Reconstructing, Transferring and Editing of 3D Surfaces

by

Donya GHAFOURZADEH

THESIS PRESENTED TO ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
IN PARTIAL FULFILLMENT FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
Ph.D.

MONTREAL, APRIL 02, 2021

ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

BOARD OF EXAMINERS

THIS THESIS HAS BEEN EVALUATED

BY THE FOLLOWING BOARD OF EXAMINERS

Mr. Eric Paquette, Thesis supervisor
Department of Software & IT Engineering, École de technologie supérieure

Mr. Rachid Aissaoui, President of the board of examiners
Department of Systems Engineering, École de technologie supérieure

Mr. Carlos Vazquez, Member of the jury
Department of Software & IT Engineering, École de technologie supérieure

Mr. Mikhail Bessmeltsev, External independent examiner
Department of Computer Science and Operations Research, Université de Montréal

THIS THESIS WAS PRESENTED AND DEFENDED

IN THE PRESENCE OF A BOARD OF EXAMINERS AND THE PUBLIC

ON MARCH 12, 2021

AT ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

ACKNOWLEDGEMENTS

I would like to take this opportunity to thank all the people whose assistance was a milestone in the completion of this thesis.

First and foremost, I wish to express my deepest gratitude to my thesis advisor, Prof. Eric Paquette for his insight, patience, motivation, enthusiasm, and immense knowledge. It is whole-heartedly appreciated that his great advice for my study proved monumental towards the success of this study. I would also like to pay my special regards to Professor Tiberiu Popa, for his scientific guidance, advice, and continuous support throughout my study. Besides my advisers, I wish to thank the panel members for their time and for the honor of having agreed to examine my work.

I must also acknowledge Autodesk, Ubisoft, and NSERC Mitacs for providing the funding for the research. My deep appreciation goes out to the research team members at Autodesk, Olivier Dionne and Martin de Lasa and Ubisoft, Andre Beauchamp, Adeline Aubame and olivier Pomarez. I would like to recognize the invaluable assistance that they all provided during my study. I thoroughly enjoyed working with them, and I learned a great deal from them, both academically and in everyday life. Finally, I wish to acknowledge the support and great love of my partner and my parents. I am deeply grateful to their understanding, and their encouragement. They kept me going on and this work would not have been possible without their input.

Reconstruction, transfert et édition de surfaces 3D

Donya GHAFOURZADEH

RÉSUMÉ

La création de personnages et de visages réalistes est fastidieuse. Cette thèse commence par traiter le transfert de la structure d'animation d'un personnage 3D à un autre. Nous transférons l'armature et les poids de sommets entre des personnages ayant la même topologie de maillage. Notre approche prend en entrée l'armature source (maillage, hiérarchie de joints et poids de sommets) ainsi que le maillage du personnage cible. Nous calculons la position et l'orientation des joints qui incrustent l'armature dans le maillage cible, tout en déterminant les poids qui associent le maillage à l'armature. Pour ce faire, nous calculons la correspondance entre les maillages source et cible. Cette correspondance sert à formuler le transfert en une minimisation d'énergie complémentée d'un filtrage. Nous démontrons notre approche sur une variété de personnages, topologies de maillages et morphologies.

Une opération fondamentale pour le transfert d'animation est d'établir une correspondance géométrique précise entre les deux personnages. Les méthodes de correspondances de l'état de l'art donnent des résultats globalement bons, mais entraînent souvent des imperfections localisées. En conséquence, nous proposons une nouvelle approche pour améliorer une correspondance donnée par raffinement local. L'approche procède en quatre phases : (i) création d'un ensemble épars de marqueurs dans les régions erronées; (ii) segmentation à basse distorsion par dilatation de région et aplatissement; (iii) optimisation de la déformation des segments pour aligner les marqueurs dans le domaine 2D; et (iv) agrégation des correspondances des différents segments pour mettre à jour la correspondance globale. De plus, nous proposons une nouvelle approche de déformation de maillage pour aligner les marqueurs. Nous ajustons progressivement les poids cotangents et appliquons la déformation de sorte à éviter l'inversion des triangles tout en maintenant une faible distorsion. Notre nouvelle approche, *Iterative Least Squares Conformal Mapping (ILSCM)*, surpasse les autres méthodes de déformation.

Finalement, nous proposons une approche pour construire des modèles de morphage 3D (3DMM) de visages avec un contrôle intuitif d'édition. Les méthodes actuelles de 3DMM manquent de contrôle. Nous créons un 3DMM en combinant des 3DMM locaux pour les yeux, le nez, la bouche, les oreilles et le masque facial. Nous proposons une nouvelle approche de sélection des meilleurs vecteurs propres des différents 3DMM locaux pour s'assurer que le 3DMM combiné est expressif et précis. Les contrôles exposés à l'utilisateur proviennent de la littérature en anthropométrie. D'un vaste ensemble de mesures anthropométriques, nous sélectionnons celles qui permettent de reconstruire avec précision notre base de données. Nous associons les mesures aux vecteurs propres des 3DMM par des matrices de correspondances calculées à partir d'une base de données de visages numérisés.

Mots-clés: infographie, animation, modélisation de formes, modèles de géométrie maillée

Reconstructing, Transferring and Editing of 3D Surfaces

Donya GHAFOURZADEH

ABSTRACT

The authoring of realistic 3D faces and making a 3D character ready for animation are time-consuming and tedious tasks. This thesis first addresses the problem of transferring the animation setup between 3D characters. We transfer skeletons and skinning weights between characters with distinct mesh topologies. Our pipeline takes as inputs a source character rig (mesh, hierarchy of joints, and skinning weights) and a target character mesh. We compute joint locations and orientations that embed the source skeleton in the target mesh, as well as skinning weights to bind the target geometry to the new skeleton. We first compute the geometric correspondence between source and target meshes. The resulting geometric correspondence is then used to formulate attribute transfer as an energy minimization and filtering problem. We demonstrate our approach on a variety of source and target bipedal characters, varying in mesh topology and morphology.

A fundamental task of transferring animation is to establish an accurate geometric correspondence between the two characters. State-of-the-art mapping methods that exist today are globally good, but imperfections often exist and are often localized to a specific region. Consequently, we propose a novel approach to improve a given surface mapping through local refinement. The approach receives an established mapping between two surfaces and follows four phases: (i) creation of a sparse set of landmarks in mismatching regions; (ii) segmentation with a low-distortion region-growing process based on flattening the segmented parts; (iii) optimization of the deformation of segmented parts to align the landmarks in the planar parameterization domain; and (iv) aggregation of the mappings from segments to update the surface mapping. In addition, we propose a new approach to deform the mesh in order to meet constraints (in our case, the landmark alignment of phase (iii)). We incrementally adjust the cotangent weights for the constraints and apply the deformation in a fashion that guarantees that the deformed mesh will be free of flipped faces and will have low conformal distortion. Our new deformation approach, Iterative Least Squares Conformal Mapping (ILSCM), outperforms other low-distortion deformation methods.

Lastly, we propose an approach to construct realistic 3D facial morphable models (3DMM) that allows an intuitive facial attribute editing workflow. Current face modeling methods using 3DMM suffer from a lack of local control. We thus create a 3DMM by combining local part-based 3DMM for the eyes, nose, mouth, ears, and facial mask regions. Our local PCA-based approach uses a novel method to select the best eigenvectors from the local 3DMM to ensure that the combined 3DMM is expressive, while allowing accurate reconstruction. The editing controls we provide to the user are intuitive as they are extracted from anthropometric measurements found in the literature. Out of a large set of possible anthropometric measurements, we filter those that have meaningful generative power given the face data set. We bind the measurements to the part-based 3DMM through mapping matrices derived from our data set of facial scans.

Keywords: computer graphics, animation, shape modeling, mesh geometry models

TABLE OF CONTENTS

LIST OF TABLES

Page

LIST OF FIGURES

Page

LIST OF ABBREVIATIONS

3DMM	3D Morphable Model
AAM	Active Appearance Model
BIM	Blended Intrinsic Maps
ILSCM	Iterative Least Squares Conformal Map
LBS	Linear Blend Skinning
LIM	Locally Injective Mappings
LSCM	Least Squares Conformal Map
PCA	Principal Component Analysis
SLIM	Scalable Locally Injective Mappings
SPLOCS	Sparse Localized Deformation Components
WA	Weighted Averages

PREFACE

A version of Chap. 2 was published in Computer Graphics Forum (Avril, Ghafourzadeh, Ramachandran, Fallahdoust, Ribet, Dionne, de Lasa & Paquette, 2016). My key contributions were transferring skin weights and improving the geometric correspondence approach. I have also carried out most of the relevant experiments, evaluations, as well as video assembly. I wrote the manuscript of my contributions as well.

A version of Chap. 3 was published in Graphics Interface (Ghafourzadeh, Ramachandran, Lasa, Popa & Paquette, 2020b). All of the ideas originated in the paper between myself, Eric Paquette, and Tiberiu Popa. I carried out most of the experiments, implementation, evaluations, as well as video assembly. Srinivasan Ramachandran carried out some experiments in Sec. 3.2 as well as helped for preparing part of the video. I wrote the initial version of the paper and created all the figures. Afterwards, I have updated the paper according to Eric Paquette and Tiberiu Popa's comments.

A version of Chap. 4 was published in Graphics Interface (Ghafourzadeh, Rahgoshay, Fallahdoust, Beauchamp, Aubame, Popa & Paquette, 2020a). All of the ideas originated in the paper between myself, Eric Paquette, Tiberiu Popa, Andre Beauchamp, and Adeline Aubame. I carried out most of the experiments, implementation, evaluations, as well as video assembly. Cyrus Rahgoshay contributed to Sec. 4.5.4 and helped with its implementation. Sahel Fallahdoust carried out some experiments in Sec. 4.5.4 as well as helped for preparing part of the video. I wrote the initial version of the paper and created all the figures. Afterwards, I have updated the paper according to Eric Paquette and Tiberiu Popa's comments.

INTRODUCTION

Livening up a character mesh to get a compelling animation remains a time-consuming, fastidious, and labor intensive task, whether it is for visual effects in movies, for video games or for virtual reality. Creating a complete character animation involves at minimum creating the mesh of the character and setting up the skeleton and skin weights. The skeleton should be activated underneath the mesh representing their skin, clothes, and accessories. Joint placement, painting of a realistic skin weight, and many manual corrections of animation artifacts are compulsory stages that animators need to go through to define a complete animation setup for any new character.

Some recent techniques simplify the animation process and minimize the computational costs by reusing the animation setup from existing characters instead of creating them from scratch. This has the additional benefit of preserving authoring details in the source skeletons and skinning weights, limiting unwanted deformation artifacts, and easing animation reuse. Some approaches (Poirier & Paquette, 2009; Seo, Seol, Wi, Kim & Noh, 2010; Xu, Zhou, Kalogerakis, Landreth & Singh, 2020) have focused solely on skeleton transfer, requiring manual skinning weight painting before characters are ready for animation. To reduce the need for manual intervention, several methods have been proposed that seek to automate skinning weight selection (Baran & Popović, 2007; Kavan & Sorkine, 2012; Dionne & de Lasa, 2014). Although such approaches can provide good initial skinning weights, they cannot guarantee perfect results in all cases; artists may still need to manually adjust the weights to satisfy animation requirements. Alternative approaches Dicko, Liu, Gilles, Kavan, Faure, Palombi & Cani (2013); Miller, Arikan & Fussell (2011) rely on a template character or a database of previously authored results which restricts the application domain. This dissertation is part of the “Animation setup transfer for 3D character ” (Avril *et al.*, 2016) paper which is a general method for transferring skeletons and skinning weights between characters with distinct mesh topologies. It allows the artist to provide his own skeleton structure and skinning weights, instead of being constrained to

those found in a template character or in a database of characters. From these inputs, it computes joint locations and orientations that embed the source skeleton in the target mesh, as well as skinning weights to bind the target geometry to the new skeleton. The proposed approach allows for the preservation and transfer from the precise work of the artist who positioned the joints and painted the weights on the source character. It works for characters with meshes differing in topologies. Additionally, it can be applied to source and target characters within a broad range of morphologies. The approach is based on first establishing a geometric correspondence between a source and a target character mesh. The more accurate and precise this geometric correspondence is, the more accurate the skeleton and skinning weight transferring will be.

The geometric correspondence aims to align the corresponding features of the meshes as closely as possible, which is a fundamental problem with applications in attribute transfer (Kraevoy & Sheffer, 2004; Avril *et al.*, 2016), morphing (Alexa, 2002), shape database analysis (Allen, Curless & Popović, 2003), and even deep learning on geometries (Bronstein, Bruna, LeCun, Szlam & Vandergheynst, 2017).

In general there are two ways to handle the problem of finding a geometric correspondence. The first type of methods consists in finding the correspondence by spectral analysis such as functional maps (Ovsjanikov, Ben-Chen, Solomon, Butscher & Guibas, 2012; Ezuz & Ben-Chen, 2017; Gehre, Bronstein, Kobbelt & Solomon, 2018) or by automatically detecting the sparse correspondences (Jain, Zhang & Van Kaick, 2007; Kim, Lipman, Chen & Funkhouser, 2010; Kim, Lipman & Funkhouser, 2011; Liu, Kim & Funkhouser, 2012). This type of methods is fully automatic, but does not allow control over the final mapping, which can sometimes exhibit mismatches at certain semantic areas. The second type of methods relies on the user to define sparse correspondences, and thus control the final mapping to some extent (Kraevoy & Sheffer, 2004; Sumner & Popović, 2004; Zell & Botsch, 2013).

This dissertation is conducted in collaboration through the paper "Joint planar parameterization of segmented parts and cage deformation for dense correspondence" (Ramachandran, Ghafourzadeh, de Lasa, Popa & Paquette, 2018) concentrating on a dual flattening geometric correspondence method by relying on a novel cage deformation method. The key contributions of this dissertation is to discover that using larger patches still results in a bijective mapping. Using larger patches makes the approach more efficient and reduces the amount of manual work required from the user. Moreover, it is capable of handling a wide range of surfaces and does not impose too many constraints on the choice of surfaces, handling non-isometric pairs and pairs in different poses. The results show that the proposed approach is superior to current state-of-the-art methods. The geometric correspondence results is often good overall, but some specific semantic features, such as articulations and facial features, may remain misaligned. In fact, no single method results in a perfect mapping in every case. These imperfections of the final result are often unacceptable in a production setting where the artist needs a high degree of precision and control over the final result, and will often sacrifice automation of a method for higher control. The typical way to improve the surface mapping results is that the user iteratively inserts some landmarks and then solve for the mapping globally. However, since these problems are typically localized to a specific region, a local solution that does not change the mapping globally is preferred in order to make sure that the method does not introduce issues elsewhere on the map. While a lot of research has been done on creating correspondences between 3D objects, comparatively fewer methods have been proposed on correspondence editing (Ezuz & Ben-Chen, 2017; Vestner, Litman, Rodolà, Bronstein & Cremers, 2017; Panozzo *et al.*, 2013). Most state-of-the-art methods improve the mapping globally which makes it hard for the user to add small adjustments without risking modifying areas that should not be touched. Furthermore, some methods face significant limitations such as being constrained to isometric deformations and requiring compatible meshes on both surfaces.

This dissertation proposes a geometric correspondence editing approach providing local and precise control over the map adjustments. Our paper "Local Editing of Cross-Surface Mappings with Iterative Least Squares Conformal Maps" (Ghafourzadeh *et al.*, 2020b) improves the mapping in the related regions without causing a degradation of the overall mapping. It allows the user to restrict editing to a specific region by adding as few or as many landmarks as necessary to achieve a desired result. For each such region, it automatically extracts a patch in order to localize the changes in the mapping, and then flattens on a common planar domain. The mapping is improved based on a novel 2D deformation optimization that steers the landmarks toward correspondence while limiting distortion and having theoretical guarantees to maintain the local injectivity of the map.

As for reusing animation skeleton and skinning weights, is desirable to use the facial features from a previously made face model for generating a new model. Another contribution of this dissertation is to reuse available data through deformation instead of creating them from scratch. Authoring realistic 3D faces is particularly a labor-intensive and time-consuming task given the intricate details found in the eyes, nose, mouth, and ears. Accordingly, creating face models with a minimal amount of labor is a big challenge in face modeling.

Some recent techniques aided modeling by deforming existing data instead of creating them from scratch. Many methods (Blanz & Vetter, 1999; Tena, De la Torre & Matthews, 2011; Neumann, Varanasi, Wenger, Wacker, Magnor & Theobalt, 2013; Cao, Weng, Zhou, Tong & Zhou, 2014) use 3D morphable face models (3DMM) for animation (blend shapes), face capture, and face editing. 3DMMs are typically constructed by computing a Principal Component Analysis (PCA) on a data set of scans sharing the same mesh topology. New 3D faces are generated by changing the relative weights of the individual eigenvectors. These methods are popular due to the simplicity and efficiency of the approach, but suffer from two fundamental limitations: they impose global control to the new generated meshes, making it impossible to edit a localized

region of the face, and the control mechanism is very unintuitive. Some methods compute localized 3DMM, but they focus on facial animation instead of face modeling. Facial animation database is a collection of facial expressions of a range of emotions of an identity. These methods assume large, yet localized deformations caused by facial expressions. While in this instance, we consider a database of identities which each face is completely different from the others. This dissertation proposes an approach to construct realistic 3DMMs (Ghafourzadeh *et al.*, 2020a) using an easy-to-use tool for the creation and manipulation of 3D faces. It eases this process by providing high-level controls, such as anthropometric measurements, to edit available human-like character heads. The proposed approach increases the controllability of the faces by segmenting them into independent subregions and selecting the most dominant eigenvectors per part. Furthermore, it proposes a measurement selection technique to bind the essential measurements to the 3DMM eigenvectors.

CHAPTER 1

LITERATURE REVIEW

This chapter discusses, geometric correspondence and geometric modeling. The first part presents a review of existing methods which are currently used for animation transfer. In a second part, it discusses different approaches to establish mappings and then how to edit the mapping between 3D shapes. In the last part, the related work to the methods of morphable 3D face model is discussed here. Finally, this chapter makes a comparison of these approaches and tries to outline their strengths and weaknesses.

1.1 Animation Transfer

In this section, we provide an overview of methods related to character animation including skeleton, and skinning weights, as well as their creation and retargeting. It should be noted that an in-depth discussion of skeleton retargeting research is beyond the scope of this dissertation.

1.1.1 Skeleton and Skin Weights

Animation pipelines require iteration. Even late into a production cycle, edits to the character's geometry, skeleton, or deformations may be needed to meet evolving directorial requirements. Many production environments place restrictions on modeling operations that can be performed once deformations have been specified. Failure to do so could result in expensive mistakes and the need to redo work. Several papers ease the task of creating the character skeleton or the skin weights. Some methods derive skinning weights solely from a skeleton and mesh (Wareham & Lasenby, 2008; Dionne & de Lasa, 2014). Other solutions require a set of example poses to compute the skeletal structure (De Aguiar, Theobalt, Thrun & Seidel, 2008; He, Xiao & Seah, 2009; Le & Deng, 2014) or to solve for the optimal skinning weights (Wang & Phillips, 2002; Li & Lu, 2011; Le & Deng, 2012).

Au, Tai, Chu, Cohen-Or & Lee (2008) use Laplacian smoothing to extract a skeleton from a mesh. This method creates the skeleton for fine geometry (more than 5K of vertices) and relies

on other techniques to generate the skinning weights. He *et al.* (2009) use several poses of a given mesh and apply a harmonic function on the given example poses to construct the skeleton-like Reeb graph. They calculate the initial location of the joints through examining the changes in the mean curvatures and then solve a constrained optimization problem to refine the joint locations. Le & Deng (2014) propose an example-based rigging approach which automatically creates Linear Blend Skinning (LBS) models including skeletal structure and skin weights. The main idea is an iterative rigging algorithm to update skinning weights, joint locations, as well as bone transformations. Its main limitation is the low computational efficiency and the artifacts from using the LBS model. Baran & Popović (2007) automatically embed a template skeleton and derive skinning weights for arbitrary meshes by formulating a linear system. They propose bone heat (heat map binding) to model the skin weights attaching to each bone as a heat diffusion system. Their approach relies on a distance smoothing operation in order to behave well and robustly. Additionally, other example-based methods are available to generate the skin weights (Wang & Phillips, 2002; Li & Lu, 2011; Le & Deng, 2012, 2014). Le & Deng (2014) use a given set of example poses of a character to generate the skin weights. The geodesic voxel binding method by Dionne & de Lasa (2014) computes binding weights for a given skeleton and mesh, independent of mesh resolution. They first voxelize the input geometry. The resulting voxelization is classified as bone, interior, boundary, and exterior voxels. Afterward, they compute the geodesic distance between each voxel falling on the bones of the skeleton and non-exterior voxels. This is repeated for all bones of the skeleton. At the end, they calculate the corresponding weight using a falloff function based on the computed geodesic distance.

To take full advantage of the skeletons and skinning weights of existing characters, there exist few methods which retarget both attributes. Allen *et al.* (2003) present a method for transferring the animation setup from one character to other scanned character meshes. Each joint is retargeted according to three markers on the meshes. It works for near-similar morphological bodies, but can result in sub-optimal joint positions for morphologies with large differences (e.g., a human vs. a gorilla). Moreover, as weights are transferred using a vertex-to-vertex correspondence between meshes, discontinuities and animation artifacts can arise when transferring attributes among meshes with different polygon densities. Anatomy Transfer (Dicko *et al.*, 2013) can

retarget several attributes, including skeleton and skinning weights. Nevertheless, the method cannot retarget user-provided skeletons and skinning weights as it relies on a template character which restricts the application domain. Frankenrigs (Miller *et al.*, 2011) proposes an animation setup retargeting method using a database of characters. It overcomes several limitations by finding, transferring, and blending weights from different fully-rigged characters. The main limitation lies in the creation of the database of valid rigged meshes, and so characters with unexpected limb shapes are not likely to pull a very good match from the database. Moreover, the user is constrained to the types from the database. RigNet (Xu *et al.*, 2020) proposes a deep learning approach for rigging and motion generation. It predicts a skeleton which matches the joint placement and topology. Afterwards, it generates skin weights for the predicted skeleton. RigNet (Xu *et al.*, 2020) needs a database of designed skeletons as supervision which restricts the application domain for general models.

To summarize, methods that create skeletons and skinning weights from scratch cannot provide the level of quality that are obtained by an artist. Some methods that retarget both skeleton and skinning weights use template characters instead of allowing the user to provide his own skeleton structure and skinning weights. Some methods (Dionne & de Lasa, 2014; Baran & Popović, 2007) can provide good initial skinning weights, but they cannot guarantee optimal results in all cases; manual adjustment of skinning weights may still be necessary to satisfy all artists' requirements. We propose an approach in our collaboration in the paper by Avril *et al.* (2016) to transfer skinning weights between characters based on the geometric correspondence and robust to differences in mesh resolution. Our approach (Chap. 2) outperforms the state-of-the-art methods.

1.2 Geometric Correspondence

Geometric correspondence aims to relate semantically similar surface components to one another. This fundamental problem plays a preliminary and also significant role in many computer graphics fields such as morphing, animation transfer, and correspondence estimation. Since finding correspondence between surfaces has been a long standing problem in computer

graphics, there exist different methods which the most relevant methods will be discussed here. For ease of reference, we classify these methods into those that deform a template surface until it fits the given surface in Sec. 1.2.1. Next, those that embed the input surfaces into spaces where finding the correspondences is easier in Sec. 1.2.2 and then we discuss the use of common parameterization domain in Sec. 1.2.3. Finally, we discuss the correspondence editing methods which improve correspondences based on the user demands in Sec. 1.2.4. The reader is referred to the surveys of Sahillioğlu (2020) and Van Kaick, Zhang, Hamarneh & Cohen-Or (2011) for a more exhaustive list of geometric correspondence methods.

1.2.1 Deformation in 3D Euclidean Space

To obtain dense correspondences between the surfaces, non-rigid registration methods deform the given surfaces until they match (Allen *et al.*, 2003; Huang, Adams, Wicke & Guibas, 2008; Li, Sumner & Pauly, 2008); however, most such methods are limited to near-isometric objects. Generally, few methods try to extend the range of objects to handle non-isometric pairs. Sumner & Popović (2004) propose an iterated closest point method with regularization based on input landmarks to deform one surface into another, and allowing the extraction of a mapping through the deformed surface. Zell & Botsch (2013) combine the concepts of deformation-based registration and transformation of surfaces into smoother shapes. While their method works relatively well for character heads, it has a strong tendency to collapse protruding extremities, such as legs and arms, which causes artifacts in the resulting mapping. Methods that deform surfaces in 3D Euclidean space are prone to fail if the surfaces have different poses; accordingly, their resulting mapping depends greatly on how well the surfaces are initially aligned. Moreover, most of these methods only handle near-isometric objects or small non-isometric deformations (Huang *et al.*, 2008), which in turn highly restricts their application domain.

1.2.2 Möbius and Functional Spaces

There is a wide range of methods for transferring correspondences which the most of them embed the given object into spaces where detecting the correspondences is easier. For instance, Möbius methods (Kim *et al.*, 2010; Lipman & Funkhouser, 2009) rely on the hypothesis that isometries are a subspace of conformal maps, which could be explored based on Möbius transformations. These methods are limited to isometric and near-isometric surfaces. Kim *et al.* (2011) present Blended Intrinsic Maps (BIM) to handle non-isometric surfaces by using weighted combinations of low-dimensional intrinsic maps to generate a blended map. The BIM method provides an efficient search procedure to find smooth maps between surfaces in a fully automatic fashion. The method handles surfaces with different poses, but it fails for examples containing small features such as facial details and fingers.

The functional space of the Laplace-Beltrami decomposition is also used to express mappings based on real valued functions instead of the regular point-to-point maps (Ovsjanikov *et al.*, 2012; Ovsjanikov, Corman, Bronstein, Rodolà, Ben-Chen, Guibas, Chazal & Bronstein, 2017). This provides a flexible representation of the maps between the shapes, but as with the Möbius methods, it struggles in handling mappings between non-isometric surface pairs. More recently Ren, Poulenard, Wonka & Ovsjanikov (2018) propose constraints on functional maps to overcome the limitations of non-isometric correspondence. They introduce an orientation-preserving regularizer and a refinement technique named Bijective and Continuous ICP (BCICP). However, their method does not provide exact bijectivity and works for low resolution meshes (10k vertice).

1.2.3 Correspondences Via a Parameterization to a Common Domain

Cross-parameterization is a powerful geometry processing tool to establish geometric correspondence. The main idea is to map the mesh into a common simple parameter domain which finding the correspondence is easier. We discuss the most popular domain choices which are a sphere and a plane.

1.2.3.1 Spherical Parameterization

The spherical domain allows a seamless and continuous parameterization of genus-zero surfaces (Alexa, 1999; Praun & Hoppe, 2003). Athanasiadis, Fudos, Nikou & Stamati (2012) drive a geometrically-constrained optimization technique to map 3D genus-zero surfaces on a sphere. Then, they apply a feature-based method to morph between surfaces with structural similarities. Mocanu & Zaharia (2012) present a spherical parameterization method relying on Gaussian curvature in order to align feature correspondences of the input surfaces. Then, they apply a morphing step by establishing correspondences. Their method generates artifacts on hands, feet, and facial feature examples due to triangle degeneration and fold-over problems. Accordingly, in spherical parameterization methods, surfaces with higher degrees of complexity and features increase the distortions.

1.2.3.2 Planar Parameterization

To deal with higher genera and objects with finer details, a widely used approach is to cut the surfaces prior to extracting the mapping. Bronstein, Bronstein & Kimmel (2006) propose a framework allowing such matching of partial shapes through an optimization that computes a minimum-distortion mapping. The approach presented in the Chap. 3 of this thesis is most similar to methods that cut the surfaces and that also flatten the resulting pieces. Some methods (Kraevoy, Sheffer & Gotsman, 2003; Sheffer, Praun & Rose, 2007) can be applied to objects of arbitrary genus, but they require a very carefully chosen and, in some cases, large set of corresponding landmarks. Aigerman, Poranne & Lipman (2014) provide a bijective mapping between the surfaces based on user landmarks and a cut-graph. It cuts the surfaces and flattens them based on a minimization of isometric distortion. Their method generates artifacts and jumps, with both of these problems occurring across the cuts. In a follow-up work, Aigerman, Poranne & Lipman (2015) overcome the cut-graph problem. The method applies a surface flattening (G-flattening), which is optimized with an energy functional. G-flattening denotes a group of affine transformations, which are optimized to minimize distortion energy. All these planar parameterization methods present the advantage of being able to handle non-isometric

surfaces and different poses, but the mappings resulting from them depend greatly on the user inputs, and distortion occurs near the landmarks. In addition to spherical and planar domains, other domains such as hyperbolic orbifolds have been proposed (Aigerman, Kovalsky & Lipman, 2017; Aigerman & Lipman, 2016). These methods are also constrained to genus-zero models. This dissertation (Chap. 3) was done as part of the paper by Ramachandran *et al.* (2018) which proposes a dual-flattening geometric correspondence method based on a cage deformation approach. It exploits a user-provided sparse set of landmarks, positioned at semantic locations, along with closed paths connecting sequences of landmarks. The approach segments the mesh and then flattens the segmented parts to adjust them with a cage deformation for aligning the interior landmarks. Afterward, it extracts the dense registration from the flattened and deformed segmented parts. The approach is capable of handling a wide range of surfaces, and is not limited to genus-zero surfaces. The results show that the proposed approach is superior to current state-of-the-art methods.

