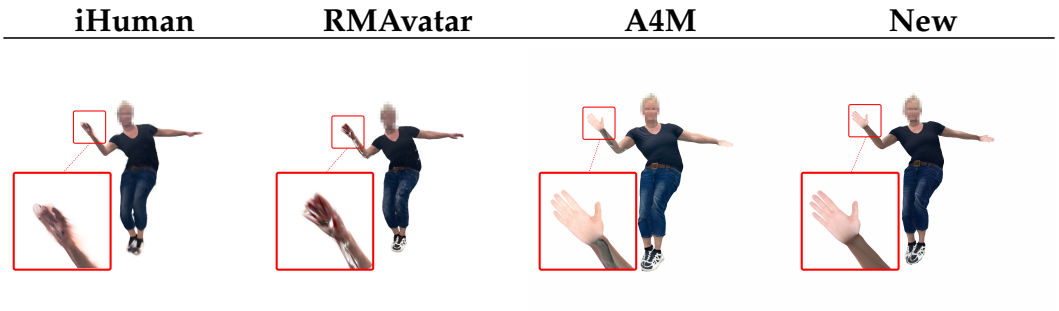


Figure B.22: Subject 11 in novel pose

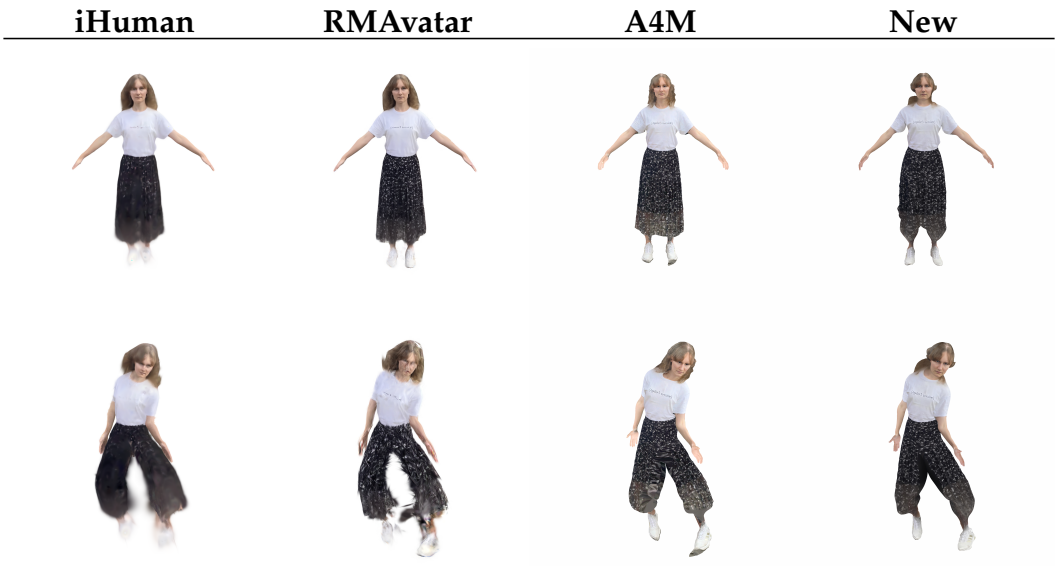


Figure B.23: Comparison of a female subject with loose, curly hair and a skirt. First in train-pose, then in novel-pose animation.