1.2.4 Edited Mapping Methods

In the last several decades, we have seen a shift in methodology in computer graphics from full automation to striking the optimum balance between automation on the one hand, and user intervention and control on the other. This shift towards the later was driven by the end users for whom control in many cases is more important than automation. In the context of geometric correspondence, there exist different methods to find mappings, but no method, even the most advanced, results in a perfect mapping every time. Some of these methods do not require any input from the user, but as such lack control over the mapping; subtle semantic features such as articulations and facial features of characters are often affected by lateral drift. Other methods allow the user to provide landmarks resulting in a certain level of control over the mapping. Nevertheless the mapping results may be good globally, but some specific semantic features can remain misaligned. Thus, the user needs to iteratively put some landmarks, and solve for the mapping globally. Consequently, deriving a local solution would be more efficient for the end-user, allowing faster visual feedback.

In comparison with establishing geometric correspondences between 3D surfaces, fewer methods have been proposed on correspondence editing. Nguyen, Ben-Chen, Welnicka, Ye & Guibas (2011) measure and optimize the consistency of sets of maps between pairs belonging to collections of surfaces. They compute a score for the map, and then apply an optimization to iteratively improve the consistency. The limitation of their method lies in its requirement of having multiple maps instead of a single map for a pair of surfaces. Ovsjanikov *et al.* (2012) propose the functional maps representation to establish correspondences between surfaces based on Laplacian eigenfunctions rather than points. Working in that smooth basis function space makes it easy and efficient to generate smooth mappings, but significant modifications to the underlying method would be required to allow local adjustments of the mapping guided by the user. Another limitation is that their method is limited to near-isometric surfaces since non-isometric deformation overrides the assumption that the change of basis matrix is sparse. Ezuz & Ben-Chen (2017) remove the isometric restriction in their proposed method for the deblurring and denoising of functional maps. They smooth a reconstructed map by mapping the eigenfunctions of the Laplacian of the target surface in the span of the source eigenfunctions. Their technique can be incorporated into existing functional mapping methods, but selecting the right number of eigenfunctions to perform the denoising is difficult. Compared with a ground truth mapping, increasing the number of eigenfunctions decreases the error until a minimum is reached, but adding more eigenfunctions beyond this point increases the error. While this can be observed on a ground truth mapping, there are no methods to achieve the minimum error for an arbitrary mapping. Gehre *et al.* (2018) incorporate curve constraints into the functional map optimization to update the mapping between non-isometric surfaces. They provide an interactive process by proposing a numerical method which optimizes the map with an immediate feedback. While their method is not limited to the isometric surfaces, it does however need several curve constraints to obtain a meaningful functional map. Vestner *et al.* (2017) improve dense mappings even in the case of non-isometric deformations. Their method is an iterative filtering scheme based on the use of geodesic Gaussian kernels. An important restriction of their method is that it requires both surfaces to be discretized with the same number of vertices and vertex densities. Panozzo *et al.* (2013) propose the weighted averages (WA) on surfaces framework. They use

WA with landmarks in order to define mappings and then the user can improve the mapping by adjusting the landmarks. Although, WA generates good mapping results, its improved mapping application cannot use an arbitrary mapping as input.

Since most state-of-the-art methods improve the mapping globally, this makes it hard for the user to fine-tune the mapping without risking modifying areas that should not be affected. Furthermore, some methods face significant limitations such as being constrained to isometric deformations and requiring compatible meshes on both surfaces. Chap. 3 proposes an approach to edit geometric correspondences by locally adjusting the mapping. It uses an automatic compact segmentation allowing a low-distortion flattening and a low-distortion deformation when aligning the landmarks set by the user in regions where the mapping exhibits some discrepancy. The presented approach aims to improve a mapping based on an iterative 2D deformation optimization to deform one mesh with respect to the other and therefore steer the landmarks toward their expected positions. As mentioned above, an alternative way to frame the correspondence editing problem is a deformation method in a planar parameterization space which we only discuss the most relevant methods here.

1.2.4.1 Non-flipping Deformation Energies

An ideal deformation energy to frame the correspondence editing problem must meet two criteria: preserving the shape and maintaining the injectivity of the mapping, so no flipped triangles happen. The planar parameterization is the best option for geometric editing since it replaces the complex 3D operations of the surface with a simpler 2D computations (Sheffer, Lévy, Mogilnitsky & Bogomyakov, 2005). Least Squares Conformal Maps, LSCM (Lévy *et al.*, 2002) apply a deformation energy which contains a term for preserving surface features and a term for position constraints. In contrast, Jacobson, Baran, Popovic & Sorkine (2011) provide a smooth and shape-preserving deformation using biharmonic weights. The main drawback of the LSCM and biharmonic weights methods is that they can introduce fold-overs while deforming the mesh. Injectivity is a key property that should be achieved in mapping editing, but extracting a mapping from meshes with fold-overs breaks this property. An approach to provide

locally injective maps is to apply inequality constraints solely to the boundaries (Lipman, 2012; Chen & Weber, 2015). First, an initial locally injective map is generated using a parameterization algorithm and then it is optimized when adhering to a specific distortion bound. The main problem is that their solution has suboptimal distortion. A recent series of approaches (Schüller, Kavan, Panozzo & Sorkine-Hornung, 2013; Smith & Schaefer, 2015; Rabinovich, Poranne, Panozzo & Sorkine-Hornung, 2017) propose a strategy where the energy of a triangle tends to infinity as the triangles become degenerate, and thus, any locally minimal solution will be, by construction, exempt of fold-overs. While this is an elegant approach, the problem is that in some cases where, due to the user constraints, some triangles will come close to becoming degenerate, they carry a disproportionately high share of the total energy as compared to the rest of the triangles. For mapping editing applications, such cases occur frequently and therefore Locally Injective Mappings, LIM (Schüller *et al.*, 2013) and Scalable Locally Injective Mappings, SLIM (Rabinovich *et al.*, 2017) often produce an inferior result both qualitatively and quantitatively. The KP-Newton method by Golla, Seidel & Chen (2018) outperforms LIM and SLIM by modifying the Newton iteration for the optimization of nonlinear energies on triangle meshes. They present that the Hessian of the isometric energies has a simple analytic expression and converge the Newton iteration faster using a simple global scaling. Moreover, they add an smoothness term to reduce distortion and improve the results.

We found that current deformation methods had drawbacks (flipped triangles and high distortion) forbidding their use in our mapping editing framework. We thus derived a new deformation approach (Chap. 3) that iteratively minimizes a conformal energy, making sure that in each iteration we have no flipped triangles. More specifically, our proposed approach optimizes the quadratic LSCM energy, but it relaxes the user constraints to avoid flips. Therefore, after each iteration, the user constraints may not be satisfied, but by repeating the process, we reach a configuration that has low conformal energy (lower than LIM, SLIM, or KP-Newton), and the user constraints are guaranteed to be better satisfied than initially.

1.3 Morphable 3D Face Model

Creating realistic 3D faces plays a key role in different areas of computer graphics, and benefits a broad range of applications. 3D morphable models are powerful statistical models widely used in many applications to generate new models. One of the most well-known previous work is proposed by Blanz & Vetter (1999). Their pioneer work proposes a model using PCA from face scans. Although they propose a multi-segment model and decompose a face into four parts to augment expressiveness, the PCA decomposition is computed globally on the whole face. Other global PCA methods have been proposed (Allen *et al.*, 2003; Cao *et al.*, 2014; Booth, Roussos, Zafeiriou, Ponniah & Dunaway, 2016; Li, Bolkart, Black, Li & Romero, 2017; Booth, Roussos, Ponniah, Dunaway & Zafeiriou, 2018; Lüthi, Gerig, Jud & Vetter, 2018). A downside of global PCA-based methods is that they exhibit global support: when we adjust the eye, the nose may also undergo undesirable changes. Another downside is a lack of intuitive user control for face editing. While the eigenanalysis is good at extracting the dominant modes of variation of the data, they provide weak intuitive interpretation. To address the former problem, several local models have been proposed that segment the face into independent sub-regions and select the most dominant eigenvectors per part. Tena *et al.* (2011) propose a method to create localized clustered PCA models for animation. They select the location of the basis using a spectral clustering on the geodesic distance and correlation of vertex displacement considering variation in the expressions. SPLOCS (Neumann *et al.*, 2013) propose the theory of sparse matrix decompositions to produce localised deformation from an animated mesh sequence. They use vertex displacement in the Euclidean coordinates to select the basis in a greedy fashion. We noticed that when considering variation in identity instead of variation in expression, the greedy selection leads to basis which are far less local than those shown in the paper of Tena *et al.* (2011). Chi, Gao & Zhang (2017) adaptively segment the face model into soft regions based on user-interaction and coherency coefficients. Afterwards, they estimate the blending weights which satisfy the user constraints as well as the spatio-temporal properties of the face set. Cao, Chai, Woodford & Luo (2018) segment the face with the same spectral clustering followed by manual adjustment as Tena *et al.* (2011). While their method focuses mostly on expression,

they also provide some identity modeling, as they rely on the FaceWarehouse (Cao *et al.*, 2014) global model, which they decompose using the segments defined by spectral clustering. In their case, the goal is to do the real-time adaptation of a 3DMM to a face from a video feed. While their method works remarkably well for the real-time application of "virtual makeup", it does not provide a very detailed facial model compared to ours, and it does not support a face editing workflow. Overall, these methods address facial animation instead of face modeling and therefore assume large yet localized deformations caused by facial expressions which are different from our context where each face is globally significantly different from the others.

Other methods supplement decomposition approaches with the extraction of fine details, allowing to reconstruct a faithful facial model (Cao, Bradley, Zhou & Beeler, 2015; Garrido, Zollhöfer, Casas, Valgaerts, Varanasi, Pérez & Theobalt, 2016; Shi, Wu, Tong & Chai, 2014). The major problem with these approaches is that they work for a specific person and do not provide editing capabilities. The Phace method (Ichim, Kadleček, Kavan & Pauly, 2017) allows the user to edit fat or muscle maps in texture space on the face. While this provides a physically-based adjustment, the control is implicit. The user modifies the texture and then the system simulates muscles and fat to get the result. Wu, Bradley, Gross & Beeler (2016) propose an anatomically-constrained local deformation model to improve the fidelity of monocular facial animation. Their model uses 1000 overlapping parts and then decouples the rigid pose of the part from its non-rigid deformation. While this approach works particularly well for reconstruction, the parts are too small for editing semantic face parts such as the nose or the eyes.

As opposed to the methods described so far, the methods of Allen *et al.* (2003) and BodyTalk (Streuber, Quiros-Ramirez, Hill, Hahn, Zuffi, O'Toole & Black, 2016) greatly ease the editing by mapping intuitive features to modifications of global 3DMM eigenvector weights. In particular, BodyTalk (Streuber *et al.*, 2016) relates transformations of the meshes to keywords such as "fit" and "sturdy". While the mapping between the words and the deformations is not perfect, it still makes it reasonably intuitive to edit the mesh of the body. One problem with this method, is that it provides words for bodies, not faces. A second major problem is the inability to do local adjustments; adjustments that increase the length of the legs will result in changes to other regions such as the torso and arms.

A downside of global PCA based methods is that they exhibit global support: adjusting parameters to change one part has unwanted effects on other unrelated parts. To address this problem, we propose an approach (Chap. 4) to segment the face into independent sub-regions and provide a process to select the best set of eigenvectors given a target number of eigenvectors. This dissertation demonstrates that our approach is better suited to the task of face editing than these methods. Another problem of most of the previous work is that it does not allow facial model editing through the adjustment of objective measurements. In contrast, the proposed approach relies on anthropometric measurements used as controls for the editing.

1.4 Summary

This chapter presented a review of the current state-of-the-art in geometric correspondence and modeling. To recap, all of the methods used for retargeting skeleton and skin weights between characters have different limitations. Recent binding methods ease the creation of binding weights but they still result in artifacts that require manual weight painting. Other methods dependant on a template character or a large database of rigged characters limit applicability in a typical animation pipeline. However, the proposed approach by Avril *et al.* (2016) – Chap. 2 in this thesis – presents a step toward providing an easy-to-use, automatic, and high quality character attribute retargeting pipeline. Since it relies on establishing a meaningful geometric correspondence between characters and the more accurate this geometric correspondence is, the more accurate the skeleton and skinning weight transferring will be, a section was also dedicated to geometric correspondences. Current state-of-the-art methods for geometric correspondences often have problems in addressing combined goals. They are able to handle a wide range of surfaces, but impose genus based limitations such spherical parameterization methods. Often handling small features and non-isometric surface pairs in different poses remains to be a challenge. Our paper (Ramachandran *et al.*, 2018) – discussed in Chap. 3 – proposes a way in handling these practical problems. While computing a map between two surfaces is a fundamental problem in digital geometric processing, methods and tools for editing such a mapping have a critical practical value in several contexts. The most popular approaches were

limited to a specific initial mapping method which highly restricts their application domain. Other methods improved the mapping globally instead of locally. Locality has an additional advantage that the user is guaranteed that the update of the mapping will not have negative effects elsewhere in the mapping. Out of the few methods dedicated to the improvement of mappings, all of them operate globally. Accordingly, we proposed an approach in Chap. 3 (Ghafourzadeh *et al.*, 2020b) to enable the user to locally adjust the mapping and solving for much fewer constraints. The proposed approach relies on a 2D deformation optimization to align the landmarks while limiting distortion and improving the mapping. Since current state-of-the-art deformation methods were not satisfactory and produced distortion, we proposed an iterative deformation approach to minimize a conformal energy without producing flipped faces.

A review of some of the currently available 3DMM algorithms was also presented. Most of the aforementioned approaches do not provide a good user control to edit a face locally and exhibit global support. Consequently, adjusting parameters to change one part has unwanted effects on other unrelated parts. Some of the methods focus on bodies or address facial animation instead of facial modeling. To address these problems, Chap. 4 proposes an approach (Ghafourzadeh *et al.*, 2020a) to separate the face into regions and compute independent PCA decompositions on each region allowing accurate reconstruction. It also provides local and global editing relying on a subset of anthropometric measurements selected based on error reconstruction.

CHAPTER 2

ANIMATION SETUP TRANSFER FOR 3D CHARACTERS

We collaborated on the "Animation setup transfer for 3D character" paper (Avril *et al.*, 2016) which is about transferring the animation setup between characters. This chapter presents a short explanation of that paper mostly limited to the key contributions of this dissertation. For a full description of the approach, we refer the reader to the paper.

Skeletal animation is the common way to animate characters and generally used in movie industries and video games. A complete animation setup needs a skeleton and skin weights. The skeleton is a hierarchy of bones and joints connecting bones to each other. The skeleton is embedded into a detailed character mesh and then skin weight sets the impact of each joint on each vertex of the mesh. Consequently, each vertex has one weight value for each joint which varies from zero (no impact) to one (maximum impact). Skeletons and skinning weights – herein referred to as *animation setup* – form the basis of most animated characters.

As making a character ready for animation by an artist is a time consuming and complex process, it can be useful to reuse the animation setup from existing characters. This has the additional benefit of preserving authoring details in the source skeleton and skinning weights, limiting unwanted deformation artifacts, and easing animation reuse. Avril *et al.* (2016) propose to transfer animation setup between characters in three key steps. First, a geometric correspondence between the source and target characters should be established. In the next step, the skeleton is retargeted from the source to the target character. Finally, the source character's skin weights are transferred to the target character. This dissertation is mostly focused on the geometric correspondences and skin weights transferring steps of this paper. To date, there exists few methods to transfer the entire animation setup between characters. Some methods have focused to automate skinning weight selection (Baran & Popović, 2007; Kavan & Sorkine, 2012; Dionne & de Lasa, 2014). They provide good initial skinning weights, but they do not guarantee perfect results in all cases and artists usually need to manually adjust the weights. Alternative approaches (Baran & Popović, 2007; Dicko *et al.*, 2013; Miller *et al.*, 2011) use a template

character or a database of previously authored results to ease the creation of new characters. Animation setup transfer (Avril *et al.*, 2016) is complementary to these methods, as it allows the artist to provide his own skeleton structure and skinning weights, instead of being constrained to those found in a template character or in a database of characters. It relies on establishing a geometric correspondence (Zell & Botsch, 2013) between a source and a target characters. Since this step is a key step in animation setup transferring, we improve this step by reducing triangle flipping and self-intersections. Energy minimization techniques are then used to transfer animation setup, while accounting for the interrelations between mesh vertex positions, skinning weights, joint positions, and joint orientations. This dissertation makes a specific contribution by filtering the skinning weights from the source based on the geometric correspondence and differences in mesh resolution. This enhancement approach leads to more accurate and precise transferring results. Afterwards, we evaluate the stability and convergence of the approach by performing several animation setup transfer tests and present artifacts that arise when not using the proposed skin weight resampling.

2.1 Geometric Correspondence

Geometric correspondence is a key step in animation setup transferring. The more accurate the correspondence is, the more accurate the transferring will be. Avril *et al.* (2016) relies on determining the point-to-point correspondence between $M_{\mathcal{S}}$ and $M_{\mathcal{T}}$, linking together points with similar semantic meanings (e.g., tip of the right finger, top of the head). Techniques from the broad class of geometric correspondence methods can be used to extract the correspondence, but the required geometric correspondence properties are smoothness and bijection.

In this section, we provide a brief summary of the *Elastiface* method; for full details, refer to Zell & Botsch (2013). *Elastiface* is a semi-automatic method, which computes a dense correspondence from a set of user-specified markers. It can be used for non-isometric source and target meshes, a key requirement for attribute transfer between production geometries. *Elastiface* uses a three-step process: (1) setting markers, (2) deforming $M_{\mathcal{S}}$ so that it matches $M_{\mathcal{T}}$, and (3)

extracting the dense correspondence based on closest locations between the deformed $M_{\mathcal{S}}$ and $M_{\mathcal{T}}$.

First, markers are manually placed on $M_{\mathcal{S}}$ and $M_{\mathcal{T}}$ (Fig. 2.1). As each marker is represented in barycentric coordinates, it can be precisely positioned anywhere on the surface of the mesh. For our experiments, which requires detail preservation in areas such as the hands and feet, we found that a set of 325 markers was needed to obtain high quality results (see the accompanying video).

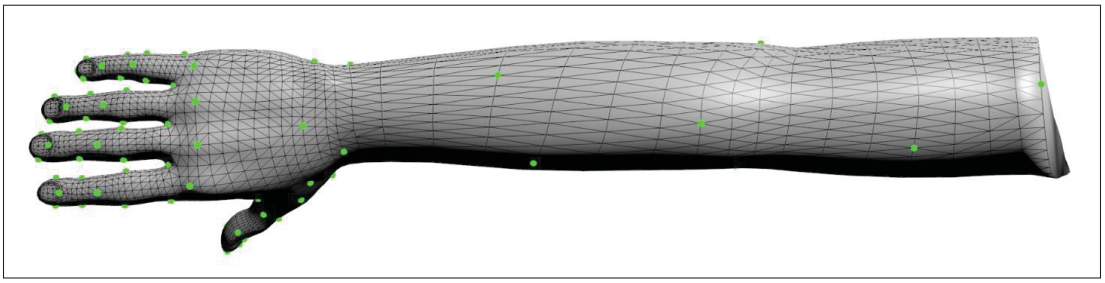


Figure 2.1 Markers on the arm of the Man

The second step of the geometric correspondence is to deform $M_{\mathcal{S}}$ so that it matches $M_{\mathcal{T}}$. This is done through a series of four separate substeps to align, scale, and deform $M_{\mathcal{S}}$ so that its position, orientation, scale, and shape become similar to those of $M_{\mathcal{T}}$. These computations involve energy minimizations based on marker positions, vertex Laplacian, and Voronoi areas. Some of these substeps also involve per-vertex closest location between the deformed and the initial target meshes, which we improved by rejecting locations with inconsistent normal vectors. When computing the closest location for a specific vertex, normal vectors of each candidate polygon are inspected and the polygon is rejected if the angle with the vertex normal is greater than 90°. As the joint fairing step can severely distort the vertex and face normals, we rely on the input mesh normals. This rejection test greatly improves the robustness by reducing triangle flipping and self-intersections.

Once M_S has been deformed to match M_T, the last step is to extract the dense correspondence based on closest locations between the deformed source mesh and M_T (again rejecting locations with inconsistent normal vectors). With this correspondence, each vertex of the deformed source is linked to a barycentric coordinate on a polygon of M_T, and vice versa. We adjusted *Elastiface* parameters to improve test results. Overall, we found this to be a straightforward process. In the end, all test models shared several parameter values: for substep two $\lambda_2 = 100$ and $\lambda_3 = 10$, for substep three $\mu_1 = 0.01$, $\mu_2 = 1\,000$, and $\mu_3 = 1$, and for substep four $\mu'_2 = 1\,000$, and $\mu'_3 = 0.1$. Compared to *Elastiface*, smoothing parameters (λ_1, μ_1, and μ'_1) and marker parameters (μ_3 and μ'_3) were smaller, while mesh projection parameters (μ_2 and μ'_2) were larger. Smoothing parameters (λ_1 and μ'_1) were adjusted in powers of ten; increasing the value improved mesh smoothing, while decreasing it helped preserve the mesh volume and limit collapsing ($\lambda_1 \in [1e-7, 0.1]$ and $\mu'_1 \in [1e-6, 0.1]$). All of these adjustments helped to constrain the source mesh to fit the different target meshes. Examples of the resulting geometric correspondences are highlighted in Fig. 2.2.

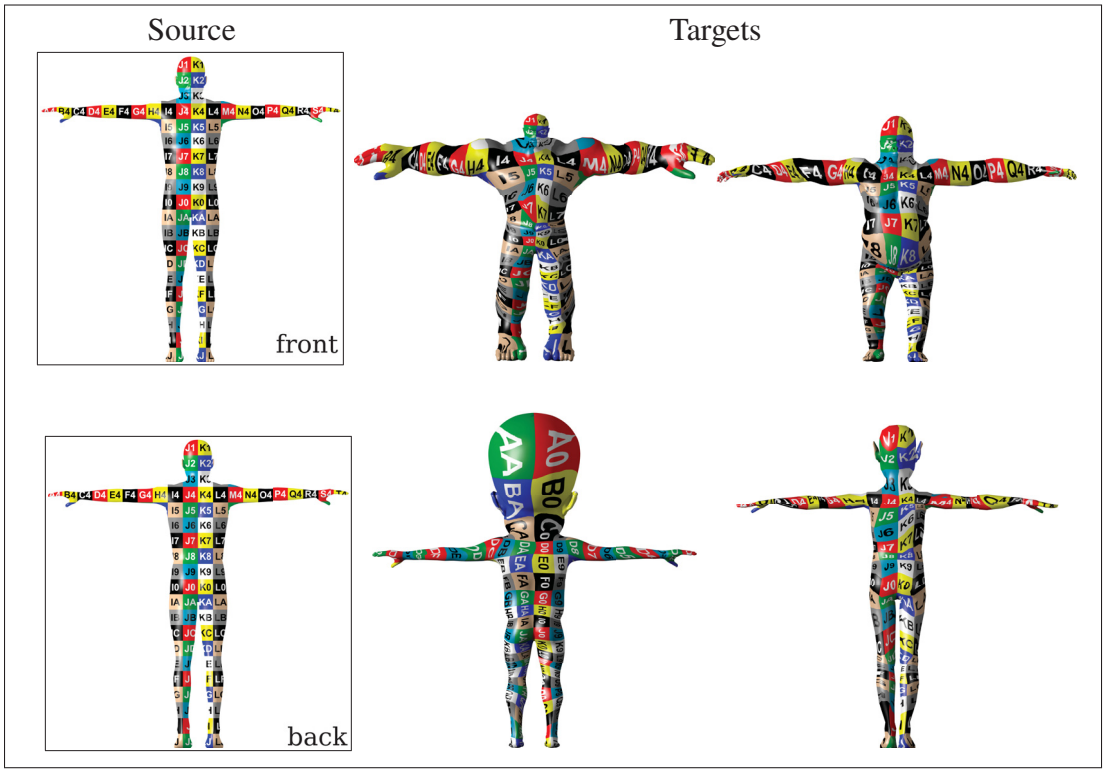


Figure 2.2 Geometric correspondences among various characters

2.2 Skinning Weight Transfer

Skinning weights are necessary to animate a character based on its skeleton. This section explains how we transfer the skinning weights from the source to the target character. We derived an approach to filter the skinning weights from the source based on the geometric correspondence and differences in mesh resolution.

Skinning weights $W_{\mathcal{T}}$ are obtained using the geometric correspondence from $M_{\mathcal{T}}$ to $M_{\mathcal{S}}$. For each target vertex $v_{i\mathcal{T}}$, its corresponding location on $M_{\mathcal{S}}$ is defined as a barycentric coordinate and a triangle index. For meshes with the same resolution or when $M_{\mathcal{S}}$ is coarser than $M_{\mathcal{T}}$, linear interpolation from the source skinning weights provides a good resampling. In this case, the new skinning weight for a vertex of $M_{\mathcal{T}}$ is computed using the corresponding barycentric coordinates and triangle on $M_{\mathcal{S}}$:

$$w_i^{\text{blend}} = \beta_a w_{a\mathcal{S}} + \beta_b w_{b\mathcal{S}} + \beta_c w_{c\mathcal{S}}, \tag{2.1}$$

where a, b, and c correspond to the three vertices of the source triangle, $w_{a\mathcal{S}}$, $w_{b\mathcal{S}}$, and $w_{c\mathcal{S}}$ are their weights, and β_a, β_b, and β_c are the barycentric coordinates.

When $M_{\mathcal{T}}$ is coarser than $M_{\mathcal{S}}$, linear blending with barycentric coordinates results in under-sampling. Even considering all the skinning weights within the curvilinear triangle a, b, and c, several vertices and weight values from the source may remain unused, which will lead to aliasing and artifacts in the skin deformation during animation. In Fig. 2.3, the middle column shows how barycentric interpolation causes artifacts not found on the source characters. In the right-most column, the deformation is more faithful to the source with our improved filtering.

To detect when such aliasing is likely to occur, the size of corresponding polygons should be compared between $M_{\mathcal{T}}$ and $M_{\mathcal{S}}$. This comparison can readily be computed, as $M_{\mathcal{S}}$ is re-scaled to match the dimensions of $M_{\mathcal{T}}$ during the geometric correspondence. Thus, to detect vertices requiring additional filtering, the Voronoi area of each target vertex $VA(v_{i\mathcal{T}})$ is compared to the corresponding blended Voronoi area of the source (Alg. 2.1, line 2 and 4).

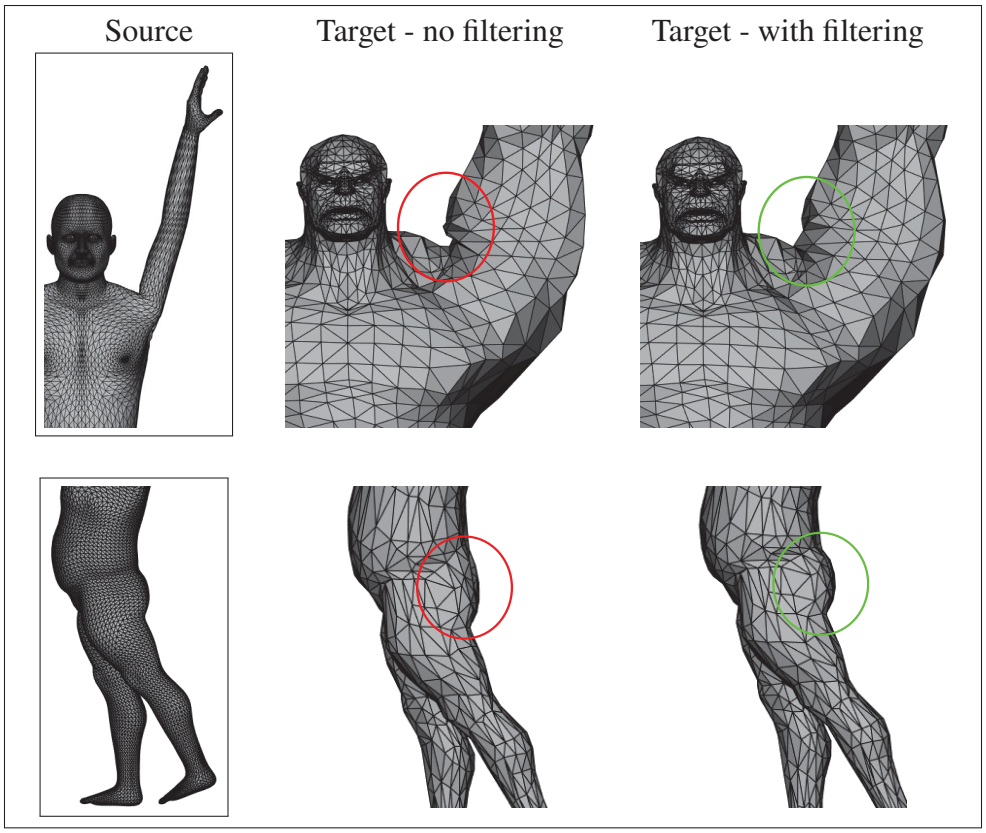


Figure 2.3 Deformation on low-resolution target meshes without and with filtering

When $VA(v_{i\mathcal{T}})$ is greater than VA^{blend}, the vertices from the next ring of neighbors around the location of the geometric correspondence are identified (Alg 2.1, line 10 and 20), these vertices are added to the set of selected vertices (Alg 2.1, line 16), and their Voronoi areas are added (Alg 2.1, line 18), until adding the next ring would result in a total Voronoi area larger than $VA(v_{i\mathcal{T}})$ (Alg 2.1, line 14). We then have:

$$VA^{\text{blend}} + \sum_{v_{j\mathcal{S}} \in \text{Sel}} VA(v_{j\mathcal{S}}) < VA(v_{i\mathcal{T}}),$$

$$VA^{\text{blend}} + \sum_{v_{j\mathcal{S}} \in \text{Sel}} VA(v_{j\mathcal{S}}) + \sum_{v_{k\mathcal{S}} \in \text{Next}} VA(v_{k\mathcal{S}}) \geq VA(v_{i\mathcal{T}}).$$

To get a Voronoi area of exactly $VA(v_{i\mathcal{T}})$, we also consider the contribution of vertices from the next ring of neighbors. Their influence will be less than that of the selected vertices. The

Algorithm 2.1 Selecting vertices for skinning weights filtering

$VA^{\text{blend}} = \beta_a VA(v_{a\mathcal{S}}) + \beta_b VA(v_{b\mathcal{S}}) + \beta_c VA(v_{c\mathcal{S}})$
if $VA^{\text{blend}} < VA(v_{i\mathcal{T}})$ **then**
    Sel = {}
    Sum = VA^{blend}
    Next = $\{v_{a\mathcal{S}}, v_{b\mathcal{S}}, v_{c\mathcal{S}}\}$
    $VA_{\text{Next}} = \sum_{v_{k\mathcal{S}} \in \text{Next}} VA(v_{k\mathcal{S}})$
    while Sum + $VA_{\text{Next}} < VA(v_{i\mathcal{T}})$ **do**
        Sel = Sel ∪ Next
        Sum = Sum + VA_{Next}
        Next = vertices from next ring of neighbors
        $VA_{\text{Next}} = \sum_{v_{k\mathcal{S}} \in \text{Next}} VA(v_{k\mathcal{S}})$
    end while
end if

weights are thus combined with respect to their Voronoi areas:

$$w_{i\mathcal{T}} VA(v_{i\mathcal{T}}) = w_i^{\text{blend}} VA^{\text{blend}} \tag{2.2a}$$

$$+ \sum_{v_{j\mathcal{S}} \in \text{Sel}} w_{\mathcal{S}}(v_{j\mathcal{S}}) VA(v_{j\mathcal{S}}) \tag{2.2b}$$

$$+ \sum_{v_{k\mathcal{S}} \in \text{Next}} w_{\mathcal{S}}(v_{k\mathcal{S}}) VA(v_{k\mathcal{S}}) \left(\frac{VA(v_{i\mathcal{T}}) - VA^{\text{blend}} - VA_{\text{Sel}}}{VA_{\text{Next}}} \right), \tag{2.2c}$$

where $VA_{\text{Sel}} = \sum_{v_{j\mathcal{S}} \in \text{Sel}} VA(v_{j\mathcal{S}})$ and $VA_{\text{Next}} = \sum_{v_{k\mathcal{S}} \in \text{Next}} VA(v_{k\mathcal{S}})$. The combination from the previous equation is then divided by $VA(v_{i\mathcal{T}})$ to extract the skinning weight. Examples of skinning weights obtained with and without the proposed weight filtering approach are shown in Fig. 2.4. Artifacts caused by the barycentric interpolation from a high-resolution mesh to a low-resolution mesh are highlighted in red. The areas highlighted in green show the improvement resulting from our filtering approach. We can see that the weights more faithfully match the source, and in this case weights are also more symmetric, when filtering is applied.

Considering a skeleton with n joints, every vertex stores n weight values, with each weight describing the relationship between the transformation of the corresponding joint J_l and the transformation of $v_{i\mathcal{T}}$. To limit unwanted animation artifacts, the last step involves "normalizing"

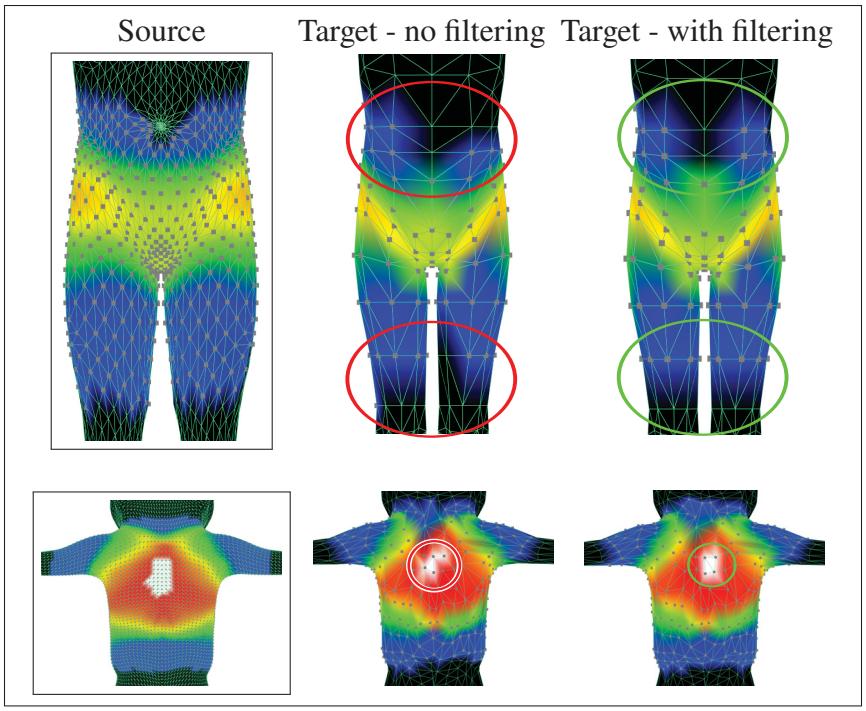


Figure 2.4 Impact of the skinning weights filtering

skinning weights:

$$\sum_{l=1}^{n} w_{J_l} = 1, \tag{2.3}$$

such that their sum is one. Once the weights are transferred, any skinning method can be applied, such as linear blend skinning or dual quaternion skinning.

2.3 Results

Fig. 2.5 highlights the fact that our approach is not limited to the use of specific characters. The skeleton and skinning weights are transferred from the source (the Man) to 11 target meshes with different topologies and with diverse numbers of vertices and faces. The animation of the target characters still match the source character for unusual characters with various morphological characteristics.

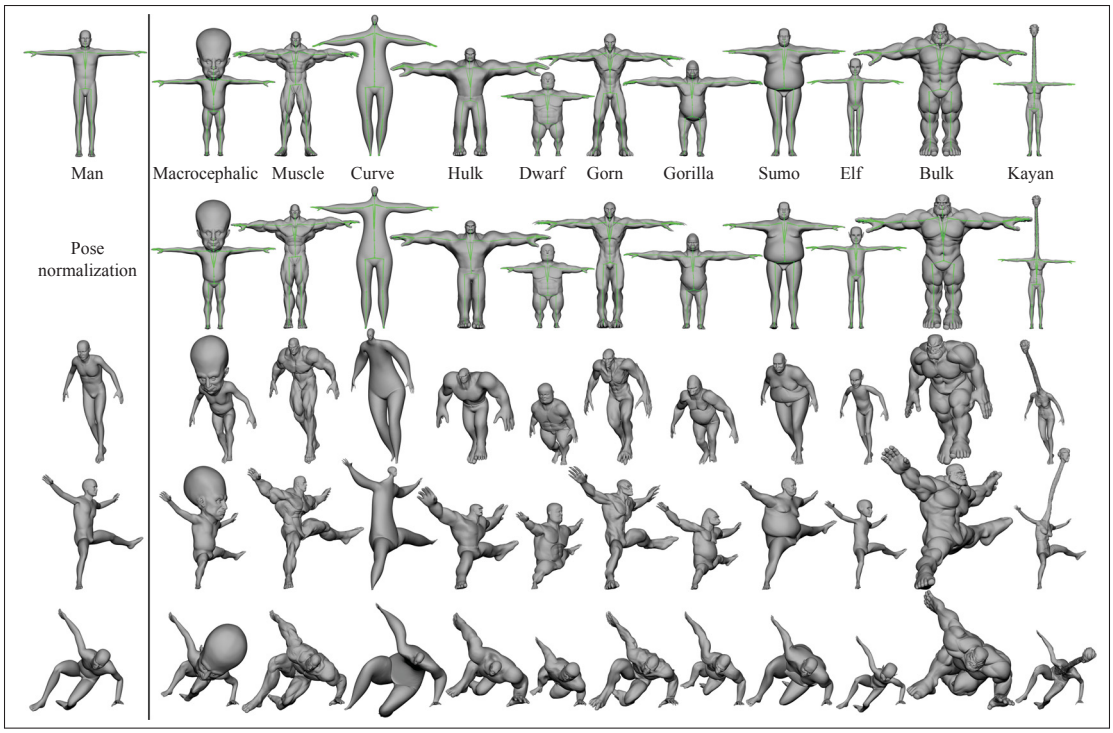


Figure 2.5 The 12 characters used in our tests contain significant differences in the shape, length, and diameter of limbs

Our approach also handles meshes with different resolutions. Examples include retargeting from a coarse proxy character (2,059 vertices) to finer resolution meshes (5,888 and 20,087 vertices) in the left side of Fig. 2.6, and retargeting from a high-resolution VFX character (23,571 vertices) to mobile game low-resolution meshes (7,072 and 2,358 vertices) of that character in the right side of Fig. 2.6. Note that, despite the coarse resolution of some test meshes (2k polygons), hands and fingers are properly retargeted.

2.4 Discussion

In this section, we discuss different aspects of our approach. We argue the geometric correspondence details. We then discuss the stability and convergence of the approach, and end by presenting different comparisons highlighting the impact of the skin weights filtering.

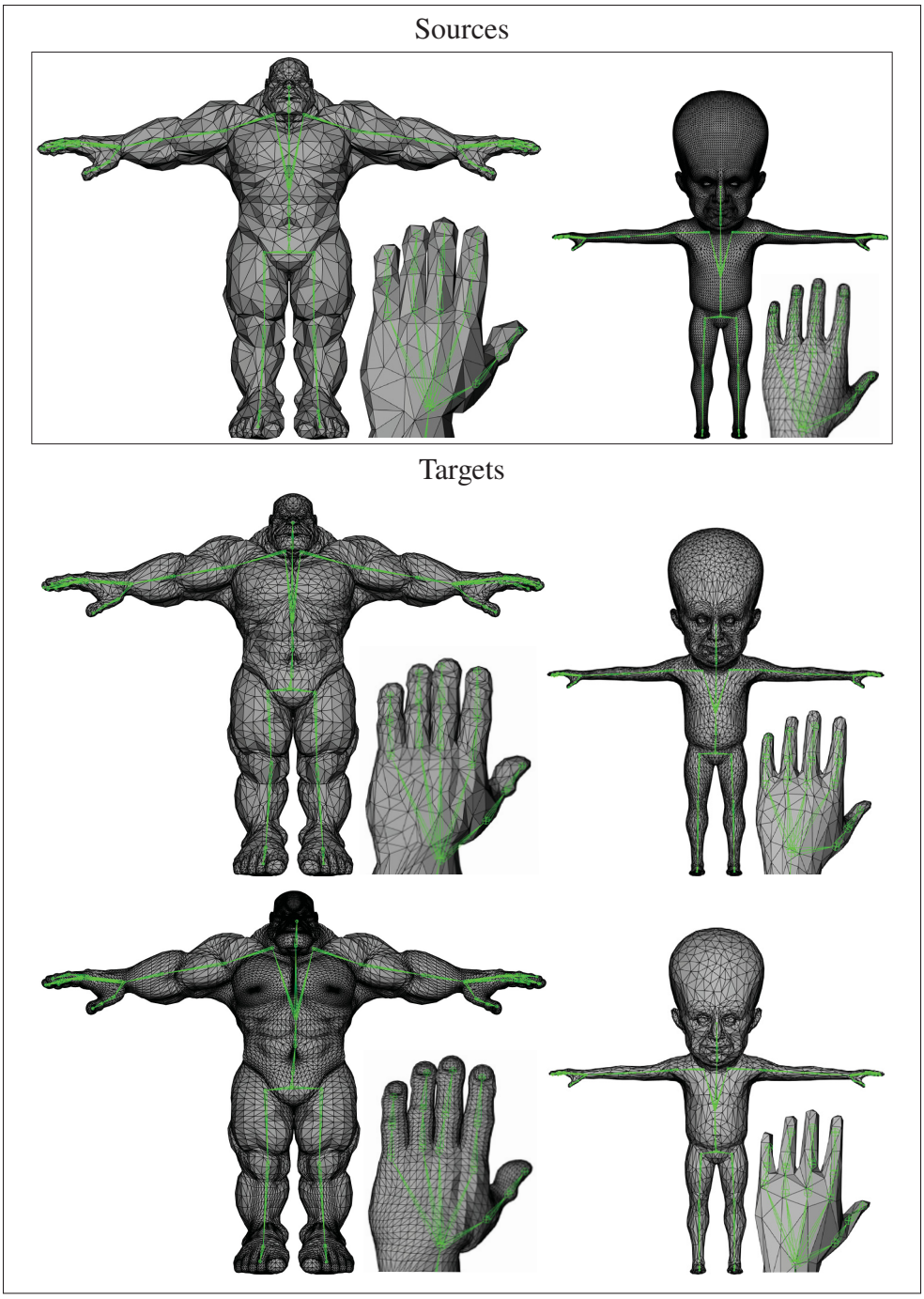


Figure 2.6 Example with different mesh resolutions and topologies. Transferring from a low-resolution mesh to finer meshes on the left and vice versa on the right

2.4.1 Geometric Correspondence

The discrete Laplacian and Voronoi area are used to compute geometric correspondence. These operators require that input geometry M_S to be manifold and watertight. As automatic conversion of arbitrary geometry to meet these requirements is an open research problem, we opted instead to manually clean-up meshes. Setting and linking markers on the characters is time-consuming, requiring 40-60 minutes for each source/target mesh pair. Since many character creation workflows rely on a single source character, which is transferred to various target characters, one of the meshes will already have markers, limiting edits to the target and reducing time to complete manual marker placement.

An alternative strategy for establishing correspondence, used widely in production studios, is to use *uv* texture maps. As our characters can have various mesh topologies and a wide range of mesh resolutions, we found it easier to set markers compared to unwrapping and deforming multiple meshes to a single consistent *uv* map. Using a consistent *uv* map may also constrain character design choices.

In developing our pipeline, we considered various geometric correspondence methods, including *Mobius Voting* (Lipman & Funkhouser, 2009) and *Blended Intrinsic Maps* (Kim *et al.*, 2011). For our non-isometric meshes, we found the method of Sumner and Popović (Sumner & Popović, 2004) and *Elastiface* (Zell & Botsch, 2013) to work best. In our tests, *Elastiface* provided better results. Fig. 2.7 shows that the method of Sumner and Popović can produce unwanted distortions (see first and second rows). In the last row, Elastiface produces significantly better remapping for the character's arms.

As *Elastiface* can transfer textures from a source to a target character, we inherit this feature, as well as the possibility to retarget other vertex-based attributes. Nevertheless, this cannot be done directly as unwrapping the source mesh in texture space introduces discontinuities by disconnecting neighbor polygons. A polygon-splitting method should be developed to introduce these discontinuities on the target mesh, so that each and every portion of the target polygons can be assigned correct texture coordinates.

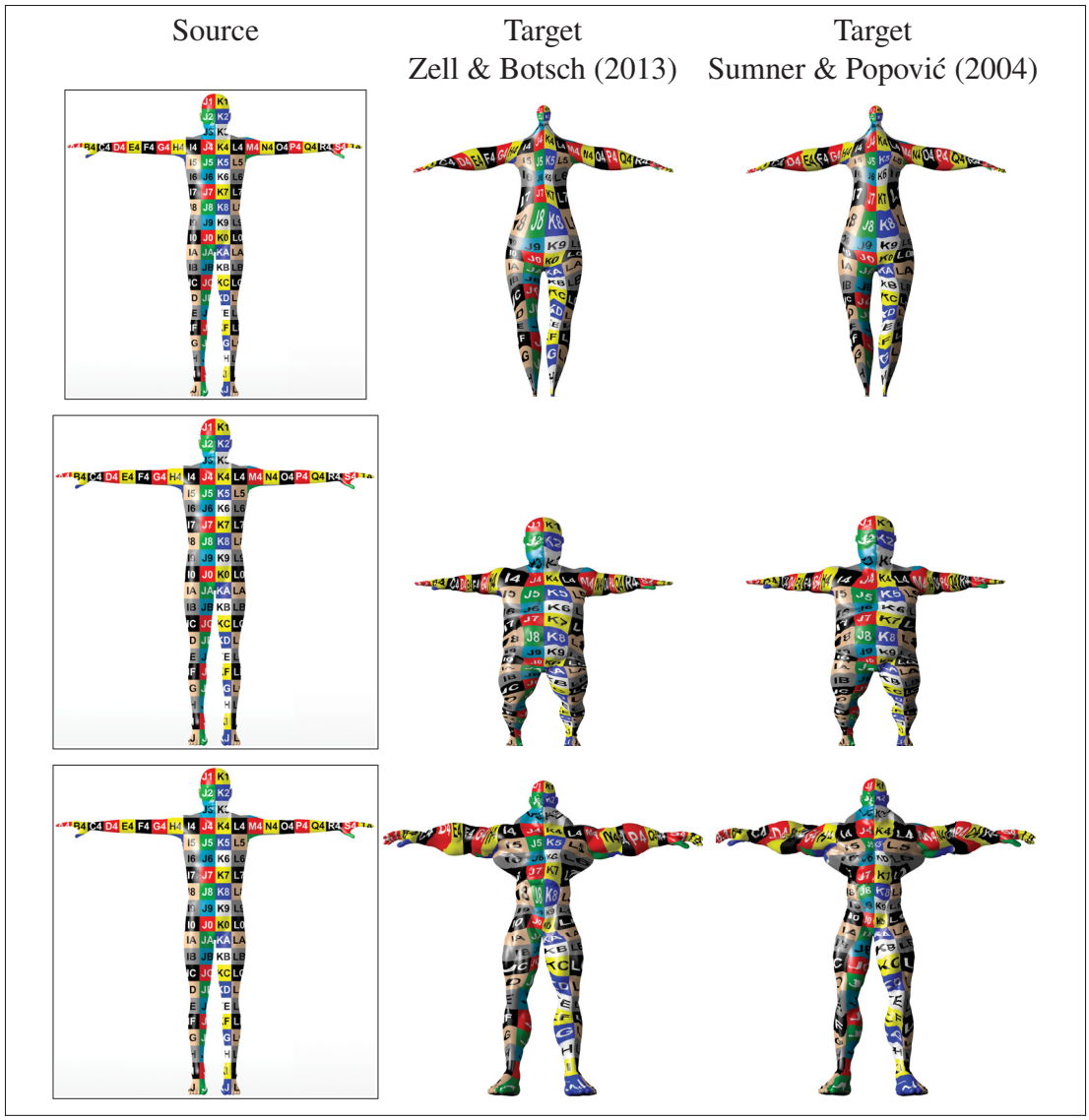


Figure 2.7 Comparison of different geometric correspondence methods

One limitation of Avril *et al.* (2016) is the geometric correspondence method itself, as it requires the meshes to be in a similar pose. One of its advantages though, is that it does not rely specifically on *Elastiface*. Consequently, any other geometric correspondence method could be used without affecting the other stages of the approach. This led to an interesting direction for our follow-up work (Ramachandran *et al.*, 2018).

2.4.2 Stability

To evaluate the stability, we performed several animation setup transfer tests. We first retargeted the skeleton and skinning weights from the Man back onto itself three times. To distinguish between errors introduced by the geometric correspondence versus retargeting steps, we first used the vertex-to-vertex correspondence for mapping the mesh back onto itself. After three iterations, the average joint position error was 0.0011% of the character's axis aligned bounding box, while skinning weights were identical. For the second test, the Man was retargeted onto itself, but this time using the geometric correspondence from *Elastiface*. The average error after the third iteration was 0.3% for the joint positions and 0.0061 for the weights. In the third test, animation setup was transferred back and forth, several times, between the Man and another character (Man $\rightarrow$ other $\rightarrow$ Man$'$ $\rightarrow$ other$'$). We could thus compare between Man and Man$'$, as well as between other and other$'$. The average error was computed for Man – Man$'$, Curve – Curve$'$, and Gorn – Gorn$'$. Out of all these combinations, the maximum error was 0.14% for the joint positions and 0.0015 for the weights. Overall, our pipeline proved stable for the case of repetitive retargeting, across a variety of input.

We compared the quality of the mesh deformation when animating characters using the skinning weights transferred with our approach, to the deformation resulting from the use of two common automatic skin binding methods (Baran & Popović, 2007; Dionne & de Lasa, 2014). While the mesh deformation from automatic binding methods are satisfactory, we could still identify several issues that do not show up when our method retargets manually crafted skinning weights (Fig. 2.8).

Recently, Moutafidou & Fudos (2019) introduced an integrated platform for animation transfer with minimal user intervention compared with our approach (Avril *et al.*, 2016). Their method is also almost 40% faster on average than our approach. They apply the L-BFGS algorithm with a gradient descent method to optimize the performance of the geometric correspondence step. Another method, Molla, Debarba & Boulic (2017), which also appeared after our paper was published proposes an egocentric normalization of the body-part relative distances. Their

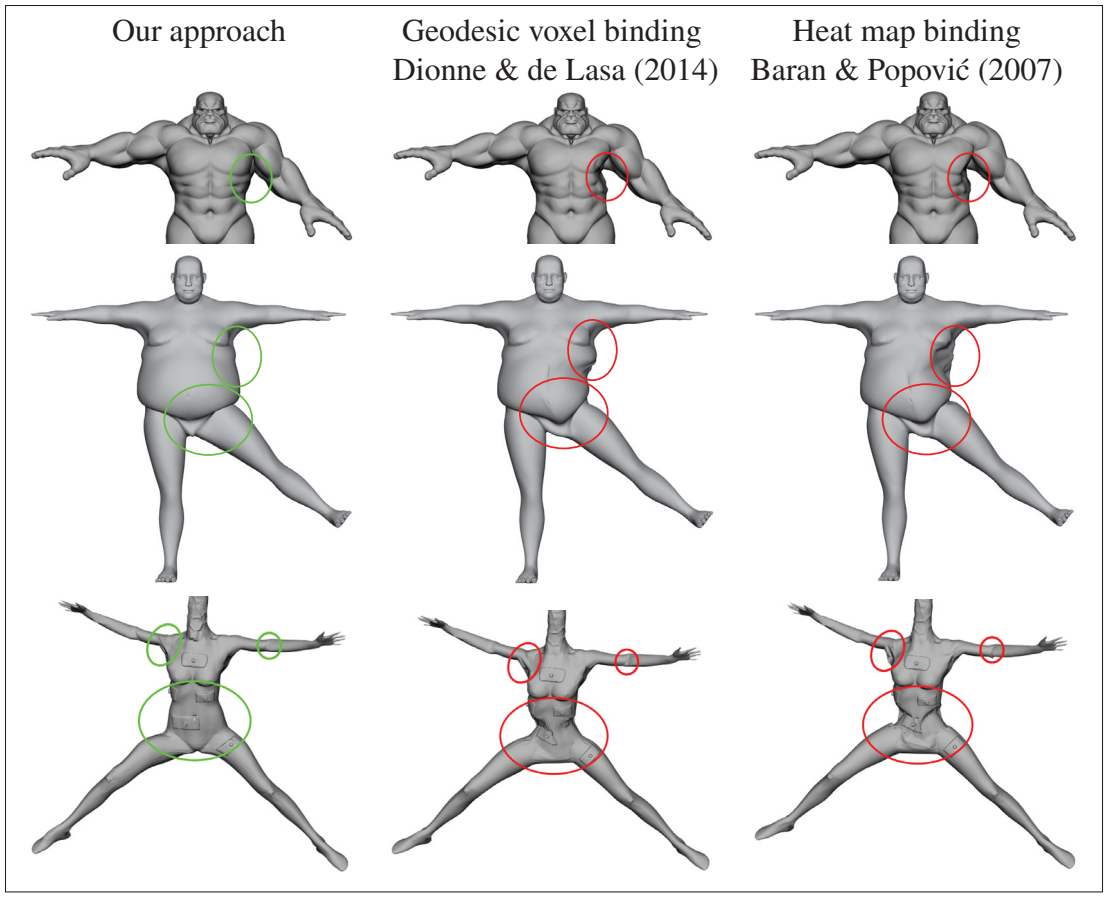


Figure 2.8 Mesh deformation using our transferred weights compared to weights generated by automatic binding methods

method preserves the consistency of self contacts for a large variety of characters and encodes the whole posture space in real-time while our approach relies on a registration-based method which is time-consuming in general and far from real-time.

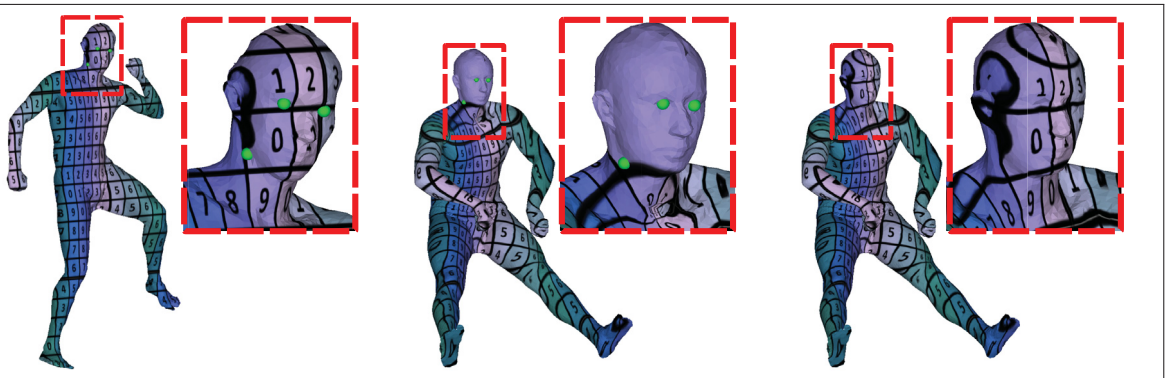
CHAPTER 3

LOCAL EDITING OF CROSS-SURFACE MAPPINGS

The work presented in this chapter was carried out in collaboration with the researchers of two papers, "Joint planar parameterization of segmented parts and cage deformation for dense correspondence" (Ramachandran *et al.*, 2018) and "Local Editing of Cross-Surface Mappings with Iterative Least Squares Conformal Maps" (Ghafourzadeh *et al.*, 2020b). The joint planar paper by Ramachandran *et al.* (2018) proposes a robust and efficient dual-flattening geometric correspondence method relying on a novel cage deformation method. Based on our experiments with several geometric correspondence methods including the one of Ramachandran *et al.* (2018), no single method results in a perfect mapping in every case. Accordingly on our follow-up work, we propose a novel approach to improve a given surface mapping through local refinement (Ghafourzadeh *et al.*, 2020b). This chapter presents a comprehensive explanation of this paper as it is one of the key contributions of this dissertation. Throughout this chapter, we also present a brief explanation of specific elements of the paper by Ramachandran *et al.* (2018), limited to the key contributions of this dissertation as well. For a full description of the approach by Ramachandran *et al.* (2018), we refer the reader to the paper.

Computing a cross-surface mapping between two surfaces (cross-parameterization) is a fundamental problem in digital geometric processing. It maps the vertices of one surface to the corresponding locations on the other by parameterizing the input surfaces on a common domain and is guaranteed to be bijective. A wide range of methods have been developed to find such mappings (Botsch & Sorkine, 2008; Nogneng & Ovsjanikov, 2017; Ramachandran *et al.*, 2018), but no single method results in a perfect mapping every time. Quite often, the mapping results may be good overall, but some specific, sometimes subtle, semantic features, such as articulations and facial features, may remain misaligned, as illustrated in Fig. 3.1. These issues over the final result are often unacceptable in a production setting where the artist needs more freedom to control the result. Typically, improving results using surface mapping methods requires the user to iteratively insert some landmarks and solve for the mapping globally. However, since the

imperfections are typically localized into a specific region, a local solution that does not change the mapping globally would be preferred in order to ensure that the method does not introduce artifacts elsewhere on the map.



a) Mesh A b) Mesh B, initial mapping Aigerman & Lipman c) Mesh B, edited mapping

Figure 3.1 The initial mapping between mesh A and B is globally good, but is locally misaligned in the head region. Using our approach, the mapping is locally improved

This dissertation proposes a surface mapping editing approach providing local and precise control over the map adjustments. The process begins with the inspection of an existing vertex-to-point surface mapping between two meshes. In regions where the mapping exhibits some discrepancy, the user sets landmarks positioned at corresponding locations on both meshes. For each such region, we extract a patch on both meshes in order to localize the changes in the mapping, and we flatten them on a common planar domain. The mapping is improved based on a 2D deformation optimization that steers the landmarks toward correspondence while limiting distortion and having theoretical guarantees to maintain the local injectivity of the map. We developed a new 2D deformation approach denoted ILSCM, which iteratively minimizes a conformal energy, each iteration ensuring that flips do not occur, and in practice, ensuring progress toward satisfying the constraints. We chose to work with a conformal energy as we want to be able to improve mappings where the deformation between the pair of meshes is not isometric. Our editing approach can successfully align the mapping around landmarks without any degradation of the overall mapping. The local surface maps are extracted from their respective deformed segments

and parameterization domains, and are then combined to form an improved global surface mapping.