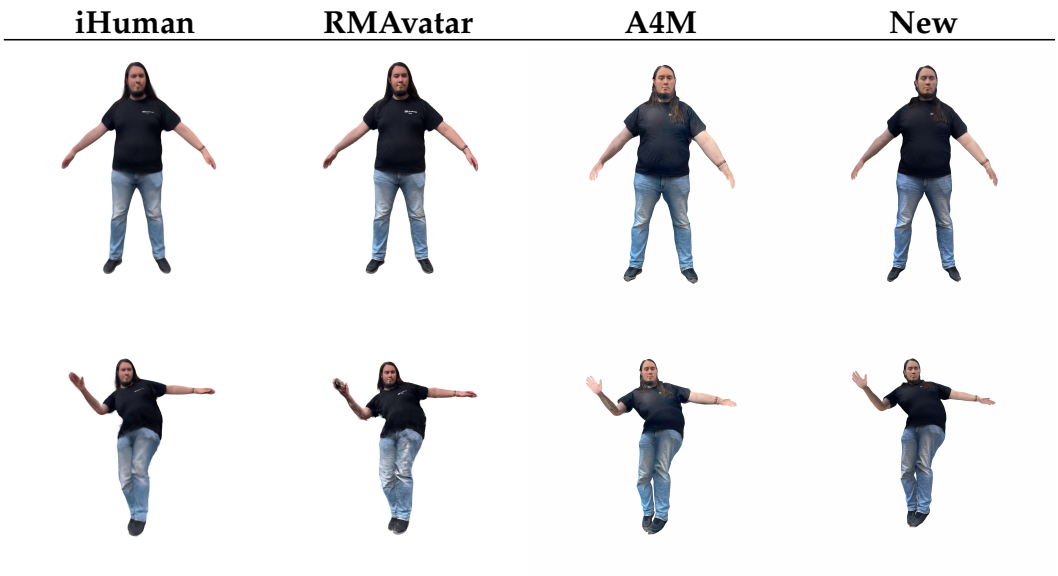


Figure B.24: Comparison of a male subject with long hair in training pose and novel poses.



Figure B.25: The point cloud generated by Agisoft Metashape [Agi25] for a female subject with loose hair. Due to windy conditions during the scan, the photogrammetry was unable to match hair features across the image sequence, resulting in only partial reconstructions.



Figure B.26: Subject 1



Figure B.27: Subject 2



Figure B.28: Subject 3

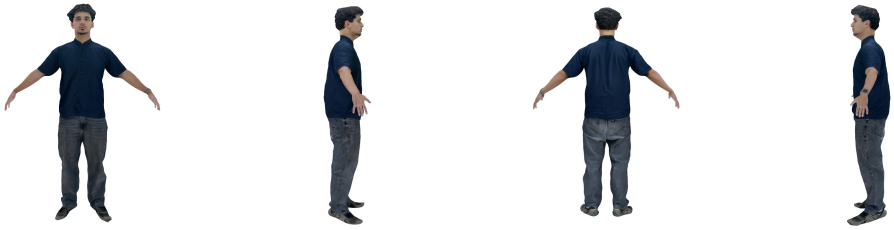


Figure B.29: Subject 4



Figure B.30: Subject 5



Figure B.31: Subject 6

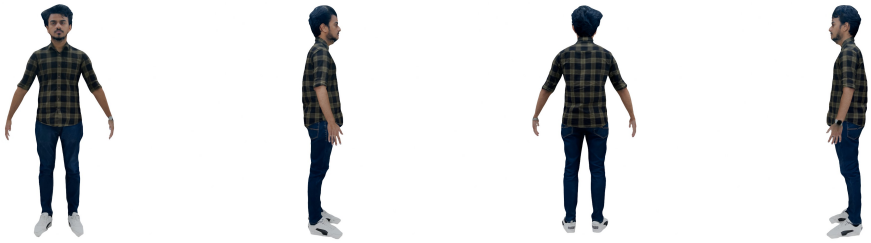


Figure B.32: Subject 7

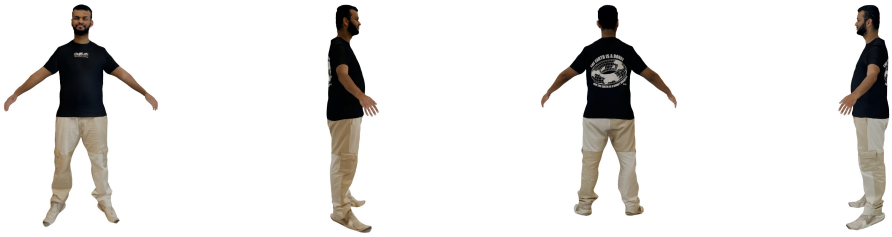


Figure B.33: Subject 8



Figure B.34: Subject 9

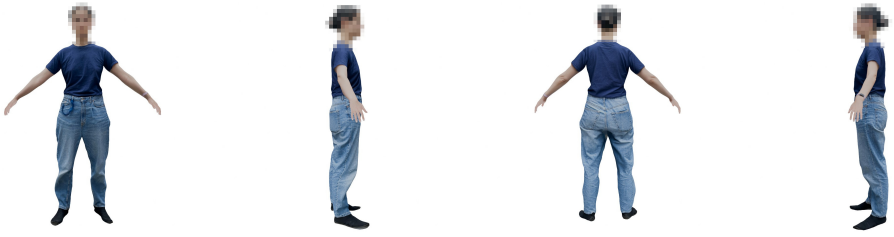


Figure B.35: Subject 10



Figure B.36: Subject 11



Figure B.37: Subject 12



Figure B.38: Subject 13



Figure B.39: Subject 14



Figure B.40: Subject 15



Figure B.41: Subject 16



Figure B.42: Additional Avatar: Female subject with loose hair and a long skirt.