Our approach solves an important practical problem and offers three novel scientific contributions. The first is a practical approach for local surface map editing, which we show, using both qualitative and quantitative metrics, provides better results than other state-of-the-art methods. The second involves a compact segmentation which results in a compromise between a low-distortion flattening and a low-distortion deformation when aligning the landmarks. The third is a new deformation approach, ILSCM, which preserves conformal energy better than other state-of-the-art methods, and that has theoretical guarantees preventing the introduction of fold-overs.

3.1 Surface Mapping Editing

As explained above, mappings computed by state-of-the-art methods are often globally good, but locally wrong in a few areas. We provide an approach to locally improve the surface mapping. The user typically inspects the mapping visually through texture transfer. In local regions where the mapping should be improved, the user sets landmarks at locations that should correspond on the pair of surfaces (Fig. 3.2a). Segmentation for the local adjustment of the surface mapping are exported (Fig. 3.2b). We edit the mapping by deforming parts of the meshes with respect to each other (Fig. 3.2c) to improve the alignment of the user-provided landmarks, and then we rebuild the mapping from the deformed parts (Fig. 3.2d).

Our main goal is to obtain a low distortion of the meshes at each phase of our approach. To deform the mapping by being able to control the amount of distortion, we conduct a planar parameterization of the meshes. Since the planar parameterization of smaller segments of a mesh leads to less distortion versus when computed for the entire mesh, we do a segmentation based on user-identified local *regions* where the mapping needs to be updated. Afterwards, we deform one segment with respect to the other in the planar parameterization space. The deformation is aided by our new ILSCM, ensuring that the deformation causes limited distortion

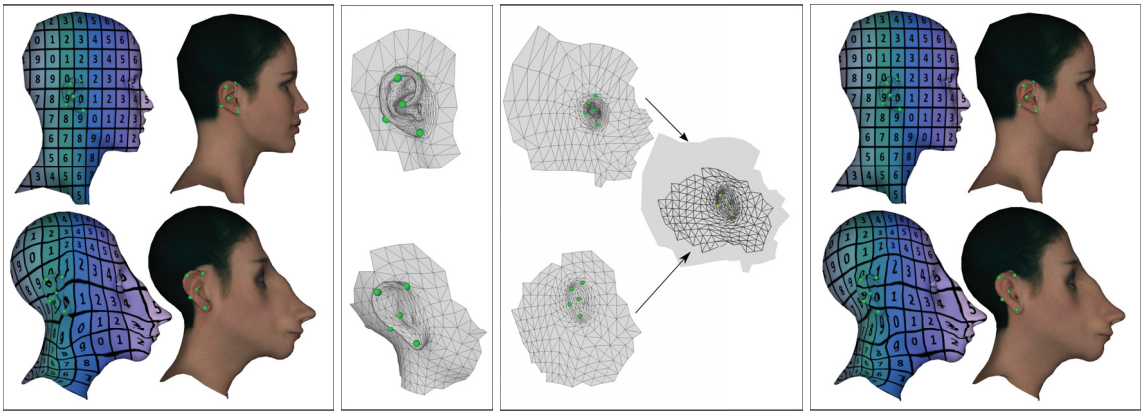


Figure 3.2 Overview of the approach

on the meshes. Finally, the mapping is extracted based on how segments overlap each other. Our approach has four key phases:

1. Initial mapping inspection (Sec. 3.1.1, Fig. 3.2a): The user inspects the mapping, identifies the mismatching regions of the meshes, and sets landmarks for each region.
2. Segmentation (Sec. 3.1.2, Fig. 3.2b): From the mismatching regions, an automatic segmentation is derived
3. Mapping deformation (Sec. 3.1.3, Fig. 3.2c): A localized deformation improves the alignment of the landmarks and the smoothness of the mapping.
4. Mapping extraction (Sec. 3.1.4, Fig. 3.2d): Partial mappings are extracted from the pairwise parameterization spaces of the segments, and aggregated to locally update the mapping.

3.1.1 Initial Mapping Inspection

Our approach works with input meshes A and B, along with a vertex-to-point surface mapping between the meshes. The mapping links each vertex of a mesh to a barycentric coordinate on the other mesh. It should be noted that our approach works regardless of the initial method used to establish the surface mapping, as long as a dense mapping is provided.

Given a mapping, the user will visualize it using texture maps to identify mismatching *regions*. These correspond to isolated zones of the meshes where the mapping is incorrect. For each region, the user sets corresponding landmarks on both meshes at locations that should match each other. The landmarks for each region i, $L_A(i) = \{l_A(i)_1, l_A(i)_2, \ldots, l_A(i)_k\}$ and $L_B(i) = \{l_B(i)_1, l_B(i)_2, \ldots, l_B(i)_k\}$, are expressed as barycentric coordinates on A and B, respectively.

3.1.2 Segmentation

The user provides hints where the mapping needs to be modified by setting pairs of landmarks on both meshes. In order to keep the map editing local, a segment is identified on both meshes where the map editing will be performed. Computing such a segment is not trivial as there are a number of requirements: the segment should be a single connected component with disk topology, should be compact, and should contain all the landmarks of the region i. While collaborating on the paper by Ramachandran *et al.* (2018), we discovered that using larger segments still results in a bijective mapping, but the size of the segment is also important. If the segment is too large we may lose locality, but if it is too small, we may introduce further distortion if the vertices need to move over a long distance. We assume that outside these segments, the mapping is satisfactory, and it can be used to set boundary conditions when deforming a segment with respect to the other to align the landmarks.

Our segmentation approach has three steps. In the first step (Fig. 3.3a), we grow an initial patch on the 3D surface from the landmarks, ensuring that it is one connected component, that it encloses all of the landmarks, as well as the positions corresponding to the landmarks from the other mesh. We flatten this patch in 2D (Fig. 3.3b), where we have more tools available to control the size and shape of the patch. In the second step, we compute a compact 2D patch from the convex hull of the landmarks in the 2D space (Fig. 3.3c), and ensure that we fill any artificial internal boundaries (internal "holes" with polygons from the full mesh missing). In the third step, we grow the compact patch from the previous step to allow enough room between the boundary and the landmarks (Fig. 3.3d), preparing the ground for a low-distortion deformation

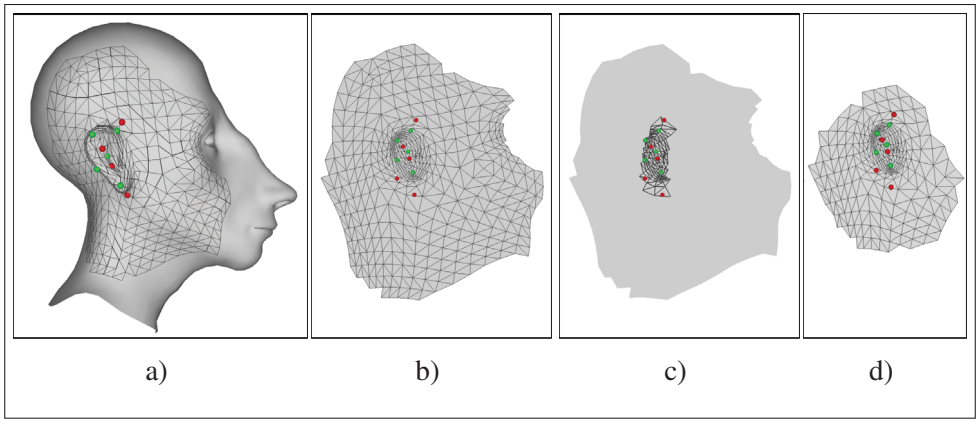


Figure 3.3 Steps of the segmentation phase

phase (Sec. 3.1.3). This segmentation is applied to each region of meshes A and B independently. We now explain in more detail the process as applied to one region i of mesh A, but the same is also conducted for mesh B and other regions.

Based on the mapping from mesh B to A, corresponding landmark positions are calculated on A for each landmark of $L_B(i)$, yielding $CP_{B\to A}(i) = \{cp_{B\to A}(i)_1, cp_{B\to A}(i)_2, \ldots, cp_{B\to A}(i)_k\}$. The goal of the first step is to extract an *initial patch* and to flatten it. To meet the two conditions of (1) having a single connected component and (2) containing all of the landmarks from $L_A(i)$ and $CP_{B\to A}(i)$, we will compute the union of face groups computed by identifying faces around each landmark $l_A(i)_j$ and around each corresponding position $cp_{B\to A}(i)_j$. For each, starting with the face containing the point, we iteratively add rings of faces until the group of faces contains at least half plus one of the landmarks from $L_A(i)$ and $CP_{B\to A}(i)$. The requirement to include half plus one of the landmarks ensures that when we combine the groups of faces, this initial patch meets the two conditions. This procedure results in a "disk with holes" topology, which is sufficient to flatten the patch using ABF++ (Sheffer *et al.*, 2005).

One disadvantage of the initial patch is that it can contain concavities, and even internal "holes". While concavities are not a problem for the flattening, they provide poor boundary conditions, making it harder to smoothly distribute the deformation error. From the initial patch of step one, the second step extracts a *compact patch* that surrounds the landmarks. To this end, we identify

the convex hull of the landmarks in the 2D parameterization space. Then, we only consider the faces which have at least one of their vertices within the convex hull (faces identified in black in Fig. 3.3c). The use of the convex hull results in a patch exempt of large concavities in its boundary. Nevertheless, depending on the meshes and the arrangement of landmarks, some of the initial patches have "holes" with polygons from the full mesh missing, creating artificial internal boundaries in the patch. Fig. 3.4 shows an example where the convex hull filling of the step 2 is important to avoid artificial boundaries. Fig. 3.4a shows flattened mesh with convex hull and region highlighted in red contains an artificial internal boundary. Fig. 3.4b zoom-in and Fig. 3.4c 3D mesh, both showing the artificial boundary and missing faces. We add the missing faces by analyzing the inner boundaries (holes). Filling the whole by adding the missing faces from the full mesh has the advantage of preventing unwanted deformation that would result from such artificial boundaries, and it ensures that there are no internal areas within the region where the mapping would not be adjusted. The third step tries to balance the conflicting

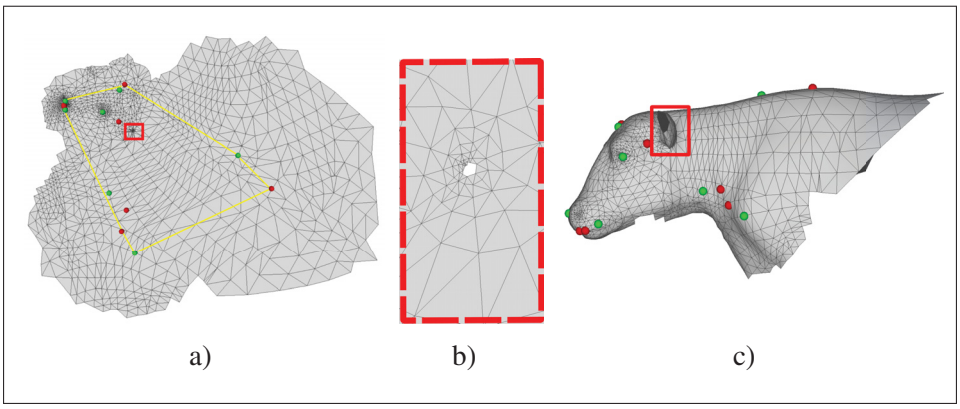


Figure 3.4 The convex hull filling of the step 2 is important to avoid artificial boundaries

goals of having a small versus a large patch. As can be seen in Fig. 3.5, the larger the patch, the greater the stretch distortion (Sander, Snyder, Gortler & Hoppe, 2001) between the patch in 3D and after flattening to 2D. The labels 1, 2, and 3 correspond to patch sizes in ascending order. The distortion would be even higher if flattening the whole mesh (Fig. 3.6). Conversely, a smaller patch means that the landmarks are closer to the boundary, and the deformation that aligns the landmarks will induce more distortion to the triangles between the boundary and

the landmarks. We thus want to grow the compact patch to ensure that there is enough room around each landmark and corresponding landmark position pairs to diffuse the distortion from the deformation phase. We also know that as we get closer to the landmarks, we are getting closer to the areas where the mapping is wrong, and as such, extending outwards is necessary in order to have a good boundary condition for the deformation. Regarding how far away the patch should be extended, we use a distance proportional to the geodesic distance (on the 3D mesh) between the landmark and its corresponding landmark position, adding faces that are within that distance from each landmark of the pair. We compared the distortion between the 3D patch and the 2D patch for 1 to 10 times the geodesic distance.

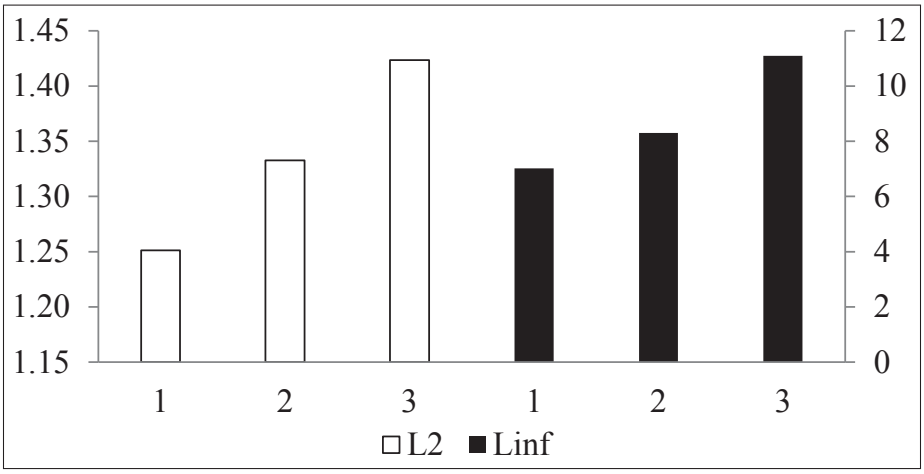


Figure 3.5 L_2 and L_{inf} graphs showing the average stretch distortion of patches between their configuration in 3D and after flattening by ABF++

Fig. 3.7 shows a pattern where very small patches do not have enough room to move the landmarks without high distortion. At the same time, patches that are too large also exhibit large distortion because of the flattening from 3D to 2D. As can be seen in Fig. 3.7, a good compromise between the two sources of distortion is around two times the geodesic distance. Accordingly, we add triangles around each landmark until we reach a geodesic distance bounded to two times the geodesic distance (on the 3D mesh) between the landmark and its corresponding landmark position.

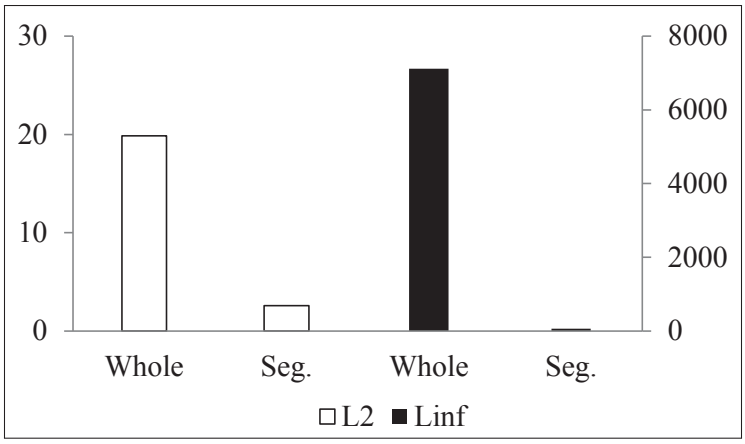


Figure 3.6 Flattening the segmented patch results in significantly lower L_2 and L_{inf} stretch distortion as compared to flattening the whole mesh

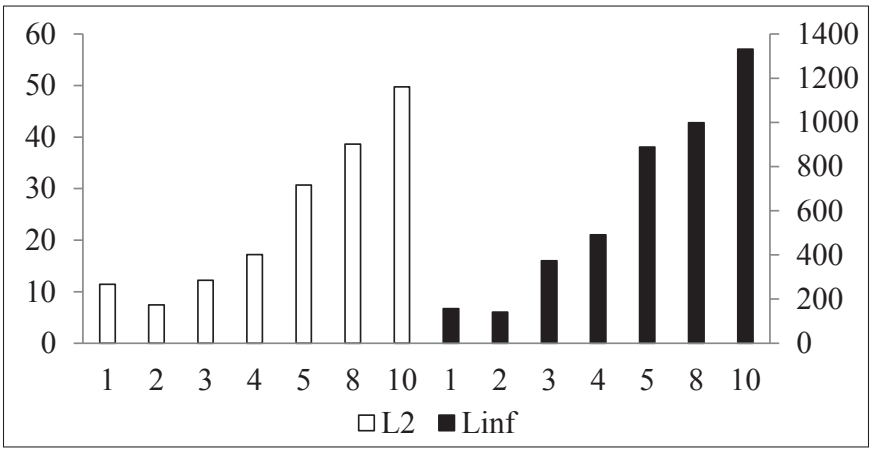


Figure 3.7 L_2 and L_{inf} graphs showing the average distortion with respect to different geodesic distances used in step three

It is necessary to apply steps one and two, because step three alone could lead to disconnected components or artificial internal boundaries that do not exist in the mesh, but that exist in the patch because of the patch growth process (Fig. 3.4). The sequence of steps one to three provides the final segments from A, $\{A_1, A_2, \ldots, A_n\}$, and the same for mesh B, yielding $\{B_1, B_2, \ldots, B_n\}$. These final segments are flattened through ABF++ and we refer to them as $\left\{\widetilde{A_1}, \widetilde{A_2}, \ldots, \widetilde{A_n}\right\}$ and $\left\{\widetilde{B_1}, \widetilde{B_2}, \ldots, \widetilde{B_n}\right\}$. We selected ABF++ to flatten our segments as we can assume it will yield a parameterization exempt of flipped triangles (injective).

For our other paper (Ramachandran *et al.*, 2018), we used a semi-automatic segmentation where the user had to manually select landmarks and closed paths to drive the segmentation process and therefore cut the meshes into the segmented parts. In comparison, the paper exposed in this chapter is designed with a more automatic process where the user only has to identify groups of landmarks that will correspond to the isolated region of the mesh where the editing will occur, and the rest of the process is automatic.

3.1.3 Mapping Deformation

Our main observation is that the initial surface mapping is globally adequate, but wrong in localized regions. With this assumption, we line up the boundary of the regions by relying on the surface mapping. We will then deform the interior to align the landmarks while keeping a low distortion.

As we have two segmented meshes, there are two ways to align the landmarks: deform $\widetilde{A}_i$ on $\widetilde{B}_i$ or $\widetilde{B}_i$ on $\widetilde{A}_i$. We select the one with the lower L_2 distortion (Sander *et al.*, 2001) (between the segment in 3D and 2D), keeping it fixed and deforming the other. Here, we will explain the deformation of $\widetilde{B}_i$ with respect to $\widetilde{A}_i$, but the deformation of $\widetilde{A}_i$ with respect to $\widetilde{B}_i$ proceeds in the same way. We deform $\widetilde{B}_i$ in order to implicitly adjust the mapping by applying an energy minimization. This is achieved by using positional user constraints (E_L – aligning the landmarks, E_B – aligning the boundary) coupled with a distortion preventing regularization (E_D – globally deforming $\widetilde{B}_i$ with low distortion), leading to the following equation:

$$E(V) = E_L(V) + E_B(V) + E_D(V). \tag{3.1}$$

The user constraints are enforced by soft constraints as follows:

$$E_L(V) = \lambda \sum_{j=1}^{k(i)} \left\| l_{aj} - (v_{(j,1)}\beta_{(j,1)} + v_{(j,2)}\beta_{(j,2)} + v_{(j,3)}\beta_{(j,3)}) \right\|^2 \tag{3.2a}$$

$$E_B(V) = \lambda \sum_{j \in \Omega(\widetilde{B}_i)} \left\| v_j - \text{map}(v_j) \right\|^2, \tag{3.2b}$$

where $k(i)$ is the number of landmarks of segment i; l_{aj} are the landmarks on $\widetilde{A}_i$; vertices $v_{(j,1)}$, $v_{(j,2)}$, and $v_{(j,3)}$ correspond to the three vertices of the triangle on $\widetilde{B}_i$ containing the related landmark l_{bj}; and $\beta_{(j,1)}$, $\beta_{(j,2)}$, and $\beta_{(j,3)}$ are the barycentric coordinates. We use $\Omega(\widetilde{B}_i)$ to denote the set of vertices on the boundary of $\widetilde{B}_i$, and $\text{map}(v_j)$ to denote the corresponding position of v_j on $\widetilde{A}_i$ based on the mapping. The energy E_B pulls the vertices of the boundary of $\widetilde{B}_i$ to the positions on $\widetilde{A}_i$ where they correspond given the mapping. When λ is small, the map will be injective, but the constraints are generally not satisfied (ultimately, if $\lambda = 0$, $\widetilde{B}_i$ stays the same). Conversely, when λ is large, the user constraints are satisfied, but flips may be introduced ($\lambda = \infty$ corresponds to using hard constraints).

3.1.3.1 Deformation Energy

An ideal deformation energy $E_D(V)$ must meet three criteria: preserving the shape, maintaining the injectivity of the mapping (i.e., no flipped triangles), and satisfying the user constraints as much as possible. During our collaboration with Ramachandran *et al.* (2018), we discovered that the proposed cage deformation method does not provide a low-distortion deformation since it was not minimizing a distortion energy. On the other hand, even if it aligns interior vertices naturally, but there is also a cost to it. In scenarios where the landmark correspondences are flipped, the cage mesh triangles will be flipped even after resolving ambiguous cages. This will cause the interior mesh triangles to overlap. The approach presented in this chapter improves both on the distortion and on the triangle flipping. We experimented with several E_D such as: LSCM (Lévy *et al.*, 2002), LIM (Schüller *et al.*, 2013), SLIM (Rabinovich *et al.*, 2017), and KP-Newton (Golla *et al.*, 2018). Each of the energies has a number of pros and cons. LSCM preserves the shape the best, but tends to introduce flips, as illustrated in Fig. 3.8c. LIM, SLIM, and KP-Newton on the other hand, guarantee injectivity (no flips), but introduce more distortion (between $\widetilde{B}_i$ before and after deformation) than LSCM. The graph in Fig. 3.9 illustrates these observations: LSCM has the least distortion, but flipped triangles would destroy the injectivity of the mapping. LIM, SLIM, and KP-Newton have no flips, but overall, they have more distortion

as compared to LSCM. LIM minimizes a joint energy where one term optimizes the distortion and the second term optimizes the flips. In such joint optimization frameworks, no one term may be close to a minimum of its own, as shown in Fig. 3.9, where the results from LIM are worse in terms of the distortion energy than LSCM.

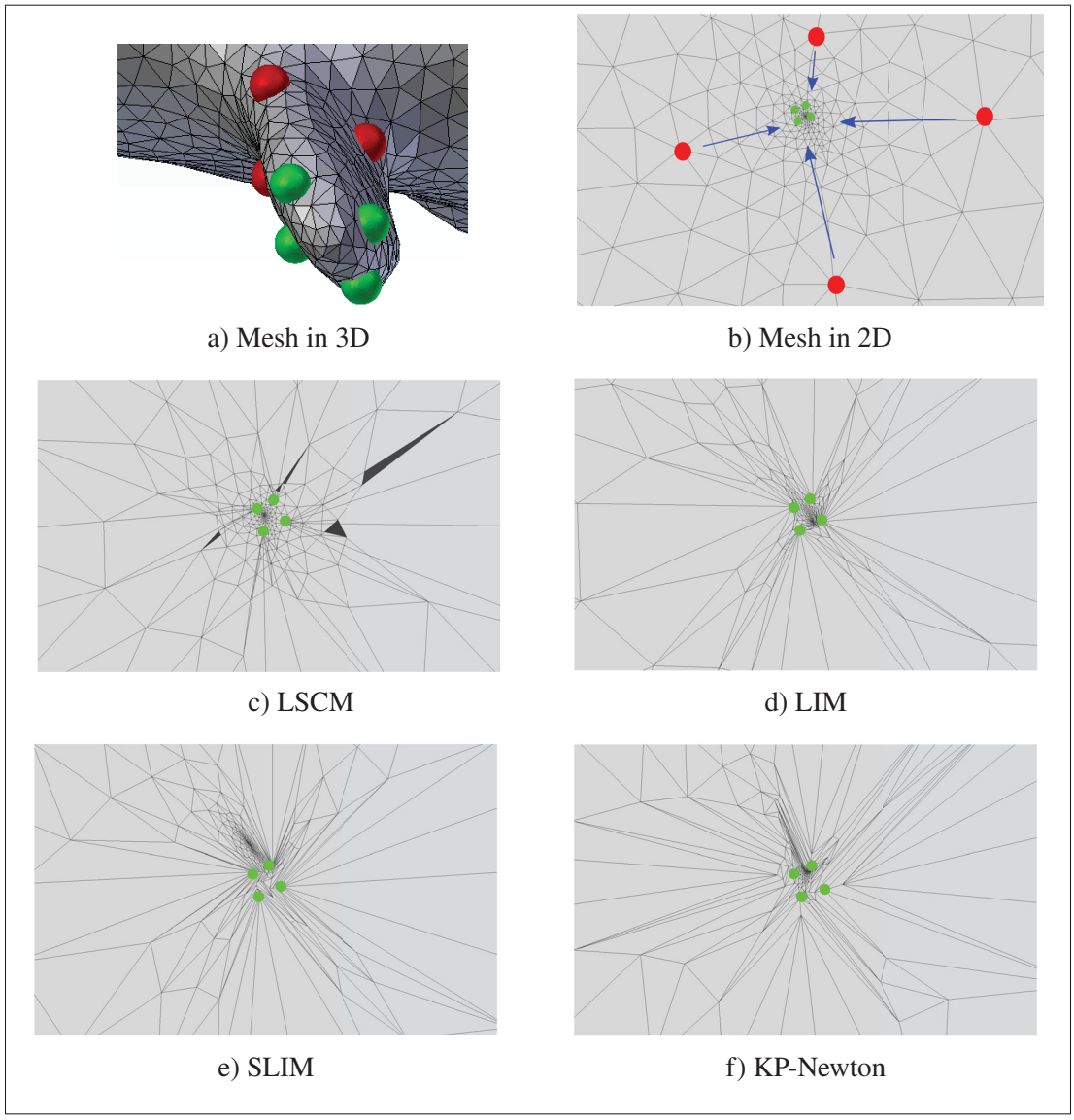


Figure 3.8 This is an example where we want to adjust the mapping by moving the red landmarks to the positions of the corresponding green landmarks

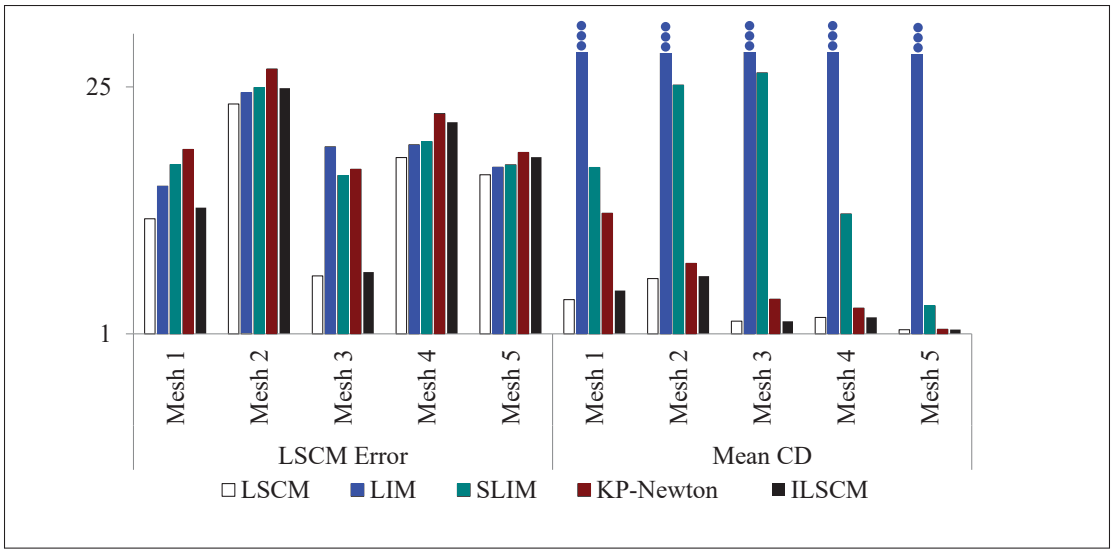


Figure 3.9 The graph compares results in terms of the residual error of LSCM energy (Lévy *et al.*, 2002) and Conformal Distortion (Lipman, 2012) for the deformation of five different 2D meshes

Since all these methods have shortcomings, we propose an approach that bridges the gap between the shape preservation of the original LSCM formulation with the injectivity preservation of LIM, SLIM, and KP-Newton. Our approach, Iterative LSCM (ILSCM), is a different approach where we iteratively optimize to decrease $E(V)$, while preventing flips from occurring. ILSCM performs iterative LSCM steps. The first iteration uses the cotangent weights from segment $\widetilde{B}_i$. The deformed segment from the first iteration is then used to set the weights for the second iteration, and so on. At each iteration, if a triangle flip is detected, we take smaller step by decreasing the value of λ and redo the same iteration. This way, we are guaranteed to eventually find a λ that prevents flips from occurring.