Figure B.43: Additional Avatar: Female subject with a skirt.

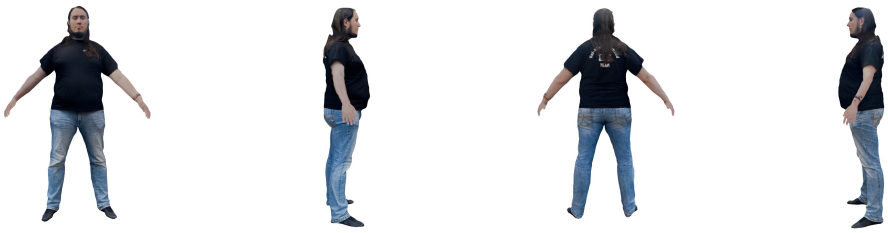


Figure B.44: Additional Avatar: Male subject with long hair.

C APPENDIX

In this appendix, we provide additional test results for our blendshape personalization method compared to example-based facial rigging by Li et al. [LWP10] and deformation transfer by Sumner et al. [SP04].

C.1 Subject 2

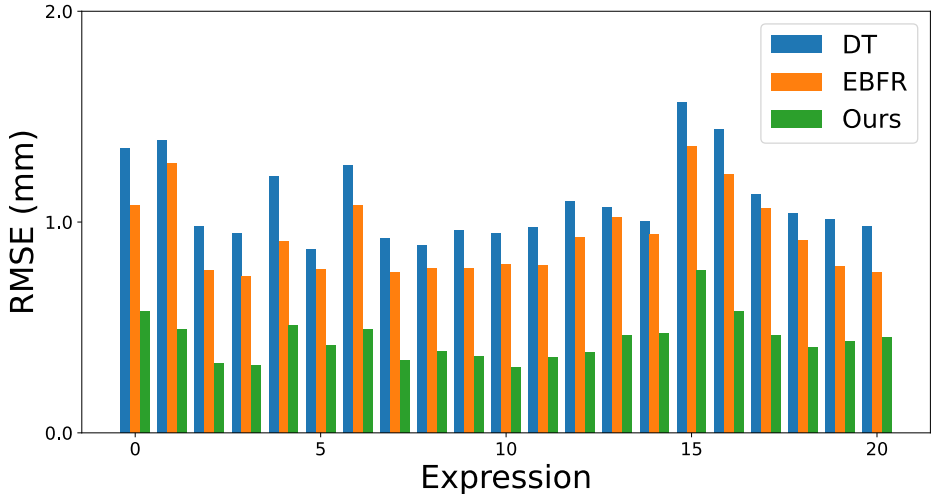


Figure C.1: Root-mean-square reconstruction error of deformation transfer (blue), example-based facial rigging (orange) and our method (green) for Subject 2.

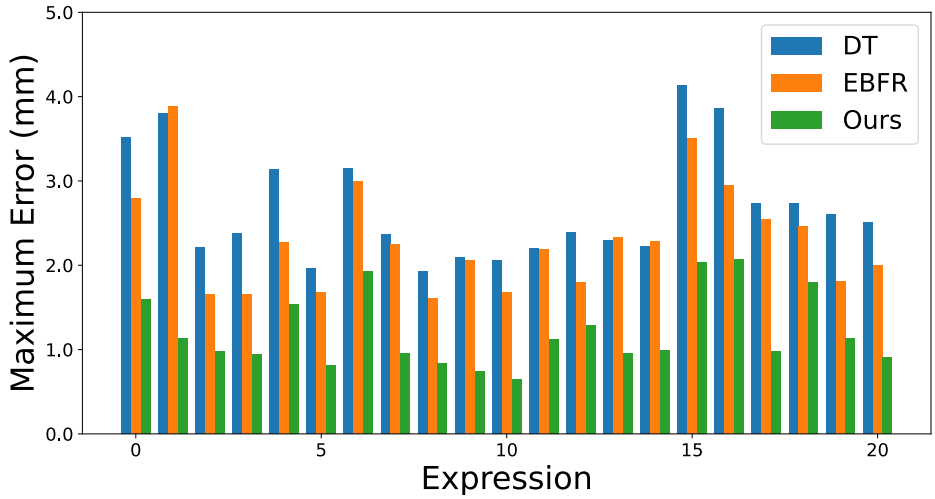


Figure C.2: Maximum reconstruction error of deformation transfer (blue), example-based facial rigging (orange) and our method (green) for Subject 2.