We will now explain how we adaptively adjust λ to guarantee that we have no flips, while making as much progress as possible toward achieving the user constraints. In order to measure if the constraints are satisfied, we consider the *initial* maximal distance between any landmark and corresponding landmark position pair $\text{dist}_0 = \max_j \left\| l_A(i)_j - cp_{B\to A}(i)_j \right\|$, and iterate until the current maximal distance is below the threshold $\epsilon = \text{dist}_0/250$. Appendix I demonstrates that since the progression of landmarks is continuous with respect to λ, the approach will always

find a λ that prevents having any flips and that enables progress toward the user constraints. The progress could asymptotically stop, but in all cases, we are guaranteed to prevent triangles from flipping and we limit the mesh distortion. For the mapping adjustment application, all of the examples we tested converged to meet the user constraints.

A larger λ will converge faster, but increases the likelihood of flipped triangles (Fig. 3.10, black bars). A small λ decreases the probability of flipped triangles, but increases the number of iterations needed to satisfy the user constraints. In Fig. 3.10, the horizontal axis labels present λ and the number of iterations (λ, #iter). The black bars highlight results with flipped triangles. As can be seen, whether using a small or large λ, the conformal residual is almost the same and it plateaus for smaller values of λ. Consequently, even in the theoretical case where our approach would take very small incremental steps, the solution remains valid in the sense that it meets the constraints and the conformal residual remains close to the solution with larger steps. In our experiments, we start with $\lambda = 1000$. After each iteration, we automatically detect if there are flipped triangles, and if so, we redo the deformation of the iteration with $\lambda = \lambda/2$. Fig. 3.11a demonstrates that the movement of a landmark is continuous with respect to different values of λ. The black point is the initial position of the landmark, the white point is the constraint position, and the grey points show landmark positions corresponding to different values of λ (computing a single iteration from the initial configuration for each). Fig. 3.11b further shows that even with small values of λ, we make progress toward satisfying the constraints. We also see that it is to our advantage to begin with a large value of λ to reduce the number of iterations before convergence

3.1.4 Mapping Extraction

As the last phase of our approach, we update the surface mapping between A and B from the planar parameterizations. We first extract the mappings from each pair $(\widetilde{B}_i, \widetilde{A}_i)$. Then, we aggregate and transfer them to A and B. With the mapping being expressed as barycentric coordinates on the other mesh, we can update it by simply getting the barycentric coordinates of vertices from $\widetilde{B}_i$ to faces of $\widetilde{A}_i$ and vice-versa.

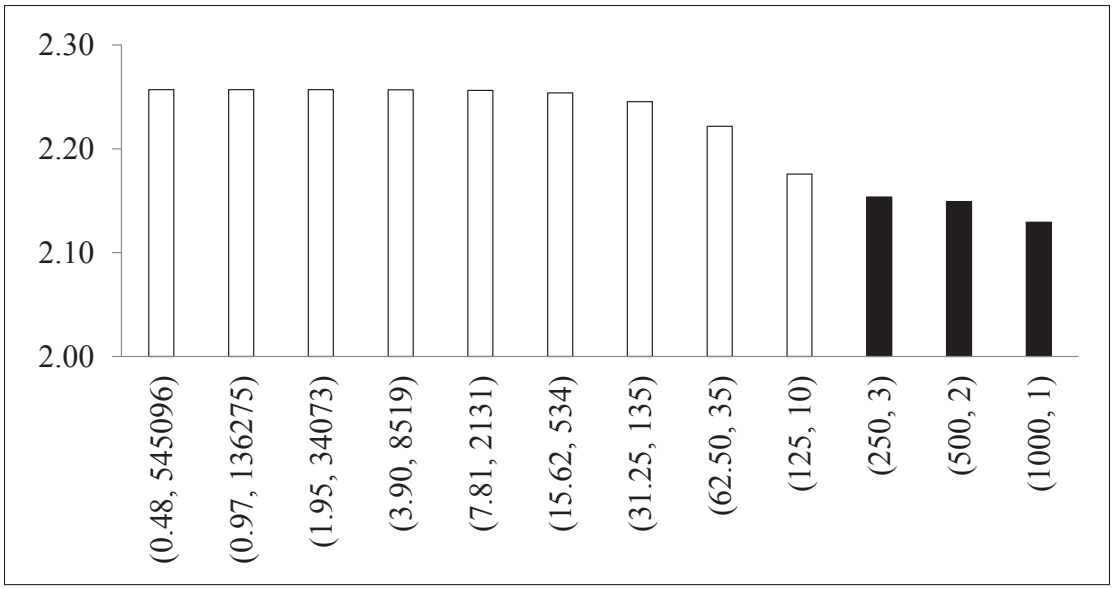


Figure 3.10 This graph compares results in terms of the residual error of LSCM energy (Lévy *et al.*, 2002) for fixed values of λ

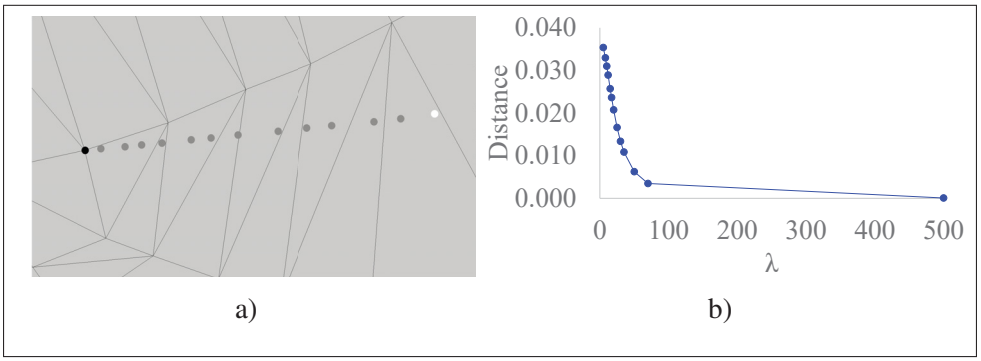


Figure 3.11 This figure shows the impact of λ for an example with a single landmark

3.2 Results

We validate our approach with various cases of faces, as well as with a wider range of objects with different morphologies from different data sets and artist contributions (see Table 3.1). Experiments are presented based on different initial mapping methods: orbifold tutte embeddings (Aigerman & Lipman, 2015), Elastiface (Botsch & Sorkine, 2008), deformation transfer (Sumner & Popović, 2004), functional mapping (Nogneng & Ovsjanikov, 2017),

weighted averages (Panozzo *et al.*, 2013), and joint planar parameterization (Ramachandran *et al.*, 2018). The number of landmarks and segments is proportional to the quality of the initial surface mapping and the complexity of the objects (see Table 3.1). We evaluate the capabilities of our approach based on a qualitative evaluation by visual inspection and a quantitative evaluation based on geodesic distance.

3.2.1 Qualitative Evaluation

Our approach prevents flipped triangles, and, essentially, it preserves the shape, while satisfying the user constraints. Furthermore, it distributes the deformation error more uniformly across the mesh surface. As can be seen in Fig. 3.9, our distortion energy is lower than SLIM, is often lower than LIM, and is only slightly greater than LSCM. We believe that this optimization strategy is more suited to this type of problem than a joint optimization strategy. Fig. 3.12 compares LSCM, LIM, SLIM, and KP-Newton to our ILSCM (iterated 257 times and final λ was 31.25) for the example from Fig. 3.8. ILSCM distributed errors more uniformly over the whole deformed mesh, as compared to LIM, SLIM, and KP-Newton. The accompanying video shows how our iterative approach progressively conducts the deformation, in comparison to LSCM, LIM, SLIM, and KP-Newton. The meshes we deform in the video are the same as some of the examples from the LIM paper (Schüller *et al.*, 2013). For a fair comparison, we perform SLIM, LIM, and KP-Newton, all using the LSCM energy (Lévy *et al.*, 2002) for all the examples in the chapter as well as the video.

For LIM, we apply a 1E−12 barrier weight, which is sufficient to prevent flips. We experimented with barrier weights of LIM ranging from 1E−4 to 1E−20. Barrier weights smaller than 1E−12 had an imperceptible impact, while those equal to or lower than 1E−20 did not converge. For each deformation energy, we experimented with two different initial states: weights from the 3D triangles and weights from the flattened $\widetilde{B}_i$. The distortion between $\widetilde{B}_i$ before and after deformation was lowest when deforming using the weights from the flattened $\widetilde{B}_i$. We thus used the weights from the flattened $\widetilde{B}_i$.

Table 3.1 Information about the meshes, data sets, number of landmarks, segments, and computation time. Method 1: Ramachandran *et al.* (2018), Method 2: Aigerman & Lipman (2016), Method 3: Sumner & Popović (2004), Method 4: Nogneng & Ovsjanikov (2017), Method 5: Panozzo *et al.* (2013), Method 6: Zell & Botsch (2013). Data set 1: SHREC07 (Veltkamp & ter Haar, 2007), Data set 2: FAUST (Bogo *et al.*, 2014), Data set 3: Character Generator, Data set 4: SCAPE (Anguelov *et al.*, 2005), Data set 5: TOSCA (Bronstein *et al.*, 2006)

Meshes	**Mapping Method**	**Data set**	**Seg.**	**Mesh # Faces**	**Seg. # Faces**	**# Land-marks**	λ	**# Iter.**	**Seg. Time (s)**	**Def. Time (s)**
Fig. 3.21: Fish	Method 1	Data set 1	Fins	8-20K	1K, 1K	2, 3	31.25, 62.5	128, 101	3.23	4.64
Fig. 3.21: Aircrafts	Method 1	Data set 1	Stabilizer	11K	3K	4	125	40	4.13	1.30
Fig. 3.13: Man	Method 2	Data set 2	Head	13K	5K	4	1000	1	2.37	1.22
Fig. 3.15: Ilana-Eagle	Method 3	Data set 3	Neck, Nose	6-10K	1-2K	3, 7	1000, 1000	1, 1	4.55	1.12
Fig. 3.16: Oldman-Sam	Method 3	Data set 3	Ear	6-11K	2K	5	125	121	1.94	3.61
Fig. 3.1 and 3.18: Man	Method 2	Data set 4	Head	24K	11K	4	1000	27	2.91	8.67
Fig. 3.19: Horse-Cow	Method 4	Data set 1	Head	8-11K	7K	7	125	6	4.14	1.27
Fig. 3.19: Lamb-Dog	Method 4	Data set 1	Head	3-7K	1K	6	62.5	31	1.36	0.84
Fig. 3.22: Tiger Woman	Method 5	Data set 3	Nose	3K	1K	2	500	2	0.22	0.16
Fig. 3.20: Bird	Method 2	Data set 1	Tail	10-15K	3K	4	250	46	1.48	1.50
Fig. 3.20: Wolf	Method 2	Data set 5	Head	8K	3K	4	1000	6	2.09	1.27
Fig. 3.23: Ilana-Badger	Method 3	Data set 3	Ear, Nose, Mouth	1-6K	1K, 1K, 1K	5, 8, 7	1000, 125, 1000	1, 6, 1	4.76	0.99
Fig. 3.24: Man-Curve	Method 6	Artist and Data set 3	Thumb	2-8K	854	2	1000	1	1.35	0.20

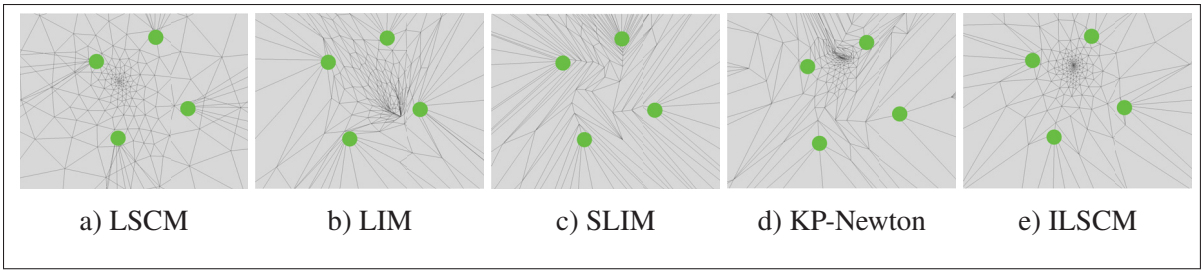
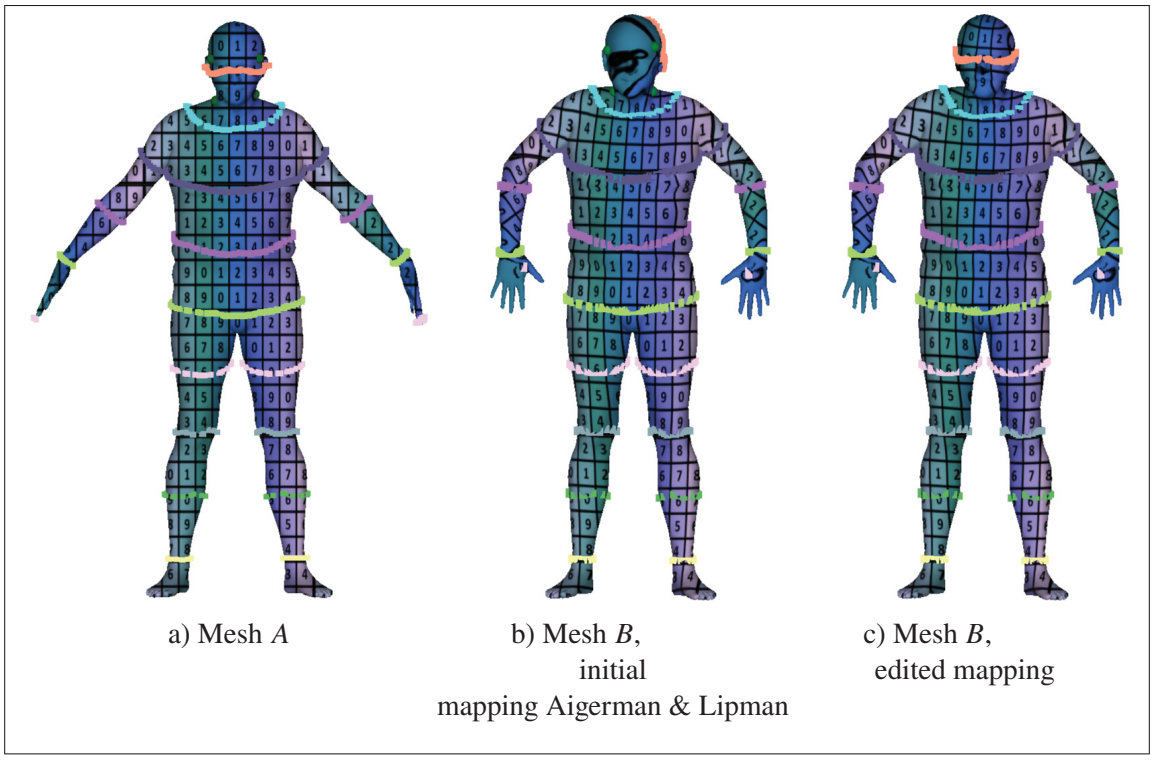


Figure 3.12 Different methods all minimizing the LSCM energy. ILSCM does not generate any flips, and additionally, it better preserves the shape of the triangles compared to LIM, SLIM, and KP-Newton

Visual inspection of results is a common form of validation for mapping problems. We use a visualization method based on texture transfer. We copy texture coordinates from one mesh to the other using the mapping, setting the uv coordinates of a vertex to the barycentric interpolation of the uv coordinates of the other mesh. For this visualization, we used two different types of textures. The first type was a grid texture. Figs. 3.1, 3.2, 3.13, 3.15, 3.16, as well as Figs. 3.19-3.23 qualitatively show that we obtain considerably better mappings using our editing approach.

An important assumption of our approach is that we can edit the mapping locally. This implies that it is important to have a smooth transition at the boundary of the regions where we conduct local editing. Fig. 3.14 shows a typical example of the smoothness of our edited mapping across the boundary of the segmented region. The accompanying video also compares the transition of the mapping across the boundary by transferring texture from mesh A to mesh B using both initial mapping and edited mappings for the test cases of Fig. 3.13, Fig. 3.21 (top row), and Fig. 3.20.

For the specific case of faces, we use realistic facial textures, making it easier to highlight important semantic facial features. These features are derived from three important considerations: modeling, texturing, and animation. A realistic facial texture is often enough to highlight modeling and texturing issues. Problems around the nose (Fig. 3.15 and 3.23), lips (Fig. 3.15 and 3.23), and ears (Fig. 3.16 and 3.23) are easy to spot with such a texture visualization approach. These examples show cases that are ideal for our approach: the initial mappings are globally good, with few local misalignments. Instead of solving for the mapping globally, our



a) Mesh A b) Mesh B, initial mapping Aigerman & Lipman c) Mesh B, edited mapping

Figure 3.13 Isopoints and grid texture visualization showing the local improvement of the initial mapping in the head region

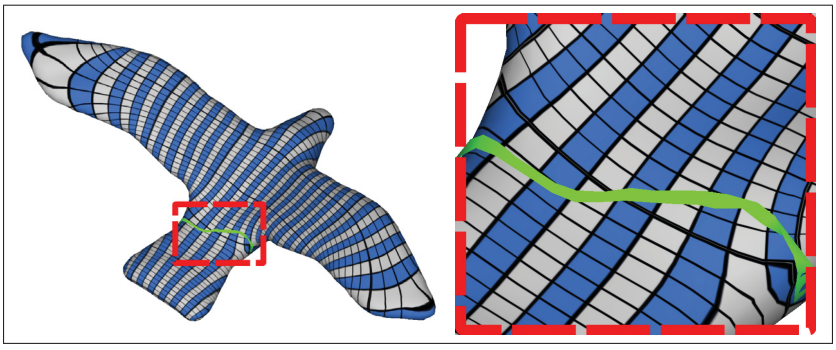
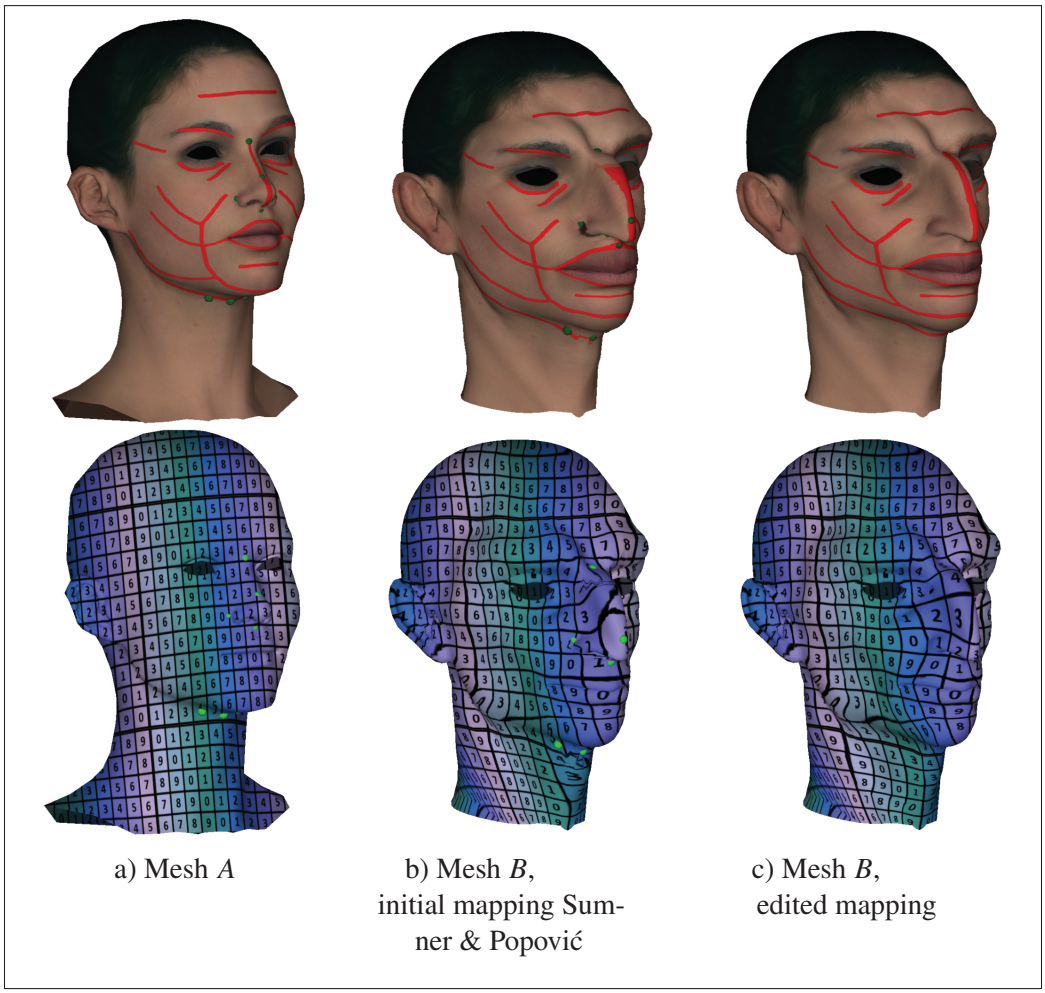


Figure 3.14 Figure showing that our edited mappings are smooth across the boundary of the edited regions

approach provides a local solution for these specific semantic regions. For facial animation, other features need to be identified in the textures. Accordingly, some of our texture visualizations use a set of curves that are positioned relative to the areas that deform during animation, based

on the facial anatomy. Fig. 3.15 illustrates the improvement in the correspondence of these animation-related features as compared against the initial surface mapping.



a) Mesh *A*

b) Mesh *B*, initial mapping Sumner & Popović

c) Mesh *B*, edited mapping

Figure 3.15 Realistic facial textures and grid textures are used for qualitative evaluation. Improvement of the initial surface mapping in the nose, mouth, and neck regions is apparent through improved red curves of c top

Our approach assumes that the segments can be flattened to 2D without any flipped triangles. While the hypothesis is essential to get injective mappings, our approach is still robust to cases where the flattened segments would contain flipped faces. Meshes used in the industry often exhibit small discrepancies such as cracks, holes, and handles. While collaborating on the paper by Ramachandran *et al.* (2018), we experimented that some geometric correspondence

methods are able to handle different genera. Even though this yields non-bijective mappings, the non-bijective region can be quite localized and is not detrimental to the majority of the map which remains bijective. Fig. 3.16 presents such a case where one of the meshes is of a different genera (contains two handles in the ear region). Although it is not possible to get injective mappings when dealing with different genera, our approach behaves robustly: it can improve the mapping in the region with different genera and it does not degrade the mapping in the edited region nor in its vicinity. Furthermore, even if it is not possible to achieve injective mappings in such cases, our edited mappings have reasonable properties: the mapping from the lower genera ($A \rightarrow B$) is injective, and the mapping from the higher genera ($B \rightarrow A$) is surjective.

3.2.2 Quantitative Evaluation

While the qualitative evaluations of Sec. 3.2.1 demonstrate that our approach results in clear improvements, we also quantitatively measure how our approach improves the mappings. We first use the same process as in the dissertation of Kim *et al.* (2011) in order to measure the accuracy of the surface mapping. Their method transfers vertex positions to the other mesh using the mapping under evaluation and a ground truth mapping. It then computes the geodesic distances from the corresponding positions. Fig. 3.17 shows the error of the initial mapping (Aigerman & Lipman, 2016) as compared to the mapping after our editing approach. The comparative evaluation shown here relies on the ground truth mapping from SCAPE (Anguelov *et al.*, 2005) (Fig. 3.17a) and TOSCA (Bronstein *et al.*, 2006) (Fig. 3.17b) data sets. We can see that applying our approach improves the mapping in the related regions without causing a degradation of the overall mapping.

Another way to measure the quality of a mapping is to morph a mesh into the shape of the other using the mapping. Then, we evaluate the mapping by computing L_2 and L_{inf} distortion between the mesh and the morphed mesh to estimate the stretch which occurs in the mapping-based morphing process. Fig. 3.18 shows the morphing of mesh A into mesh B using both the initial and new mappings. With our updated mapping (Fig. 3.18d), the vertices of the head A are pulled back to the correct place. This has the advantage of mapping the right density of vertices where

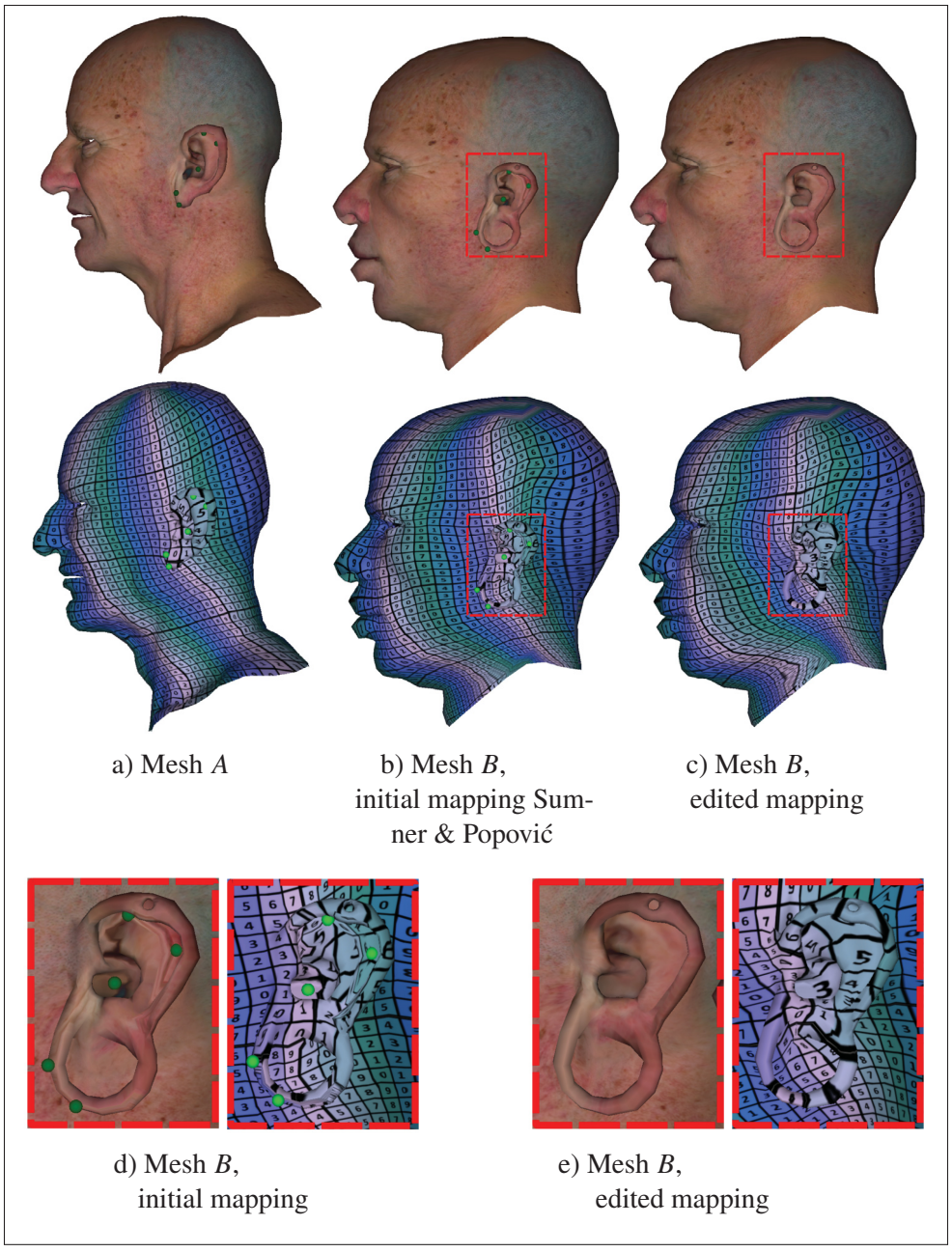


Figure 3.16 Mapping mesh *A* to mesh *B* with different genera leads to distortion in the ear region (outlined in red). Our approach reduces texture transfer issues locally even in regions with higher genera

needed, which is very important for morphing and in any transfer related to animation attributes (e.g., bones, vertex weights, and blend shapes). Table 3.2 illustrates an evaluation of the quality of the edited mapping in comparison to the initial mapping. It shows that our edited mapping is

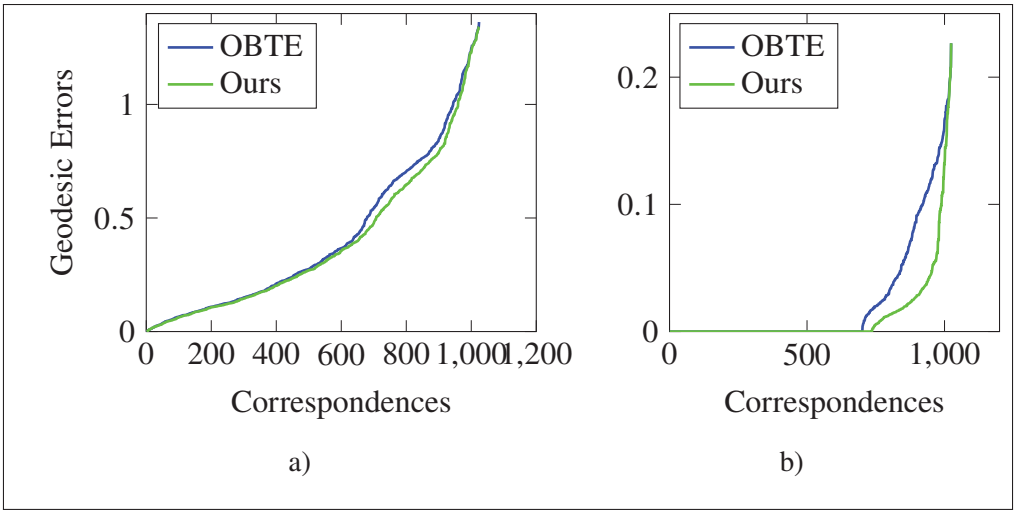


Figure 3.17 Comparison of the OBTE initial mapping (Aigerman & Lipman, 2016) and our edited mapping. a and b correspond to the examples of Figs. 3.18 and 3.20 (top row), respectively

as good as or better than the initial mapping when considering the distortion of the morphed mesh. We can see that there is a single case where this measurement of distortion is slightly higher after the map is edited. Even in this case, while the distortion is slightly higher, the edited mapping is clearly superior, as can be seen in Fig. 3.13.

Table 3.2 Distortion ratio of the meshes morphed through the edited mapping as compared to the initial mapping. The L_2 and L_{inf} ratios correspond to $L_{2_{edited}}/L_{2_{initial}}$ and $L_{inf_{edited}}/L_{inf_{initial}}$, respectively

# Figure	3.1	3.13	3.15	3.16	3.18	3.19	3.19	3.21	3.21	3.22	3.20	3.20	3.23	3.24
L_2 ratio	0.1	1.0	0.7	0.9	0.0	0.6	0.0	0.0	0.0	0.9	0.9	0.9	0.4	0.5
L_{inf} ratio	0.1	0.9	0.7	0.9	0.0	0.7	0.1	1.0	1.0	0.9	0.9	0.9	0.4	0.6

3.2.3 Comparison

We performed a qualitative comparison of the mapping editing versus the methods of Ezuz & Ben-Chen (2017). We also did comparisons using LIM, SLIM, KM-Newton, and ILSCM to conduct the mapping editing. Furthermore, we compared local editing to global editing using the method of Panozzo *et al.* (2013) and the joint planar method (Ramachandran *et al.*, 2018).

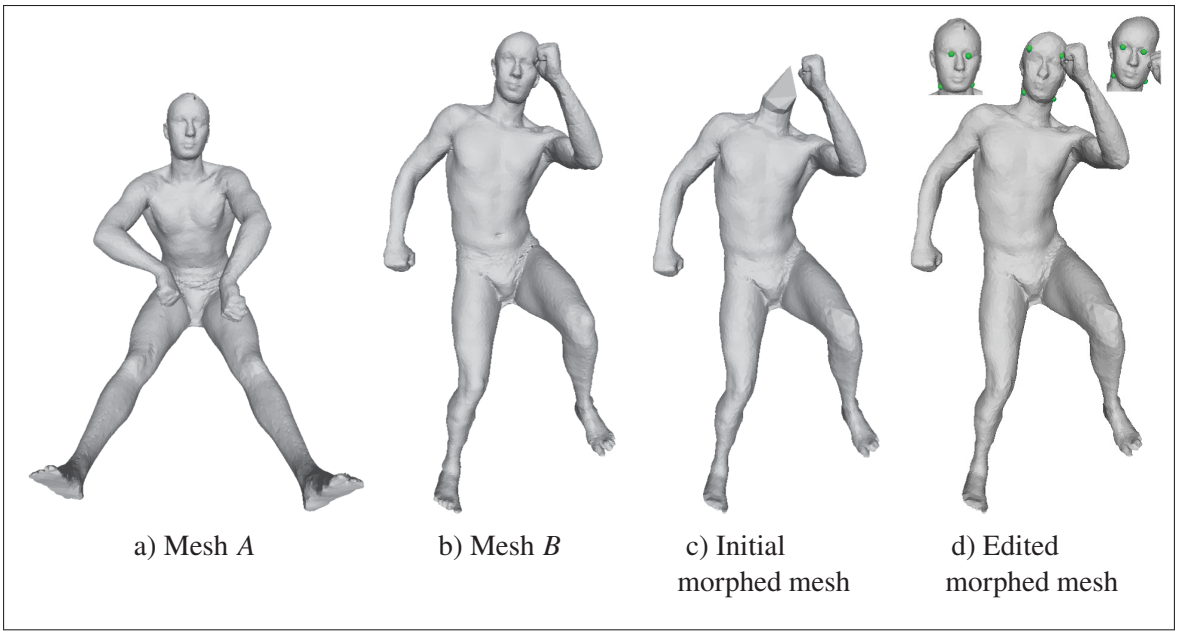


Figure 3.18 a-b: Initial meshes. c With the initial mapping (Aigerman & Lipman, 2016), mesh *A* morphed to mesh *B* is unable to recreate the head. d The two faces on top of the figure show initial meshes and their landmarks used by our approach to improve the mapping

For the comparison to the method of Ezuz & Ben-Chen (2017), we established an initial mapping using a state-of-the-art functional mapping method (Nogneng & Ovsjanikov, 2017). Note that we use the raw functional map, without the high-dimensional iterative closest point post-process refinement (Ovsjanikov *et al.*, 2012). Fig. 3.19 compares the mappings improved using our approach and the method of Ezuz & Ben-Chen (2017) (which improves the mapping without any landmark). Note how the added control of the landmarks provides a significantly improved mapping, exactly where intended.

Fig. 3.20 presents results when LIM, SLIM, KP-Newton, and ILSCM are used to conduct the mapping editing. The comparison through texture transfer visualization shows that ILSCM is superior in adjusting the mapping as compared to LIM, SLIM, and KP-Newton. ILSCM pulls back the wolf muzzle to the correct place and improves the mapping of the bird tail. The accompanying video also compares LIM, SLIM, KP-Newton, and ILSCM in editing the mapping for the test case of Fig. 3.13.

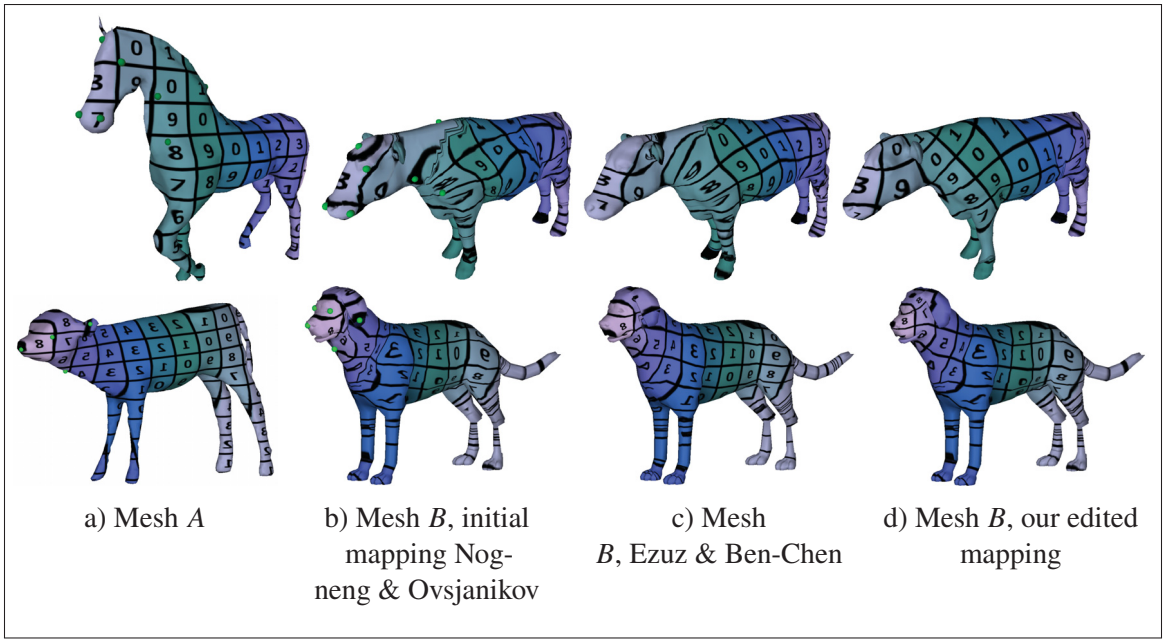


Figure 3.19 Qualitative comparison with the deblurring method. Our approach allows more control, and significantly improves the mapping in the head area

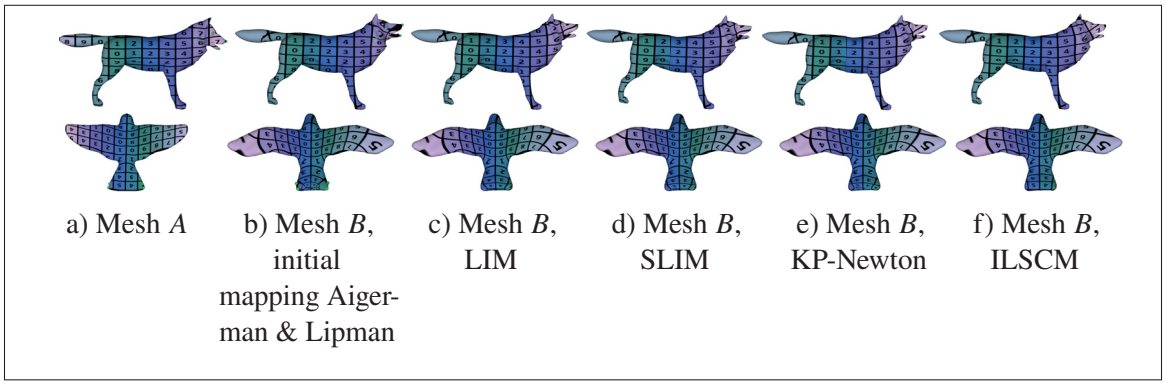


Figure 3.20 Qualitative comparison between LIM, SLIM, KP-Newton, and ILSCM to conduct the mapping deformation (minimizing the LSCM energy for all methods)

Adjusting the mapping globally requires having the initial constraints, the initial parameters of the method, and the method itself, which is constraining. In addition, some mapping methods, such as that of Nogneng & Ovsjanikov (2017), do not let the user guide the process with landmarks, while others, such as OBTE (Aigerman & Lipman, 2016), only support a fixed number of landmarks (three or four landmarks for OBTE (Aigerman & Lipman, 2016)), which

will be insufficient in many cases. Furthermore, we believe that it is advantageous to ensure that changes occur locally, avoiding unexpected changes elsewhere in the mapping. Figs. 3.21 and 3.22 compare our approach to two different methods, joint planar (Ramachandran *et al.*, 2018) and WA (Panozzo *et al.*, 2013), respectively. For the these methods, the mapping was computed with an initial set of blue landmarks, and the improved mapping was obtained by adding new green landmarks to the initial set. Conversely, our approach only uses the new landmarks to improve the mapping. Fig. 3.21c (top row) shows that solving for the mapping globally is sometimes as effective as solving it locally. Conversely, Fig. 3.21c (bottom row) shows that improving the mapping globally introduced artifacts on the fish head as compared to our local refinement (Fig. 3.21d bottom row), which is exempt of such artifacts away from the edited region.

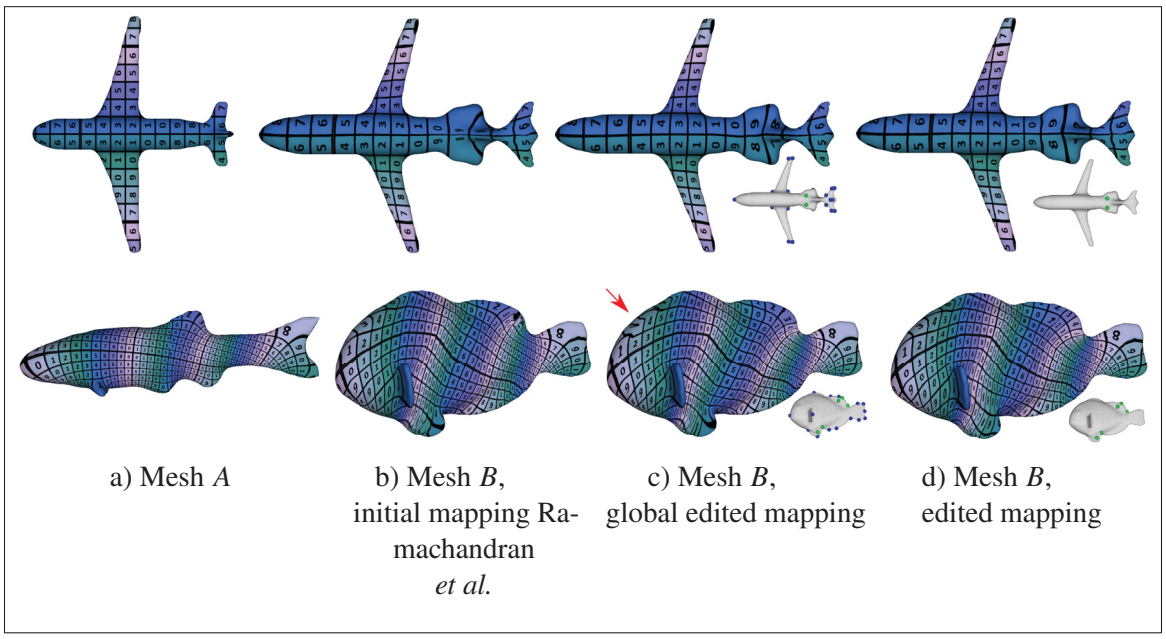


Figure 3.21 Qualitative comparison of adjustment of the mapping around the aircraft stabilizer (top row) and the fins (bottom row). c Adjustment of the mapping globally. d Adjustment of the mapping locally with our approach

Fig. 3.22 compares the mappings improved using our approach as compared to solving globally using the WA method of Panozzo *et al.* (2013). We established an initial mapping using the WA method. Afterwards, with the WA method, we added two additional landmarks to improve the

initial mapping. For our approach, we only consider the two new landmarks in improving the mapping. It can be seen in Fig. 3.22 that editing the mapping locally was beneficial for this test case as well.

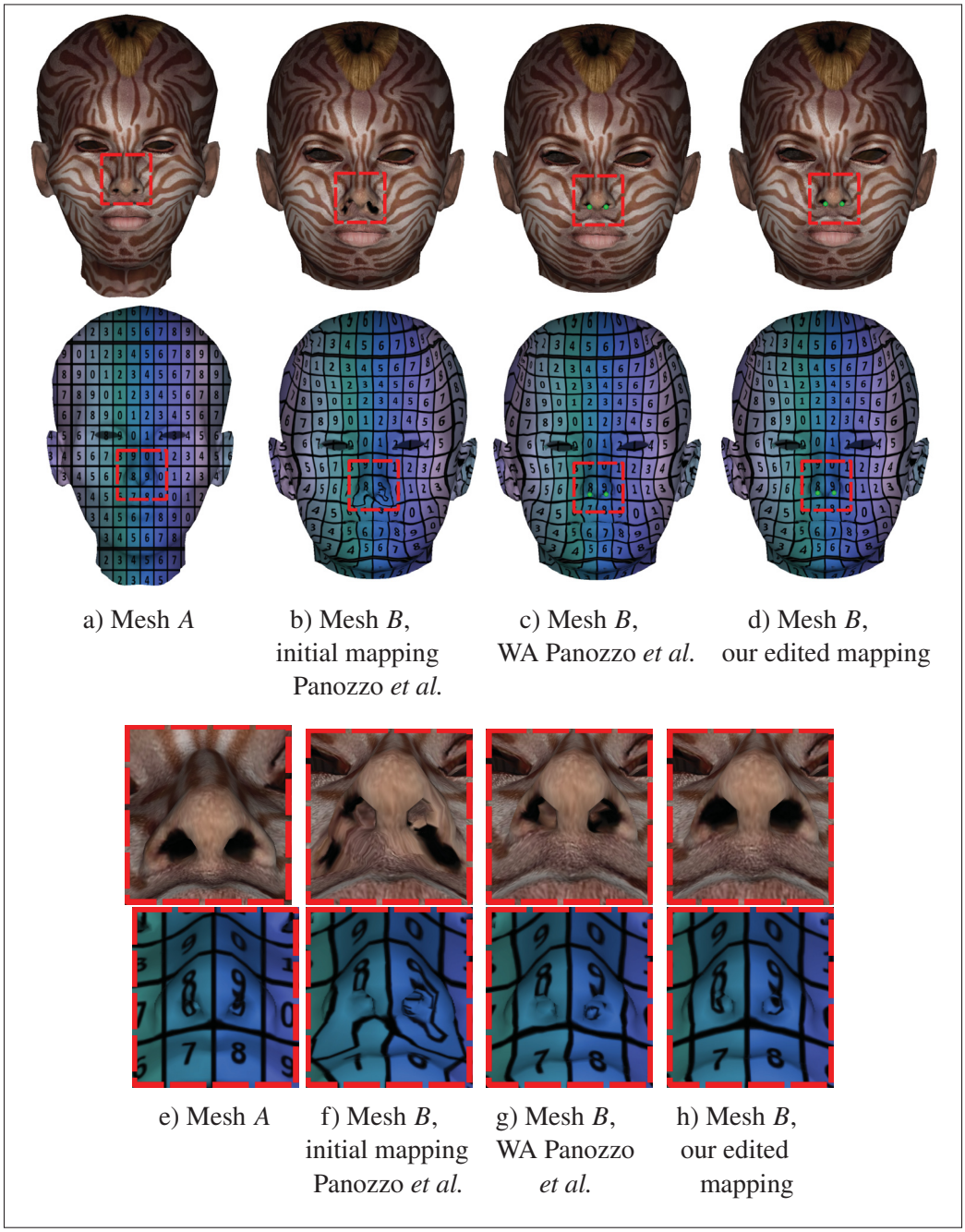


Figure 3.22 Qualitative comparison with the WA method (Panozzo *et al.*, 2013). Our approach significantly improves the mapping in the nostril area

3.2.4 Applications Relying on Surface Mapping

Several applications rely on a mapping between surfaces: texture transfer (Zell & Botsch, 2013), animation setup transfer (Avril *et al.*, 2016), and deformation transfer (Sumner & Popović, 2004). We use the methods of Sumner & Popović (2004) and Avril *et al.* (2016) to illustrate how the proposed approach can significantly improve the results of techniques relying on a mapping. Fig. 3.23 shows a facial transfer result before and after editing. Results demonstrate several issues and unpleasant deformations for fine features, such as strange deformations on the corners of the mouth. With the corrected mapping, these problems disappear. Fig. 3.24 shows a skeleton transfer (Avril *et al.*, 2016) result before and after the mapping is edited. Results demonstrate that the joint, that was erroneously positioned outside of the thumb, moves to the right place when improving the surface mapping locally in the thumb region instead of improving the mapping globally over the mesh.

Our approach works even for surfaces with boundaries inside the segments. Such boundaries are commonly encountered with the ears, eyes, nostrils, and mouths of characters. While we constrain the segment boundaries to prevent them from moving, an initial mesh boundary lying inside a segment will be free to move. Leaving these inner boundaries completely free has a negative impact on the deformation. Fig. 3.25 shows the deformation of the mouth without c and with d inner boundary fillings. Note here the improvement of the mouth deformation when filling the inner boundary.

3.3 Discussion

Our approach carves a new path in between the more classical shape-preserving methods, which often lose local injectivity, and the more current methods, which formulate the injectivity constraint as part of the optimization. These latter methods typically do not have a bound on the shape-preserving error. In our approach, we are minimizing only the shape-preserving term (i.e., LSCM energy) and iteratively improving the user constraints while maintaining a locally injective map in each iteration (a formal proof is found in Appendix I). We achieve this

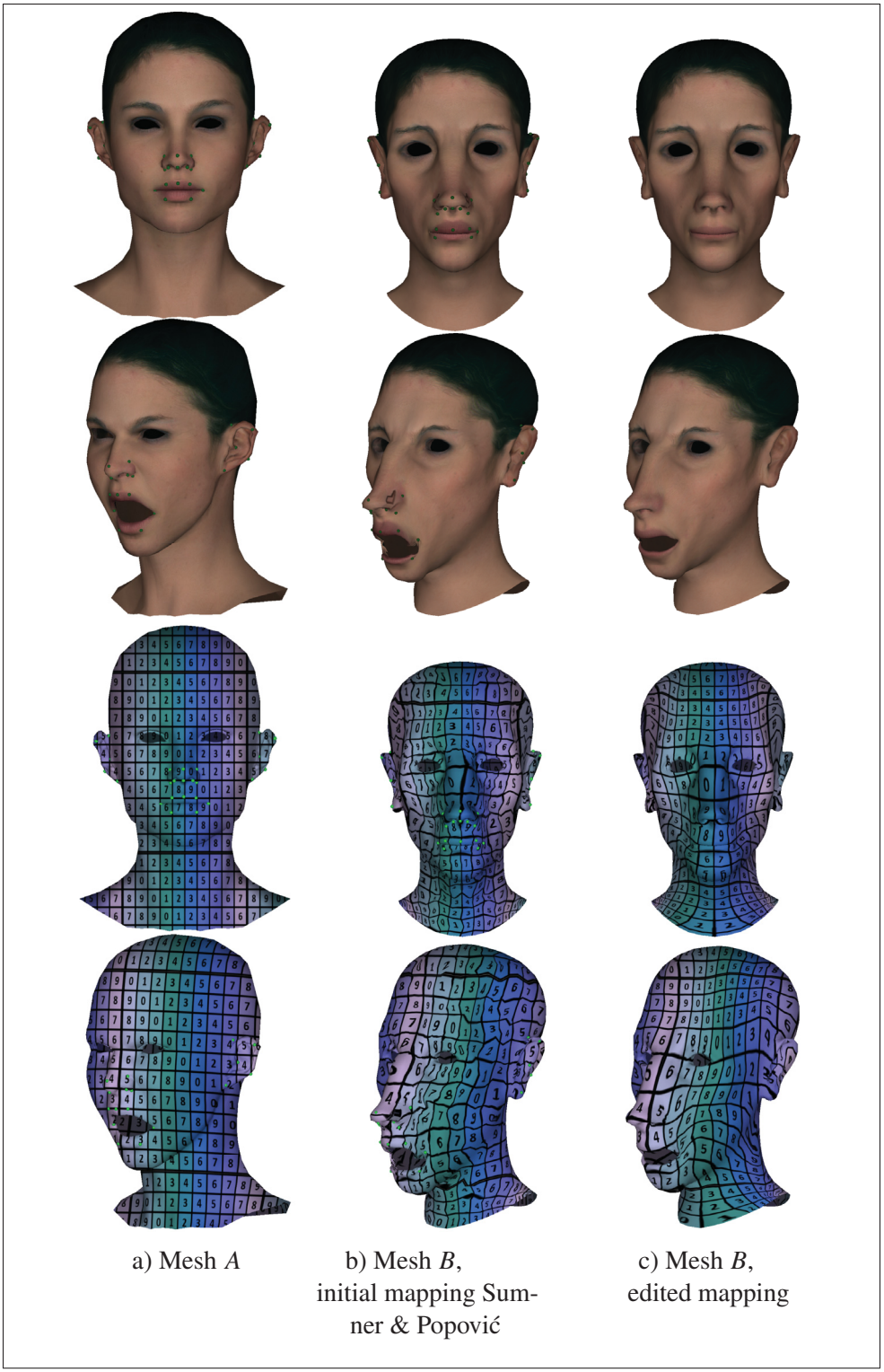
by carefully controlling the λ parameter in Eq. 3.1. At one extreme, if λ is very large (i.e., infinity), the model is equivalent to the LSCM formulation. If λ is very small, it takes many iterations for the user constraints to be satisfied, or in some cases, the user constraints may ultimately not be satisfied. Our iterative scheme relies on two important observations. If λ is 0, the solution is the same as the initial configuration. Therefore, if we start in a locally injective configuration, the final result will be a locally injective configuration. If the initial configuration is locally injective, there always exists a λ (however small) that will result in a locally injective configuration, where the user constraints are closer to the target. This scheme will converge to a locally injective configuration. Consequently, we iteratively repeat the optimization to fight against flipped faces, but convergence cannot be guaranteed. It is always possible to design a landmark configuration in which the constraints cannot be met without flipped faces. This is true for the other deformation methods as well. Appendix II demonstrates different failure cases using different deformation methods. In our experiments, except for the examples in Appendix II, the constraints are satisfied (up to numerical precision), even for extreme deformations.

In our results, we improved mappings which were initially computed from a variety of methods (Aigerman & Lipman, 2016; Nogneng & Ovsjanikov, 2017; Ramachandran *et al.*, 2018; Panozzo *et al.*, 2013; Sumner & Popović, 2004; Zell & Botsch, 2013). Even if these initial mappings minimize different deformation energies, the fact that we rely on the LSCM conformal energy to edit them did not prevent our approach to improve the mappings. One must keep in mind that the goal of the editing is not to strictly minimize a deformation energy, but to align important semantic features of the objects and maintain injectivity.

We analyzed our results to verify the degree to which the deformation deteriorates the shape of the triangles. We checked 13 of the results found in this chapter, and we considered that a detrimental deformation is one in which the angle becomes more than 20 times narrower after deformation. Eleven cases had no such triangles, while the two other cases had two and three, respectively. The worst triangle in our 13 test cases was 24 times narrower than before deformation. Any deformation method is prone to result in thin triangles, so we compared our approach to LIM, SLIM, and KP-Newton for six examples. When looking at the worst

triangle found in the deformed meshes, ILSCM performed best for four of the test cases, while KP-Newton performed best for two of the test cases. SLIM and LIM were systematically in third and fourth place behind ILSCM and KP-Newton. Furthermore, our results were better than LIM, SLIM, and KP-Newton in terms of shape preservation and final triangulation, as can be seen in Fig. 3.12 and in the video. As mentioned earlier, to ensure a fair comparison, we adapted all of the other methods so that they minimize the LSCM energy. For our approach, we minimized the LSCM energy through a least-squares solve. We ran our experiments on a 3.40 GHz Intel Core-i7-4770 CPU with 12 GB of memory, with a MATLAB implementation of our approach. Table 3.1 shows computation times for the segmentation and the deformation (including mapping extraction) phases. Since our deformation phase is iterative, the time to edit a mapping depends on the size of the mismatching regions and the number of iterations.

We have presented a novel approach for improving surface mappings locally. Our approach is based on a low-distortion region-growing segmentation followed by an independent planar parameterization of each segment. The mapping is then optimized based on an alignment of the user-prescribed landmarks in the parameterization space of each segment. Our joint planar parameterization deformation for the segments is robust, and results in low distortion. Our new iterative LSCM approach can be reused in several contexts where a deformation with low distortion is required. From a practical perspective, our approach has several advantages. It can be used to improve the mapping resulting from any surface mapping method. It also provides a great deal of control, allowing the user to restrict editing to a specific region and to add as few or as many landmarks as necessary to achieve a desired result.



a) Mesh *A*

b) Mesh *B*, initial mapping Sumner & Popović

c) Mesh *B*, edited mapping

Figure 3.23 Some mapping issues are not visible on a static mesh with a neutral expression. We show that some subtle mapping issues become obvious and lead to severe problems by opening the mouth

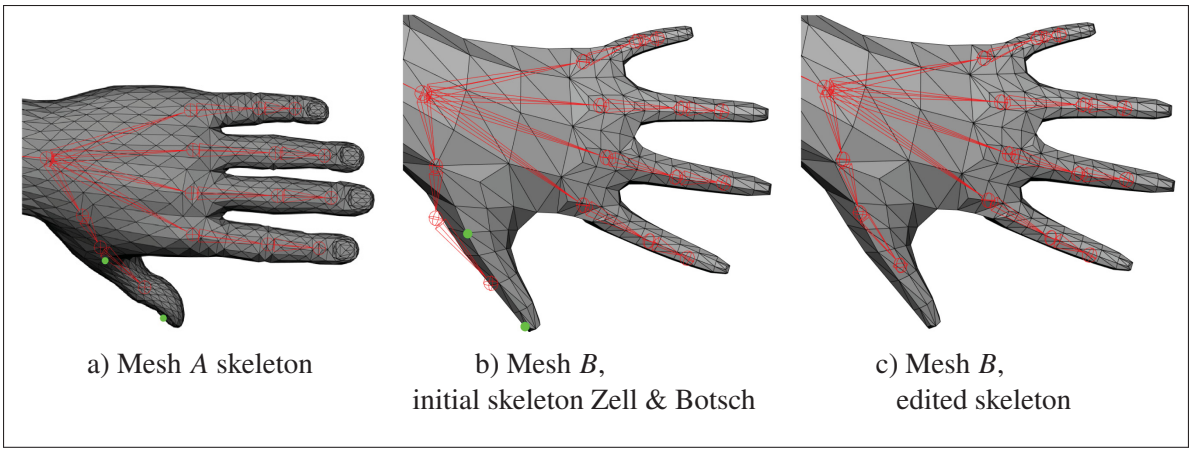


Figure 3.24 When using the mapping to retarget attributes, in this case the skeleton, an incorrect mapping will lead to problems, here putting the thumb joint outside of the mesh. By locally editing the mapping, it is easy to fix such issues

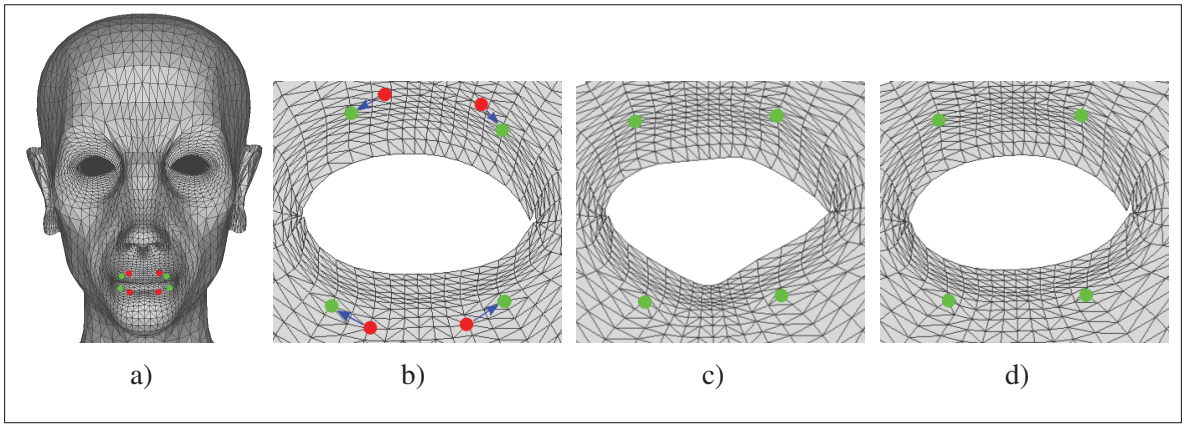


Figure 3.25 a Mesh in 3D. b Mesh in 2D. c Deformation without filling. d Deformation with filling

CHAPTER 4

PART-BASED 3D FACE MORPHABLE MODEL

In this chapter, we propose an approach to construct realistic 3D facial morphable models (3DMM) that allows an intuitive facial attribute editing workflow. This research led to a publication "Part-Based 3D Face Morphable Model with Anthropometric Local Control" by Ghafourzadeh *et al.* (2020a) which won the GI 2020 best student paper award. The complexity of facial features makes the modeling of the human face a time-consuming and challenging task. 3D morphable models (3DMMs) are powerful statistical models widely used in animation (blend shapes), face capture, and face editing. These models are generated by performing dimension reduction, computing a Principal Component Analysis (PCA), on a data set of 3D scans sharing the same mesh topology. Each 3D face is transformed to a vector of variable in PCA space by taking projections on eigenvectors. Conversely, a 3DMM can be generated using corresponding vector in PCA space with a reconstruction error due to the dimension reduction.