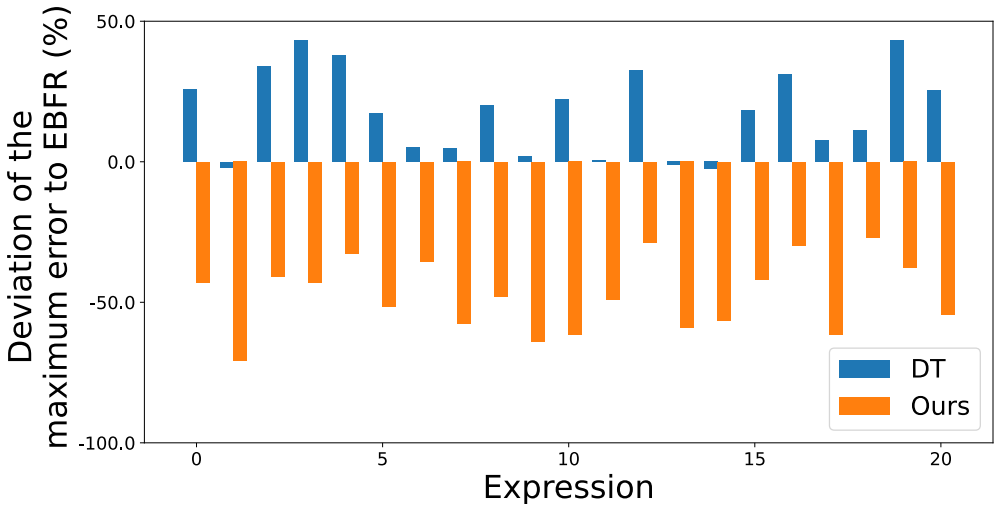


Figure C.3: Deviation of the maximum reconstruction errors from deformation transfer (blue) and our method (orange) to the errors produced by example-based facial rigging for Subject 2.

C.2 Subject 3

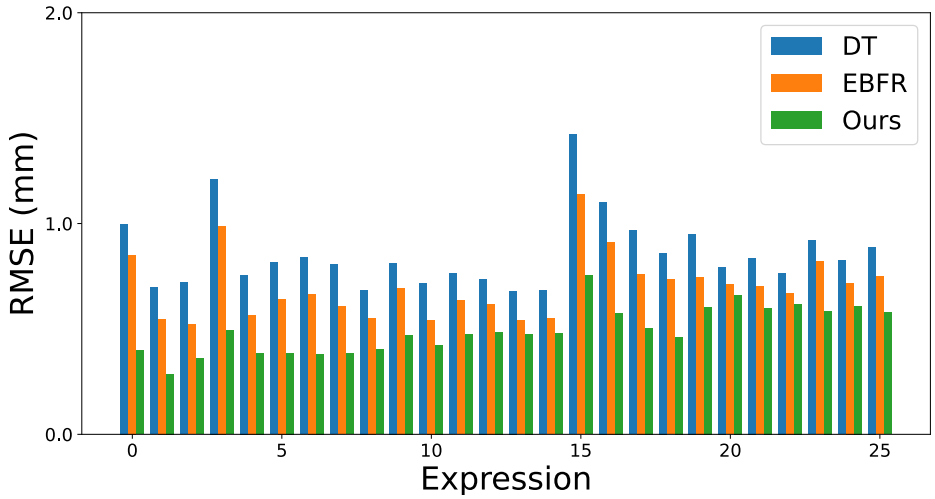


Figure C.4: Root-mean-square reconstruction error of deformation transfer (blue), example-based facial rigging (orange) and our method (green) for Subject 3.