Our local PCA-based approach uses a novel method to select the best eigenvectors from the local 3DMM to ensure that the combined 3DMM is expressive, while allowing accurate reconstruction. The editing controls we provide to the user are intuitive as they are extracted from anthropometric measurements found in the literature. Out of a large set of possible anthropometric measurements, we filter those that have meaningful generative power given the face data set. We bind the measurements to the part-based 3DMM through mapping matrices derived from our data set of facial scans. The authoring of realistic 3D faces with intuitive controls is used in a broad range of computer graphics applications, such as video games, person identification, facial plastic surgery, and virtual reality. This process is particularly time-consuming, given the intricate details found in the eyes, nose, mouth, and ears. Consequently, it would be convenient to use high-level controls, such as anthropometric measurements, to edit human-like character heads.

One of the most well-known previous work about 3DMMs is that by Blanz & Vetter (1999). Their pioneer work proposes a model using PCA from face scans. Although they propose a multi-segment model and decompose a face into four parts to augment expressiveness, the PCA

decomposition is computed globally on the whole face. Other global PCA methods have been proposed (Allen *et al.*, 2003; Cao *et al.*, 2014; Booth *et al.*, 2016; Li *et al.*, 2017; Booth *et al.*, 2018; Lüthi *et al.*, 2018).

These methods are popular due to the simplicity and efficiency of the approach, but suffer from a fundamental limitation: they impose global control on the new generated meshes, making it impossible to edit a localized region of the face. For example, when we adjust the eye, the nose may also undergo undesirable changes. Another downside is a lack of intuitive user control for face editing. While the eigenvectors are good at extracting the dominant modes of variation of the data, they provide weak intuitive interpretation.

To address the former problem, local models have been proposed. They segment the face into independent sub-regions and select the most dominant eigenvectors per part. Clustered PCA (Tena *et al.*, 2011) generates localized 3DMMs composing of a collection of PCA segment models that are independently generated. It starts by segmenting the face with spectral clustering, followed by manual adjustment. For this purpose, it calculates the inter-vertex distance on the face using the isomap algorithm to generate a distance matrix and then normalizes it into the range $\left[0, 1\right]$. These two matrices are added in a weighted manner and spectral clustering (Ng, Jordan & Weiss, 2002) is performed to obtain the region segmentation. This process can lead to different numbers of clusters depending on the used data sets. Afterwards, it applies PCA to each segment independently to generate its own subspace. Therefore, clustered PCA groups the vertices that are extremely correlated and forms compact segments which can be better compressed using PCA. This method computes localized 3DMMs better than the traditional holistic approach but it focuses mostly on facial expression. Chi *et al.* (2017) adaptively segment the face model into soft regions based on user-interaction and coherency coefficients. Afterwards, they estimate the blending weights which satisfy the user constraints, as well as the spatio-temporal properties of the face set. Here too, the required user intervention renders the segmentation somewhat similar to our user-provided segments. In order to capture localized facial details, SPLOCS (Neumann *et al.*, 2013) creates localized basis vectors. It proposes sparse matrix decompositions to extract sparse and spatially localized deformation from

animation sequences. SPLOCS uses vertex displacements in the Euclidean coordinates to capture local deformations and select the basis in a greedy fashion. Its algorithm decomposes vertex displacements into a linear combination of weights and sparse deformation components. It relies on a regularizer that is designed for animation data and then automatically finds components that are localized and sparse on the mesh. SPLOCS accepts input provided by the user to guide the process and provides a new efficient optimization and initialization layout to generate the decomposition. This method works remarkably well for face animation but it is not suitable for our work that focuses on a precise semantic face modeling. We noticed that when considering variation in identity instead of variation in expression, the greedy selection leads to bases which are far less local than those obtained from our method.

These papers address facial animation instead of face modeling and therefore assume large, yet localized deformations caused by facial expressions, which are different from our context where each face is globally significantly different from the others. Even though face animation concerns are important, our work focuses on the editing of facial meshes.

We propose an approach to construct realistic 3DMMs. As mentioned above a downside of global PCA-based methods is that they exhibit global support: adjusting parameters to change one part has unwanted effects on other unrelated parts. To address this problem, we segment the face into independent sub-regions and provide a process to select the best set of eigenvectors, given a target number of eigenvectors. Methods that segment the face in sub-regions target facial animation instead of modeling. We will demonstrate that our approach is better-suited to the task of face editing than these methods. Another problem with most of the previous related work is that they do not allow facial model editing through the adjustment of objective measurements. In contrast, our method relies on anthropometric measurements used as controls for editing. Furthermore, we propose a process to select the right set of anthropometric measurements for each facial part and bind the essential measurements to the 3DMM eigenvectors. The measurements are mapped to eigenvector weights, allowing us to compute the individual parts matching the values selected by the user. Finally, the reconstructed parts are seamlessly blended together to generate the desired 3D face. We compared our approach to previous work relying

on facial animation and saw that other automatic localized basis constructions work well for animation purposes (considering a data set composed of animations for a single person), but perform worse than our approach for modeling purposes (considering a data set made of neutral faces from different persons). Our approach has three key phases:

1. 3D Morphable Face Model (Sec. 4.1): Face decomposition and selection of the most dominant eigenvectors are the main steps to construct new facial parts.
2. Face Construction through Parts Blending (Sec. 4.2): A face is reconstructed by smooth blending of the generated facial parts.
3. Synthesizing Faces from Anthropometric Measurements (Sec. 4.3): The most relevant anthropometric measurements are selected. Afterward, the mapping between these measurements and the PCA eigenvectors are defined to construct new facial part based on specific measurements.

4.1 3D Morphable Face Model

We employ PCA on a data set of faces to construct our 3DMMs. All faces are assumed to share a common mesh topology, with vertices in semantic correspondence. We propose to segment the face into different parts in order to focus the decomposition on a part-by-part basis instead of computing the PCA decomposition on the whole face. We compute the decomposition separately for the male and female subsets. As shown in Fig. 4.1, we decompose the face into five parts: eyes, nose, mouth, ears, and what we refer to as the facial mask (which groups the remaining areas such as cheeks, jaws, forehead, and chin). We further discuss this design choice in Sec. 4.5.2. In an offline stage, we extract PCA eigenvectors and select the best ones. We also select the best subset of anthropometric measurements. The relationship between the eigenvectors and measurements is encoded in a mapping matrix. All of these are used in the online stage, where the mapper receives user-prescribed anthropometric measurement values, and applies the mapping matrices to reconstruct the parts. The last step provides the edited face through a smooth blending of the parts.

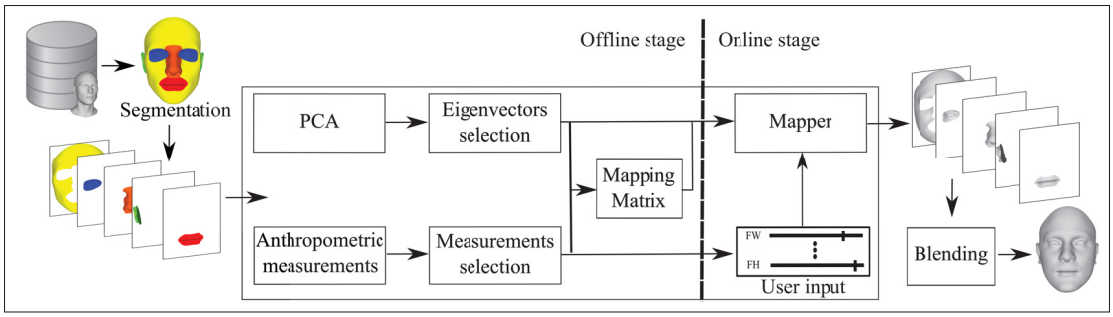


Figure 4.1 Our 3D facial morphable model workflow

The face decomposition allows us to have eigenvectors for each part. The geometry of the facial parts is represented with a shape-vector $S_d = \left[V_1 \dots V_{n_v}\right] \in R^{3n_v}$, where n_v is the number of vertices of dth facial part, $d \in \{1, \dots, 5\}$, and $V_i = \left[x_i y_i z_i\right] \in R^3$ defines the x, y, and z coordinates of the ith vertex. We perform PCA on a xyz space because smooth basis like Laplacian eigenvectors of the mesh give a smoothed version of the geometry. After applying PCA, each facial part d is reconstructed as:

$$S'_d = \overline{S_d} + \sum_{j=1}^{n_e} P_j b_j, \tag{4.1}$$

where $\overline{S_d}$ is the mean shape of dth facial part, n_e is its number of eigenvectors, P_j is an eigenvector of size $3n_v$, b is a $n_e \times 1$ vector containing the weights of the corresponding eigenvectors, and S'_d is the reconstruction, which will be an approximation when not using all eigenvectors.

Our approach selects the smallest set of eigenvectors that still reconstructs the shape accurately. We accomplish this by incrementally adding the eigenvectors, in the order of their significance, to the reconstruction until a certain accuracy is met. Even though we rely on the eigenvalues to sort the eigenvectors for each part (largest to smallest eigenvalue), we provide the user with a measurable error (in mm), which is more precise than relying solely on eigenvalues across different parts. We determine the best set of eigenvectors to achieve a balance between the quality of the per-part reconstruction and the whole face reconstruction. To evaluate the accuracy of our selection, we construct the facial parts (Eq. 4.1) and blend them together (Sec. 4.2) to

generate the whole face. Afterwards, we assess the accuracy of the reconstruction by calculating the average of the geometric error D_{GE} between the ground truth and the blended face. We first do a rigid alignment step (rotation and translation) between the facial parts of the ground truth and the blended result. We then record the average per-vertex Euclidean distance over all vertices and per part:

$$D_{GE_{all}}(S') = \frac{1}{n_{all}} \sum_{d=1}^{5} \sum_{\substack{V'_j \in S'_d \\ V_j \in S_d}} \left\| V_j - \left(R_d V'_j + T_d \right) \right\| \tag{4.2}$$

$$D_{GE_{part}}(S') = \frac{1}{5} \sum_{d=1}^{5} \frac{1}{n_d} \sum_{\substack{V'_j \in S'_d \\ V_j \in S_d}} \left\| V_j - \left(R_d V'_j + T_d \right) \right\| \tag{4.3}$$

$$D_{GE}(S') = \frac{D_{GE_{all}}(S') + D_{GE_{part}}(S')}{2}, \tag{4.4}$$

where n_{all} is the number of vertices of the face mesh, n_d is the number of vertices for part d, V_j is on the ground truth and V'_j is the corresponding point on the blended face. We compute averages over all vertices and per part to ensure that parts with more vertices do not end up using most of the eigenvector budget at the expense of parts with fewer vertices. We do so for the entire data set and for a set of 19 validation faces that were not part of the training data set. We compute the median data set error as well as the median validation error, and we average the two in a global error. The process of reconstructing the parts of validation faces is done by projecting each part onto the corresponding eigenvector basis (followed by the blending process).

At each step of our incremental eigenvector selection, we decide which of the five parts will get a new eigenvector added to its set. We compare the geometric errors resulting from each of the five candidate eigenvectors, and we select a candidate eigenvector which has a great impact on decreasing the error. When eigenvectors from multiple parts result in similar decreases in error, instead of systematically picking the eigenvector based on lowest error, we select by sampling from a discrete probability density function (PDF) created from the respective decreases in error of the five candidate eigenvectors. This PDF selection process creates a more even distribution

of eigenvectors across the parts and maintains a low error. As we iterate, the reconstruction error decreases. For the female and male data set faces, the average reconstruction errors are 2.00 and 2.13 mm when considering zero eigenvectors. The errors decrease to 0.75 and 0.74 mm after 80 iterations, and when considering all eigenvectors the errors are 0 mm. We chose an error threshold of 1 mm which balances out the cost associated with considering too many eigenvectors and the accuracy of the reconstruction. Table 4.1 shows the resulting eigenvector distribution after achieving our 1 mm reconstruction accuracy based on the training data set. We experimented with reconstructing the female and male validation faces based on using our subset of eigenvectors. The median reconstruction errors are 1.33 and 1.48 mm, respectively. Since these faces were not part of the training data set, it led to larger reconstruction errors which are still close to 1 mm.

Table 4.1 Number of eigenvectors selected for each part

Facial part	**# Eigenvectors for female**	**# Eigenvectors for male**
Facial mask	7	9
Eye	10	5
Nose	6	6
Mouth	9	10
Ear	14	16

4.2 Face Construction through Parts Blending

This step focuses on the problem of constructing a realistic new face by blending the five segmented parts together. As opposed to methods such as those of Tena *et al.* (2011) and Cao *et al.* (2018), which handle the transition between the parts by relying on the adjustment of a single strip of vertices, we spread the transition across three strips of vertices. In contrast to other methods that adjust the transition by vertex averaging (Cao *et al.*, 2018) or least-squares fitting (Tena *et al.*, 2011), we use Laplacian blending (Sorkine, Cohen-Or, Lipman, Alexa, Rössl & Seidel, 2004) of the parts and the transition, resulting in a smooth, yet faithful global surface. The vertex positions are solved by an energy minimization which reduces the surface curvature discontinuities at the junction between the parts while maintaining the desired surface curvature. To this end, we define a transition zone made of quadrilateral strips around the

parts. In our experiments, a band of two quadrilaterals (three rings of green vertices in Fig. 4.2) provides good results.

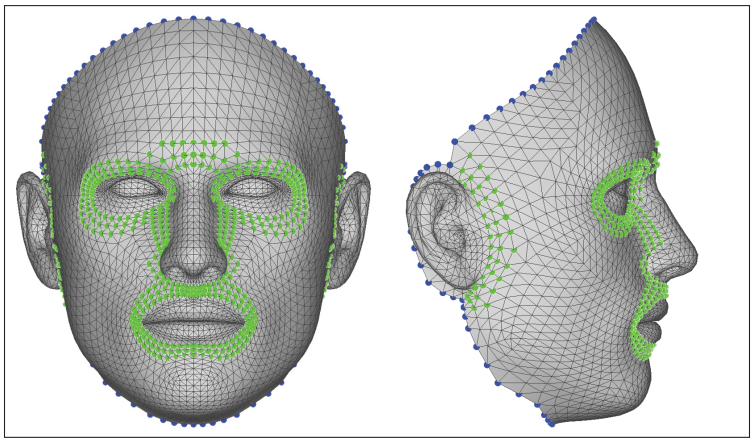


Figure 4.2 Regions highlighted in green and blue contain the transition and fixed zones, respectively

We interpolate the Laplacian $\mathcal{L}$ (the cotangent weights) of the five facial parts weighted by β_d, which has values of $\beta_d = 1$ inside the part, $\beta_d \in \{0.75, 0.5, 0.25\}$ going outward of the part in the transition zone, and $\beta_d = 0$ elsewhere. We normalize these weights such that they sum to one for each vertex. These soft constraints allow some leeway in the transition zone. The boundary conditions of our system are set to the ring of blue vertices in Fig. 4.2, and we solve for the remaining vertices. To this end, we minimize the following energy function:

$$E(V') = \sum_{i \in \text{inner}} \left\| T_i \mathcal{L}(V'_i) - \frac{\sum_{d=1}^{5} \beta_{i,d} R_d \mathcal{L}(V_{i,d})}{\sum_{b=1}^{5} \beta_{i,b}} \right\|^2, \tag{4.5}$$

where "inner" is the set of vertices of the five parts, excluding the vertices of the boundary conditions.

T_i is a transformation including rotation and scale for vertex V'_i based on the new configuration of vertices V_i and R_d is a rotation matrix for part d.

We solve T_i in a similar fashion to Laplacian Surface Editing (Sorkine *et al.*, 2004). We also use the idea of ARAP (Sorkine & Alexa, 2007) by alternating solving for the vertex position and

rotation matrices to find the best approximating rotations that align segments together. Fig. 4.3 shows a truly random experiments by preparing the meshes beginning with the average parts and changing the weight of one eigenvector per part. Each eigenvector is selected randomly (from the first ten eigenvectors if there are more than ten eigenvectors for the part). The new value for the weight is also randomly selected within the range of −2 and +2 times the standard deviation for this eigenvector. Our experiment shows that the rotation R_d quickly converges as the Frobenius norm of consecutive rotations is large only for the first few iterations. Given our experiments, we decided to stop iterating when the Frobenius norm of the rotation between two consecutive iterations fell below 0.01 or after 6 iterations. Fig. 4.4 shows an example of a blended face. In this case, the Frobenius norm was below 0.01 after five iterations. Fig. 4.5, shows the evolution of the geometric error D_{GE} between the ground truth parts and their blended counterparts for the example of Fig. 4.4. As can be seen, the error quickly reaches a plateau as the rotation stabilizes.

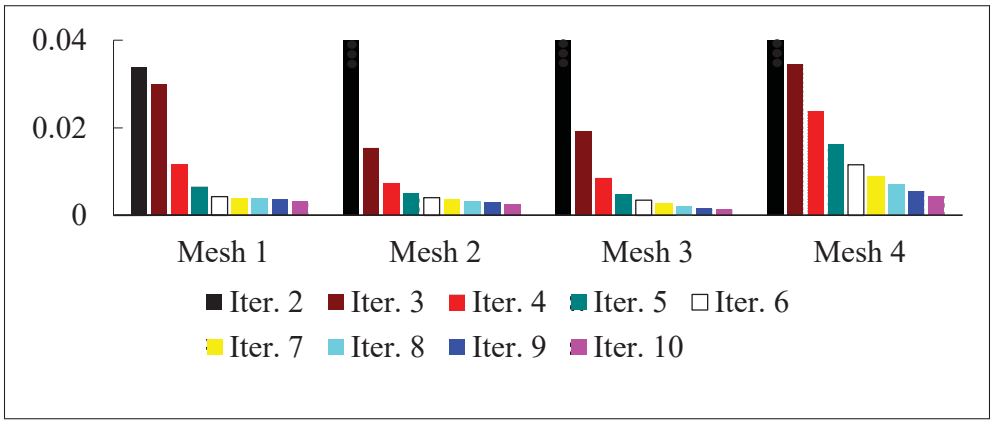


Figure 4.3 Graph showing the evolution of the Frobenius norm of the rotation R_d between two consecutive iterations (averaged across the five segments)

4.3 Synthesizing Faces from Anthropometric Measurements

PCA eigenvectors characterize the data variation space, but do not provide a clear intuitive interpretation. We focus mainly on constructing linear regression models from data using a set of intuitive facial anthropometric measurements. Facial anthropometric measurements provide

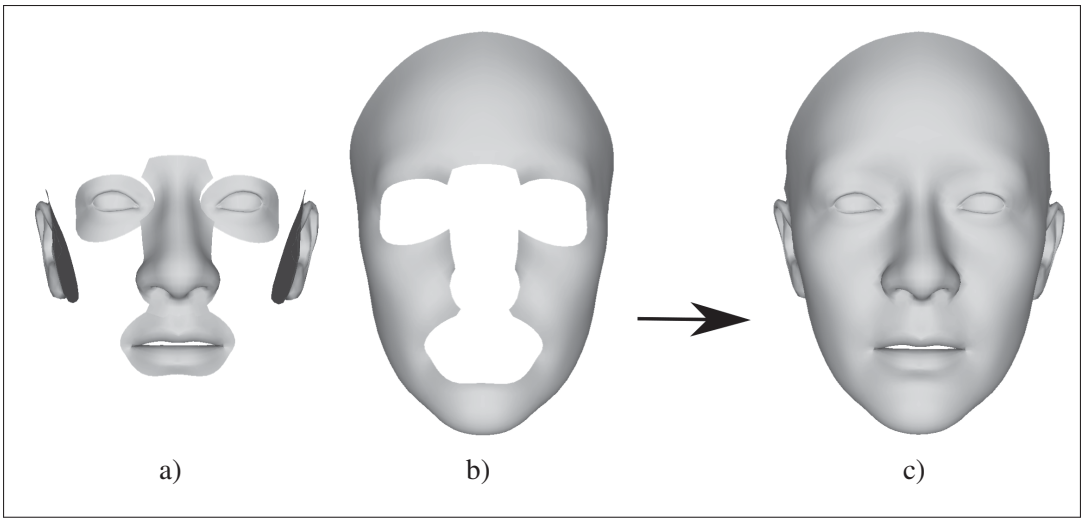
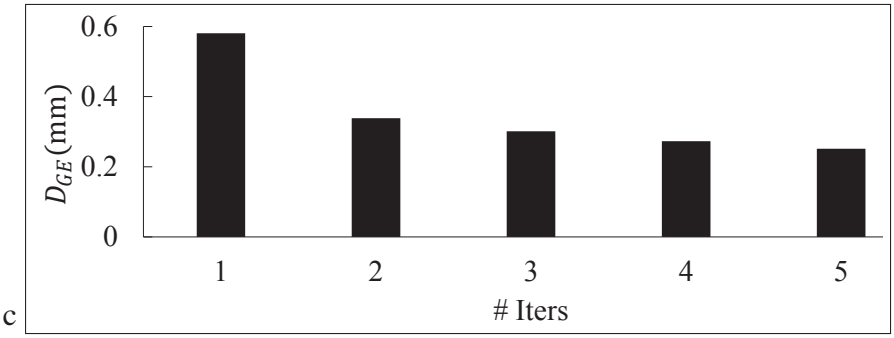


Figure 4.4 a and b are the generated parts. As in Fig. 4.3, we modified each average part by changing the weight of one eigenvector selected randomly. c shows the result of blending the facial parts.



c

Figure 4.5 Graph comparing the average geometric error (in mm) between the ground truth parts and their blended counterparts, for different numbers of iterations

a quantitative description by means of measurements taken between specific surface landmarks defined with respect to anatomical features. We use the 33 parameters listed in Table 4.2. Each measurement corresponds to either an Euclidean distance or a ratio of Euclidean distances between surface positions, as specified in each paper cited in Table 4.2. The designated facial landmarks used for these anthropometric measurements are shown in Fig. 4.6. In this section, we

propose a measurement selection technique which assesses the accuracy of each measurement, resulting in the most relevant ones for each facial part.

Table 4.2 Anatomical terms and corresponding abbreviations of our selected and discarded measurements. Ref. 1: Taken from Etöz & Ercan (2012), Ref. 2: Taken from Schendel (1995), Ref. 3: Taken from Farkas *et al.* (2005), Ref 4: Taken from Lacko *et al.* (2015), Ref. 5: Taken from Lee *et al.* (2006), Ref. 6: Taken from Zhuang *et al.* (2010)

Selected measurements		
Anatomical term	Abbrev.	Ref.
Nasal Width÷Root Width	NWRW	Ref. 1
Nasal Width÷Length of Bridge	NWLB	Ref. 1
Nasal Width÷Width of Nostril	NWWN	Ref. 1
Nasal Root Width÷Tip Protrusion	NRTP	Ref. 1
Length of Nasal Bridge÷Tip Protrusion	NBTP	Ref. 1
Nasal Width÷Tip Protrusion	NWTP	Ref. 1
Nasal Root Width÷Length of Bridge	NRLB	Ref. 1
Nasal Root Width÷Width of Nostril	NRWN	Ref. 1
Length of Nasal Bridge÷Width of Nostril	NBWN	Ref. 1
Width of Nose÷Tip Protrusion	WNTP	Ref. 1
Philtrum Width	PW	Ref. 2
Face Height	FH	Ref. 3
Orbits Intercanthal Width	OIW	Ref. 3
Orbits Fissure Length	OFL	Ref. 3
Orbits Biocular Width	OBW	Ref. 3
Nose Height	NH	Ref. 3
Face Width	FW	Ref. 4
Bitragion Width	BW	Ref. 4
Ear Height	EH	Ref. 4

Selected measurements		
Anatomical term	Abbrev.	Ref.
Bigonial Breadth	B	Ref. 5
Bizygomatic Breadth	BB	Ref. 5
Facial Index	F	Ref. 6
Nasal Index	N	Ref. 6
Mouth-Face Width Index	MFW	Ref. 6
Biocular Width-Total Face Height Index	BWFH	Ref. 6
Lip Length	LL	Ref. 7
Maximum Frontal Breadth	Max FB	Ref. 7
Interpupillary Distance	ID	Ref. 7
Nose Protrusion	NP	Ref. 7
Nose Length	NL	Ref. 7
Nose Breadth	NB	Ref. 7
Discarded measurements		
Anatomical term	Abbrev.	Ref.
Eye Fissure Index	EF	Ref. 6
Minimum Frontal Breadth	Min FB	Ref. 7

4.3.1 Mapping Method

We evaluate the measurements on the facial parts of the data set, yielding $f_{d,i} = \left[f_{i_1} \dots f_{i_{n_m}} \right]$ for the dth facial part of scan S_i considering n_m measures. The measures for all of the scans are combined into an $n_m \times n_s$ matrix, $F_d = \left[f_{d,1}^T \dots f_{d,n_s}^T \right]$, where n_s is the number of scans. We learn how to adjust the weights of the PCA eigenvectors to reconstruct faces having specific characteristics corresponding to the measures. We adopt the general method of Allen *et al.* (2003). However, while that method learns a global mapping that adjusts the whole body, we will learn per-part local mappings. Furthermore, in Sec. 4.3.2 we will derive a process to select

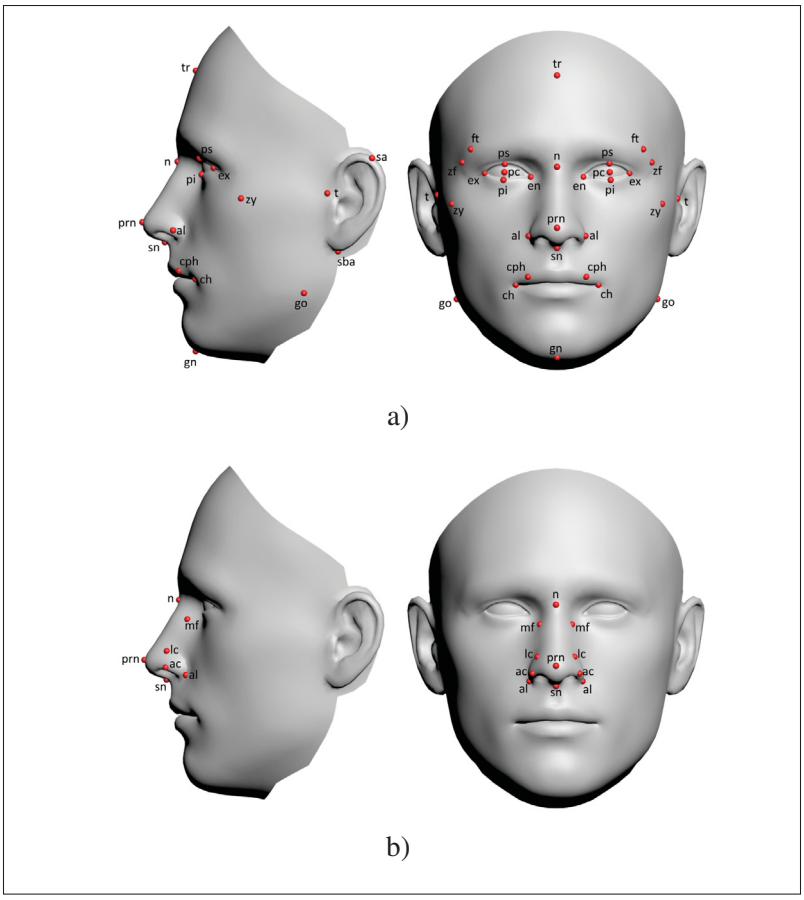


Figure 4.6 a and b show facial landmark locations used for anatomical terms on a generic model

the best measures out of the set of all measures $\left[f_{i_1} \ldots f_{i_{n_m}}\right]$, and will proceed independently for each of the five parts.