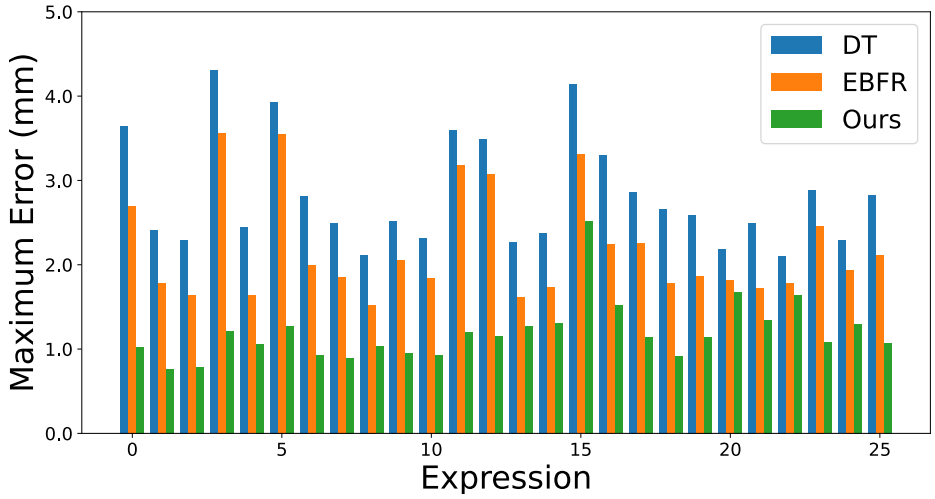


Figure C.5: Maximum reconstruction error of deformation transfer (blue), example-based facial rigging (orange) and our method (green) for Subject 3.

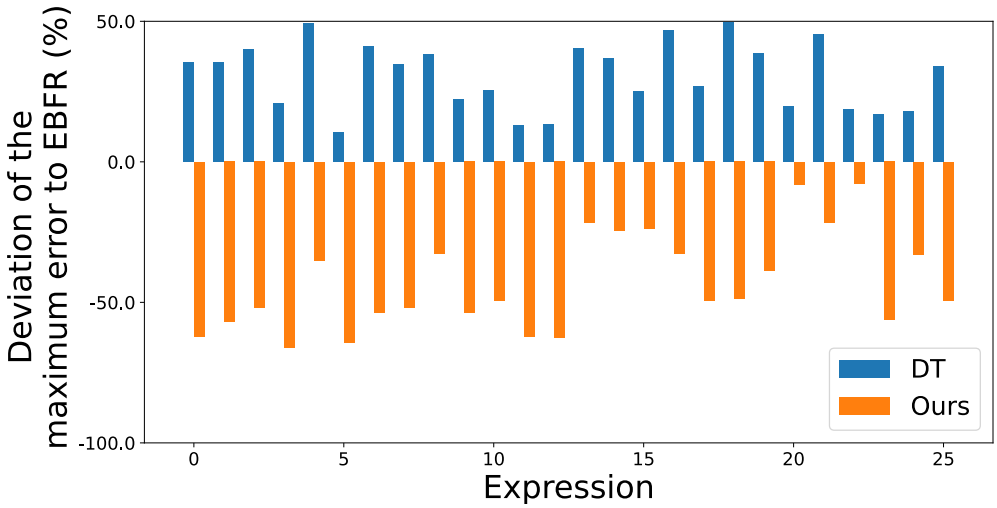


Figure C.6: Deviation of the maximum reconstruction errors from deformation transfer (blue) and our method (orange) to the errors produced by example-based facial rigging for Subject 3.

C.3 Subject 4

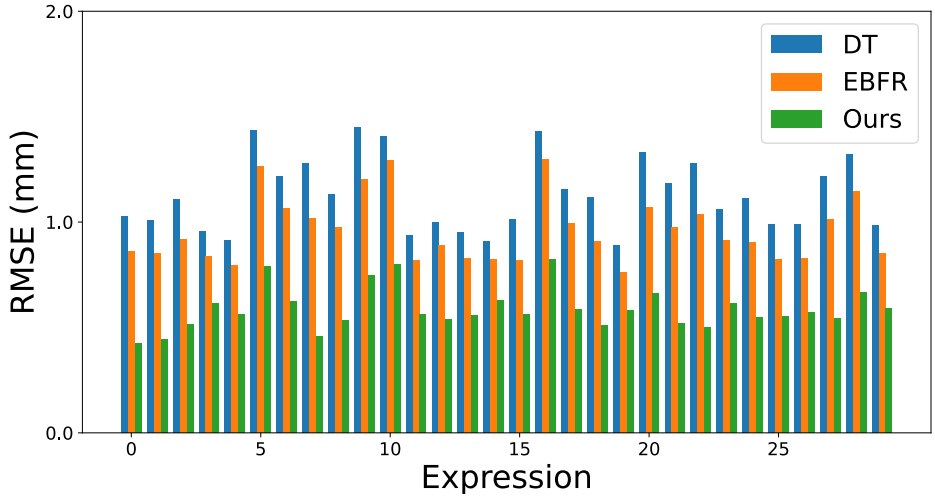


Figure C.7: Root-mean-square reconstruction error of deformation transfer (blue), example-based facial rigging (orange) and our method (green) for Subject 4.

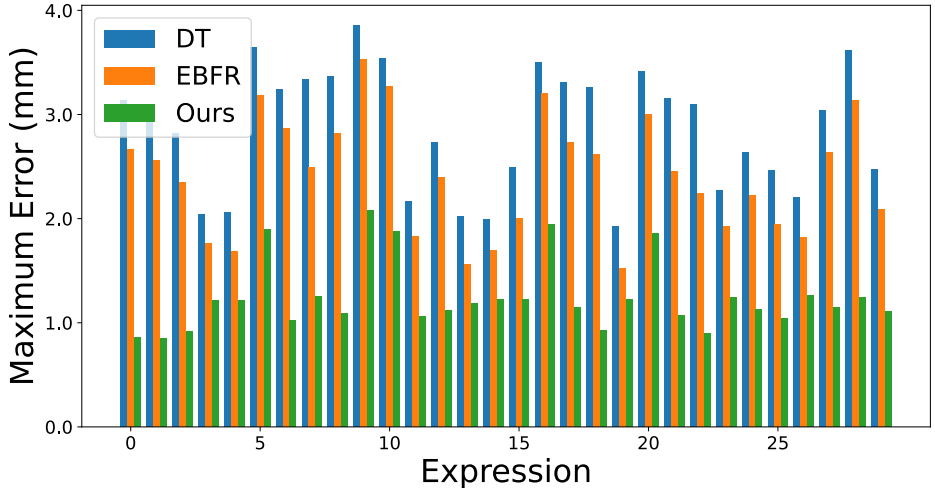


Figure C.8: Maximum reconstruction error of deformation transfer (blue), example-based facial rigging (orange) and our method (green) for Subject 4.

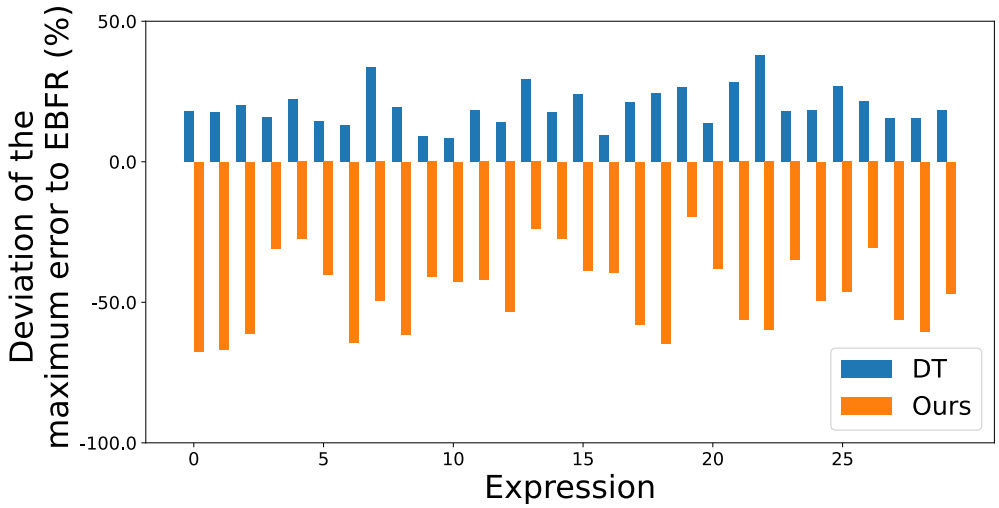


Figure C.9: Deviation of the maximum reconstruction errors from deformation transfer (blue) and our method (orange) to the errors produced by example-based facial rigging for Subject 4.

TOOLS USED

Text refinement, including grammar and spelling corrections, was supported by Grammarly[1] and ChatGPT[2].

1 https://grammarly.com
2 https://chatgpt.com