We relate measures by learning a linear mapping to the PCA weights. With the n_m measures for the dth facial part, the mapping will be represented as a $(n_e) \times (n_m + 1)$ matrix, M_d:

$$M_d\left[f_{i_1} \ldots f_{i_{n_m}} 1\right]^T = b, \tag{4.6}$$

where b is the corresponding eigenvector weight vector. Collecting the measurements for the whole data set, the mapping matrix is solved as:

$$M_d = B_d F_d^+, \tag{4.7}$$

where B_d is a $(n_e) \times (n_s + 1)$ matrix containing the corresponding eigenvector weights of the related facial part and F_d^+ is the pseudoinverse of F_d. As in Eq. 4.6, a row of 1s is appended to the measurement matrix F_d for y-intercepts in the regression.

To construct a new facial part based on specific measurements, we use b in Eq. 4.1 which is achieved by projecting 3D face (removing average) to the eigenvector. We adjust it to represents a range of $(-2, 2)$ standard deviations:

$$S'_d = \overline{S_d} + Pb, \tag{4.8}$$

where $\overline{S_d}$ is the mean shape of the dth facial part, and P is the matrix containing the eigenvectors. Moreover, we can define delta-feature vectors of the form:

$$\Delta f_d = \left[\Delta f_1 \ldots \Delta f_{n_m} 0\right]^T, \tag{4.9}$$

where each Δf contains the user-prescribed differences in measurement values. Afterwards, by adding $\Delta b = M_d \Delta f_d$ to the related eigenvector weights, it is possible to adjust the measure such as to make a face slimmer or fatter. On the left of Fig. 4.7, we show the adjustment of the "Face Width". On the right, we adjust the "Nose Breadth". At the top, we adjust the "Orbits Biocular Width", thus increasing the distance between his eyes. At the bottom, the mouth becomes wider by adjusting the "Lip Length".

4.3.2 Measurement Selection

We propose a novel technique for automatically detecting the most effective and relevant anthropometric measurements. Some might be redundant with respect to others, some might not

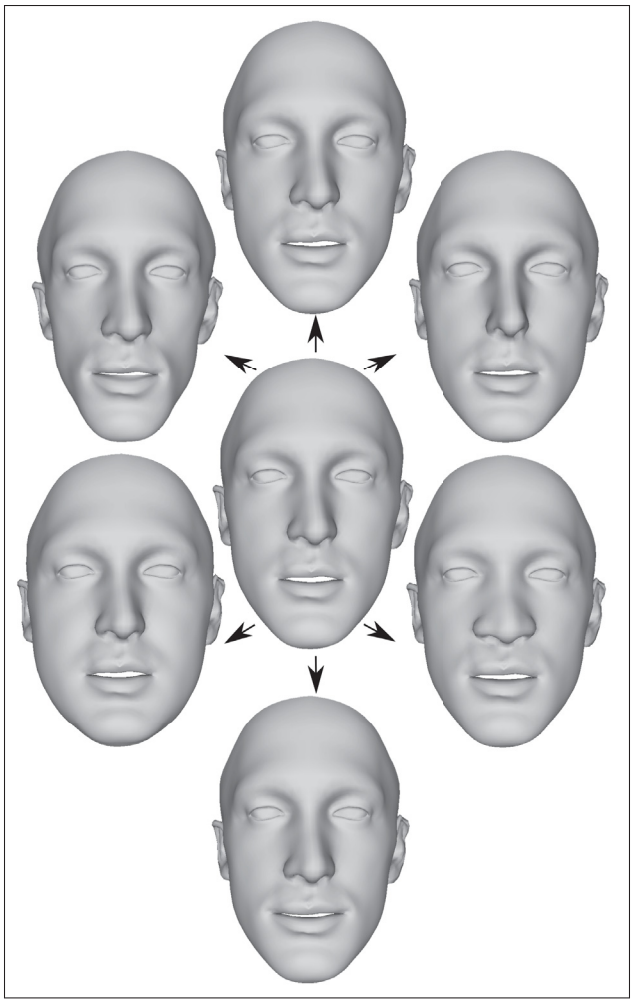


Figure 4.7 Our approach allows to take a specific face (middle) and use the measurements to edit it

make sense for a specific part (e.g., the “Ear Height” might not be relevant for the mouth), and some might even lead to mapping matrices that generate worst results. During our investigations, we discovered that considering more anthropometric measurements does not necessarily lead to a lower reconstruction average error. Fig. 4.8 illustrates that a higher error occurs considering all the measurements in comparison with our selected combination. In order to aggregate the error for the given part, we reconstruct the face, relying only on the anthropometric measurements of the selected part, and then calculate the average error, as in Sec. 4.1. Fig. 4.9 shows two odd-looking examples from using all the measurements of the nose a and facial mask b.

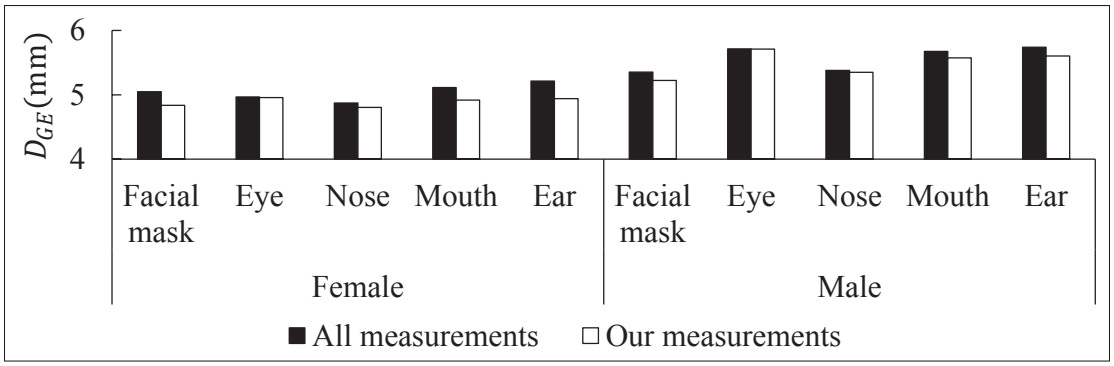


Figure 4.8 Using all measurements leads to higher reconstruction errors (mm) as compared to our set of selected measurements on the data set and validation faces

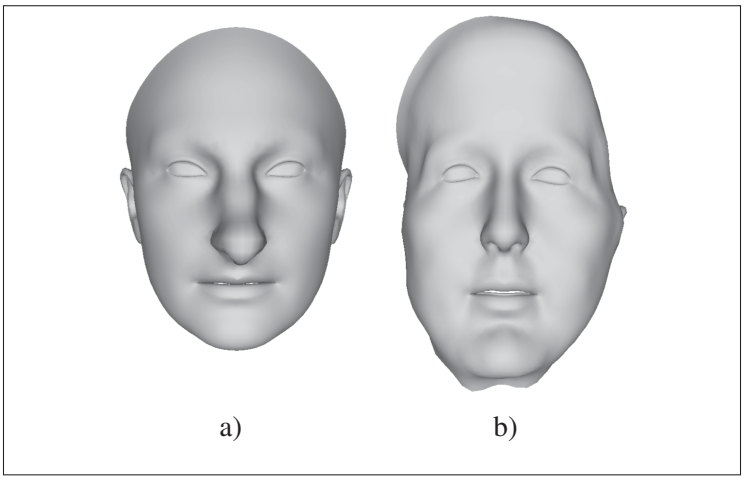


Figure 4.9 Using all the semantic measurements of the nose a and facial mask b often leads to odd-looking parts when editing through the adjustment of measurement values

We thus evaluate the set of relevant measurements, separately for each part. We begin with an empty set of *selected measurements*, and we iteratively test which measurement we should add to the set by evaluating the quality of the reconstructed faces, considering the currently selected measurements together with the *candidate measurement*. We reconstruct a face using the mapping matrix (Eq. 4.6, 4.8) based only on its measurement values. The reconstructed face is considered as a *prediction*, and thus we evaluate the prediction quality in a fashion very

similar to that used for eigenvector selection, by reconstructing all of the faces found in the data set of facial scans, as well as the 19 validation faces.

Each candidate measurement is used together with the current set of selected measurements, and we compute the candidate mapping matrix from this set of measurements. We use the mapping matrix with the data set and validation faces, and reconstruct all of the instances of the part under consideration (e.g., all of the mouths). We then evaluate a geometric error, D_{GE}, with the per-vertex distance between each predicted instance and its corresponding ground truth instance. The distance is calculated after a rigid alignment of the predicted instance to the ground truth instance is performed. We can thus ensure that we are evaluating the fidelity of the shape, and not its pose. If one or a few faces result in a large error, this could lead to the rejection of a measurement, which might still be beneficial for the prediction of most faces. To avoid this, we also measure the percentage D_{NI} of faces for which an error improvement is seen. We count the number of faces whose geometric errors have been decreased by considering the candidate measurement. We then normalize D_{GE} and D_{NI} to the $[0, 1]$ range and combine them into a single reconstruction quality measure:

$$quality = normalize(D_{NI}) + 1 - normalize(D_{GE}). \tag{4.10}$$

Considering the combined geometric error and percentage of improvement of all candidate measurements, we pick the one which will be added to the set of selected measurements. We stop adding measurements when we observe an increase of D_{GE} and a value D_{NI} below 50%. We repeat this process for each part (eyes, nose, mouth, etc.)

The selected anthropometric measurements are enumerated in Table 4.3. The description of each measurement, as well as the reference to the literature from which we obtained the measurement, are shown in Table 4.2, where we also list the measurements we rejected (measurements which were never selected for any of the segments).

Table 4.3 Combinations of anthropometric measurements

For female	
Part	Selected measures
Facial mask	B, BB, BW, BWFH, FH, MaxFB, NBWN, NH, NP, PW, WNTP
Eye	B, BB, BW, BWFH, EH, F, FW, ID, LL, NB, NBTP, NH, NRLB, NRTP, NWRW, NWWN, OBW, OFL, OIW, PW
Nose	EH, LL, N, NB, NBTP, NBWN, NH, NL, NP, NWTP, NWWN, PW
Mouth	BWFH, F, LL, MFW, NP, NRWN, NWTP, PW
Ear	EH, FH, FW, MaxFB, NB, NBTP, NBWN, NP, NRLB, NRTP, NWLB, NWTP, OBW, PW, WNTP
For male	
Part	Selected measures
Facial mask	BW, BWFH, F, FH, FW, MaxFB, NRWN, OBW, OFL, PW
Eye	B, BB, BW, F, FW, ID, N, NBTP, NBWN, OBW, OFL, PW
Nose	BB, FW, ID, LL, MaxFB, NB, NBWN, NH, NP, NRLB, NRTP, NWLB, NWWN, OBW, OIW
Mouth	B, BW, BWFH, F, LL, NBTP, NH, PW
Ear	B, BB, BWFH, EH, FH, MaxFB, N, NRTP, NWRW, OBW, OIW

4.3.3 Correlation Between Measurements

Defining the correlation between the measurements is important for the adjustment of faces. Accordingly, if the user adjusts one measurement, the system automatically calculates the adjustment of the other measurements as well. This greatly helps to create realistic faces by maintaining the correlation observed in the data set. Similarly to Body Talk (Streuber *et al.*, 2016), we use Pearson's correlation coefficient on F to evaluate the relationship between the anthropometric measurements. Considering a facial part d, the Pearson's correlation coefficient Cor_{jk} for measurements j and k is expressed as:

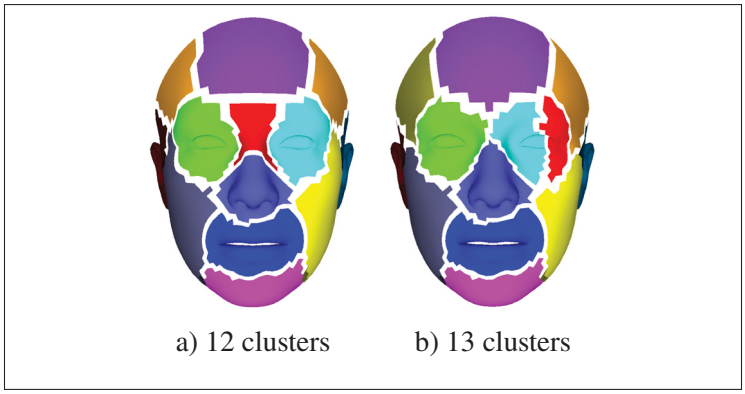
$$\mathrm{Cor}_{jk} = \frac{\sum_{i=1}^{n_s}(f_{ij} - \overline{f_j})(f_{ik} - \overline{f_k})}{\sqrt{\sum_{i=1}^{n_s}(f_{ij} - \overline{f_j})^2}\sqrt{\sum_{i=1}^{n_s}(f_{ik} - \overline{f_k})^2}}, \tag{4.11}$$

where $f_{ij}, f_{ik} \in f_{d,i}$ are measurements of scan S_i, $\overline{f_j}$ and $\overline{f_k}$ are the mean values of measurements j and k respectively, and n_s is the number of scans. The coefficient is a value between -1 and 1 that represents the correlation. When adjusting measurement k by Δf_k, we get the change for other measures as $\Delta f_j = \mathrm{Cor}_{jk}\Delta f_k$. Accordingly, we can evaluate the influence of one

measurement on the others, as well as the conditioning on one or more measurements, and create the most likely ratings of the other measurements.

4.4 Results

Compared to global 3DMM methods that compute one set of eigenvectors for the whole face, our 3DMM computes a set of eigenvectors for each part. This is at the root of one of the advantages of our approach: its ability to locally adjust faces. We compare our approach to other methods that rely on local 3DMM. We created mapping matrices (Eq. 4.7) for global 3DMMs, SPLOCS (Neumann *et al.*, 2013), clustered PCA (Tena *et al.*, 2011), as well as our part-based 3DMMs, and tested the adjustment of measurements with these models. We used 46 eigenvectors for global 3DMM, SPLOCS, and our part-based 3DMM. For clustered PCA, we first tested using 13 clusters, as is reported in their paper, but found that this leads to a non-symmetrical result (Fig. 4.10b). By checking other clusterings, we selected 12 clusters (Fig. 4.10a). Because a clustered PCA does not allow for a different number of eigenvectors for each cluster, and to avoid having too few eigenvectors per part, we used 46 eigenvectors for each cluster (selecting the 46 with the largest eigenvalues).



a) 12 clusters b) 13 clusters

Figure 4.10 Automatic part identification of clustered PCA

To compare our approach and the use of measurements with other methods, we decided on a way to use our measurements with SPLOCS and clustered PCA. We further demonstrate that

with SPLOCS and our approach, we can have more *local measurement* or *global measurement* control. For our approach, Table 4.3 shows that some measures influence more than one part. For example, the "Lip Length" is found in the lists for both mouth and nose. When a measurement is shared between different facial parts, our method allows to decide to have more localized changes by adjusting the measure for only one part, or to have more coherence across the parts by adjusting all of the parts involved in the measurement. If comparing with SPLOCS, we can also balance between local measurements and global measurements. Each measurement is based on computations involving specific *measurement vertices* (such as the corner of the mouth and the tip of the nose). To enforce locality, when considering a measurement, we check which SPLOCS "eigenvectors" infer significant movement at the related *measurement vertices*. We compute this by checking if the eigenvector displacement vector at a *measurement vertex* is large enough as compared to the maximum displacement vector of the eigenvector (we check if it is larger than 1% of the maximum displacement of all vertices of the eigenvector). A SPLOCS eigenvector is considered for a measurement only if it meets the criterion for one of the *measurement vertices* of a specific measurement. To enforce more globality with SPLOCS, we use the mapping matrices for all of the eigenvectors. Fig. 4.11 shows an example of the globality and locality of the influence of adjusting the "Lip Length". It compares global PCA eigenvectors, local measurement and global measurement SPLOCS, clustered PCA, and our local measurement and global measurement approaches. The color coding shows the per-vertex Euclidean distance. Note that the colors do not represent errors, but rather, vertex movements (blue = 0 mm, red = 8.5 mm). Thus, the goal is to have warmer colors around the location where the editing is intended, and colder colors in unrelated regions. Our method allows having global measurement influenced by adjusting the measure for both the nose and the mouth parts, as well as more localized changes by adjusting only the mouth (Figs. 4.11c-4.11f). Contrary to our approach, both global measurement and local measurement SPLOCS resulted in similar deformations all over the face, while the expected result was a modification focused around the mouth (Figs. 4.11b-4.11d).

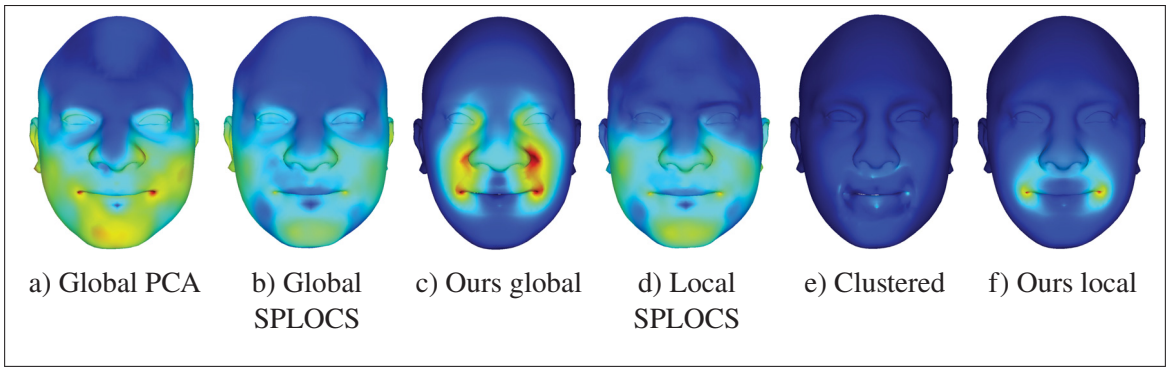


Figure 4.11 Comparison of the globality vs. locality of the adjustments (editing by increasing the "Lip Length")

We will now focus on local measurement editing. Figs. 4.12-4.14 show the adjustment of the same anthropometric measurement using global 3DMM, local measurement SPLOCS, clustered PCA, and our local measurement approach. In Figs. 4.12f-4.12g, we can see that even though we wanted to adjust the "Nose Breadth", the adjustment using the global eigenvectors and local measurement SPLOCS resulted in significant deformations all over the face, while clustered PCA and our approach could focus the deformation around the nose, as expected (Figs. 4.12h-4.12i). We can observe similar unwanted global deformations of the face in Figs. 4.13f-4.13g. Also note that the automatic segmentation of clustered PCA does not provide the desired deformation for some cases, such as in Figs. 4.13h-4.14h. We consistently outperform clustered PCA in terms of local deformation where expected. The results shown in Figs. 4.12-4.14 highlight the difficulty of locally controlling the face deformation, and the power of our approach in locally adjusting the face with respect to the anthropometric measurements.

In the accompanying video, we show multiple edits on multiple parts, starting from the average face, while Fig. 4.15 shows edits starting from four real faces. We can see that our approach allows capturing the essence of the anthropometric measurements, providing an easy-to-use workflow.

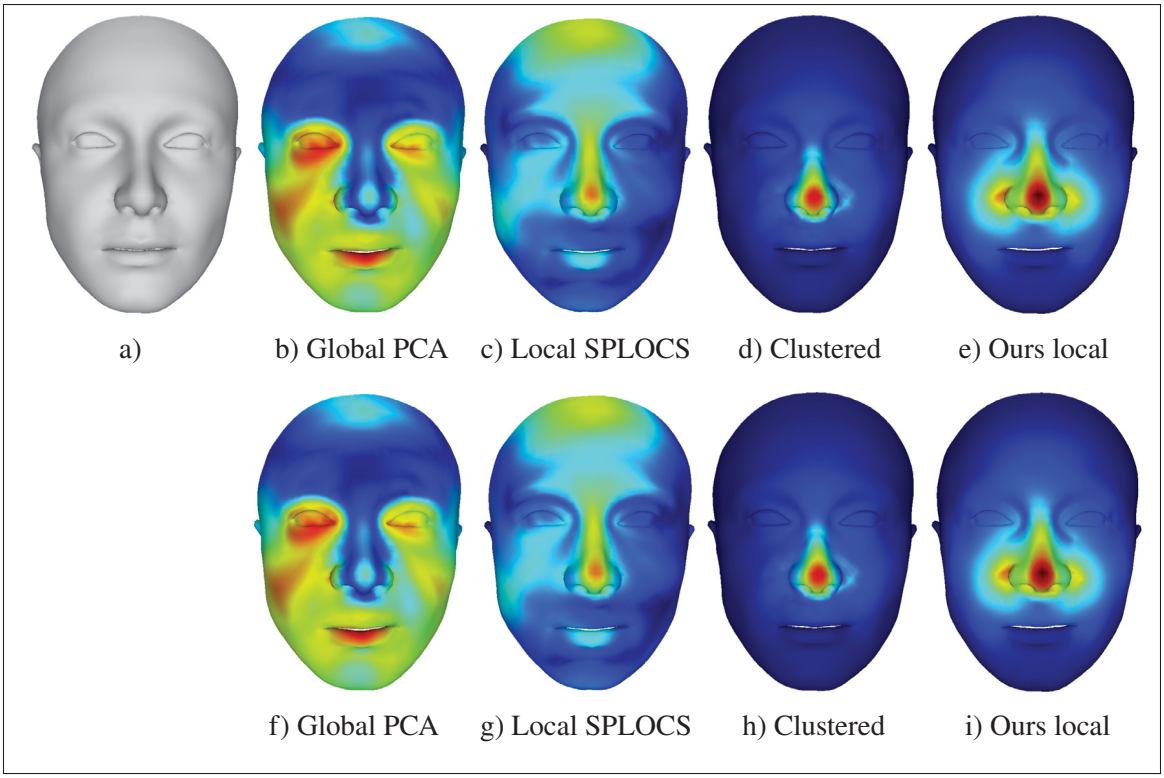


Figure 4.12 "Nose Breadth" adjustment results: a nose of a female from validation faces adjusted. The color indicates respective per-vertex Euclidean distance (blue = 0 mm, red = 5 mm)

4.5 Discussion

In this section, we discuss different aspects of our approach. We present different comparisons highlighting the impact of the eigenvector and measurement selection. We then discuss the face segmentation choice, and end by describing the procedure used to bring all of our scans to a common face mesh.

4.5.1 Measurements Error

To verify the robustness of our 3DMMs and of our set of selected measurements, we reconstruct real faces, relying on their anthropometric measurements to compute their eigenvector weights (Eq. 4.6). We then get the face with our approach, including the blending procedure (Sec. 4.2),

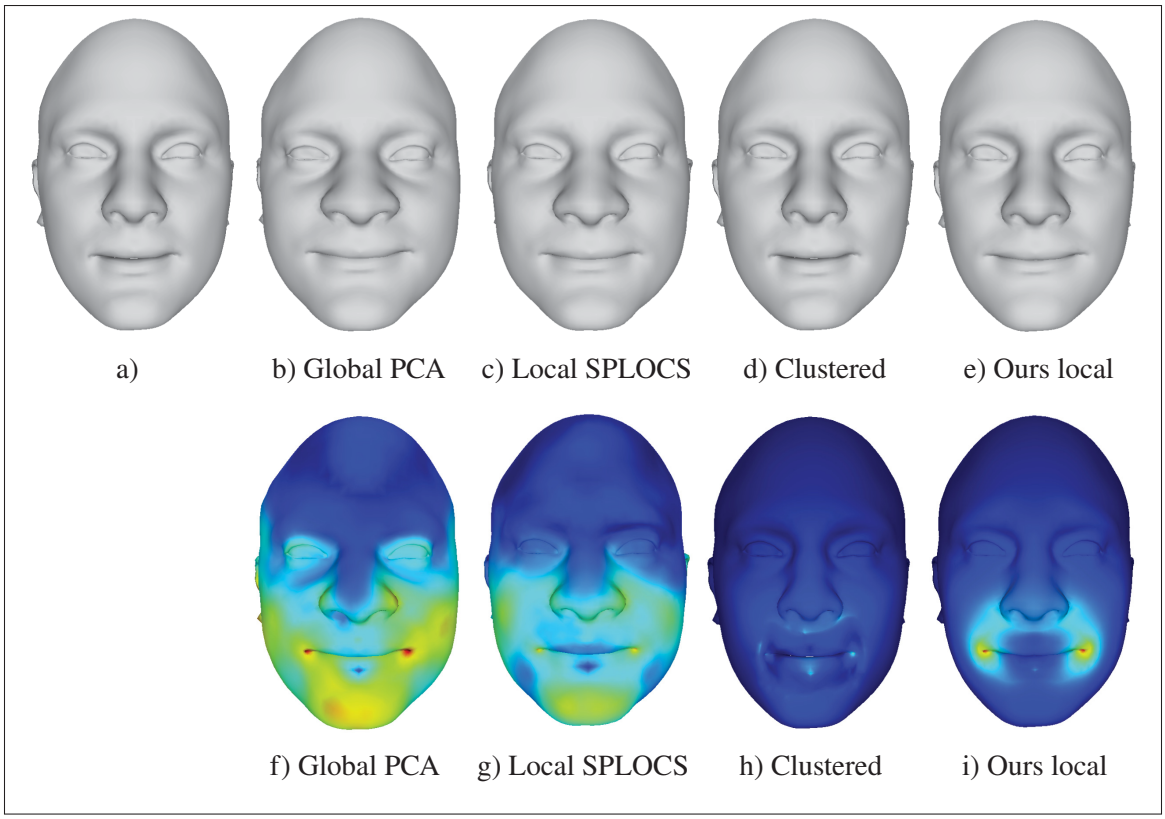


Figure 4.13 "Lip Length" increase results: a mouth of a male from validation faces edited. The color indicates respective per-vertex Euclidean distance (blue = 0 mm, red = 8 mm)

and compute its resulting anthropometric measurements. We compute the quality of the reconstruction through the absolute value of the difference between the ground truth measurement and the measurement from the reconstructed face. Since measurements correspond either to a Euclidean distance or to a ratio of Euclidean distances, we normalized all the measurements to the [0%, 100%] range. Fig. 4.16 shows that the average percentage of error is low when using "our measurements".

This means that both the selection of eigenvectors and the mapping matrix work well. Furthermore, it shows that when using "all measurements" to compute the mapping matrix (Eq. 4.7), we get larger average errors as compared to ground truth measurements. When calculating the error in Fig. 4.16 for "our measurements", we calculate the average error over our selected measurements

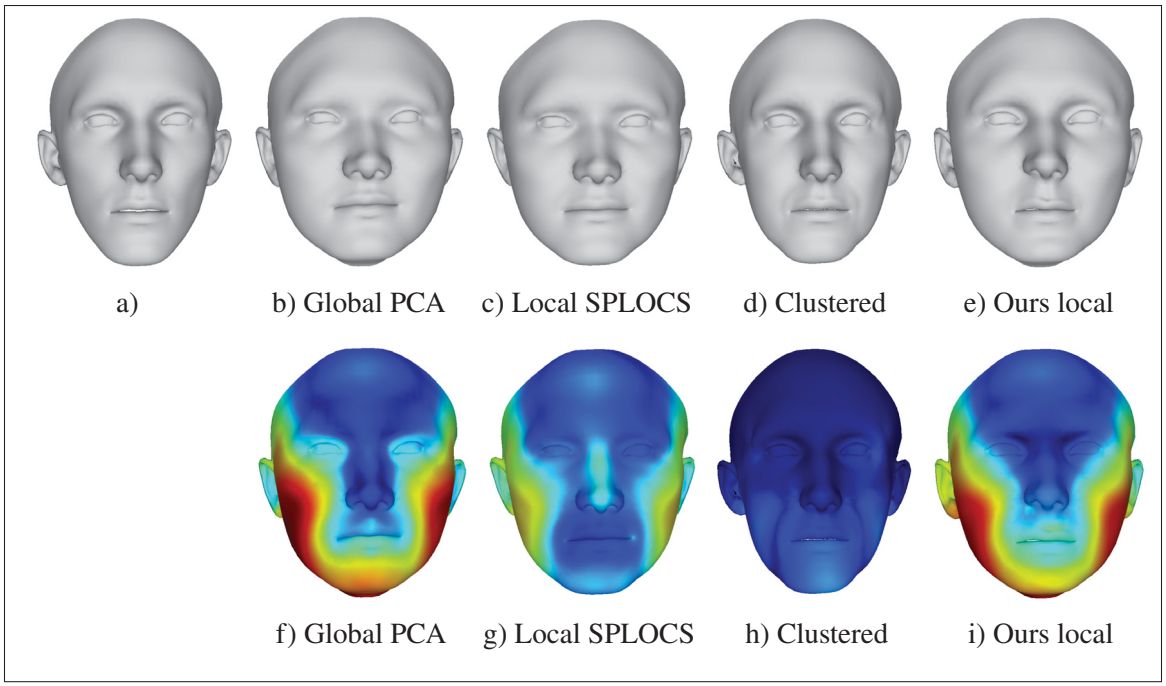


Figure 4.14 "Bizygomatic Breadth" (the bizygomatic width of the face) increase results: a man from the validation faces edited. The color indicates respective per-vertex Euclidean distance (blue = 0 mm, red = 14 mm)

only (Table 4.3). The error shown in Fig. 4.16 for "all measurements" also considers only our selected measurements (if the error across all of the measurements is considered, the comparison is even more in favor of using our selected measurements).

We evaluated how our approach compared to SPLOCS and clustered PCA with respect to achieving measurement values prescribed by edit operations. We created a set of $1,000$ random edits on 135 face meshes. We took the resulting edited face mesh from our approach, SPLOCS, as well as clustered PCA, and evaluate the difference between the measurement value prescribed by the editing and the measurement value calculated from the edited mesh. Overall, our approach is the one that performed the best, with the resulting measurement being closest to the prescribed measurement. SPLOCS was second and clustered PCA presented the greatest differences (see Fig. 4.17).

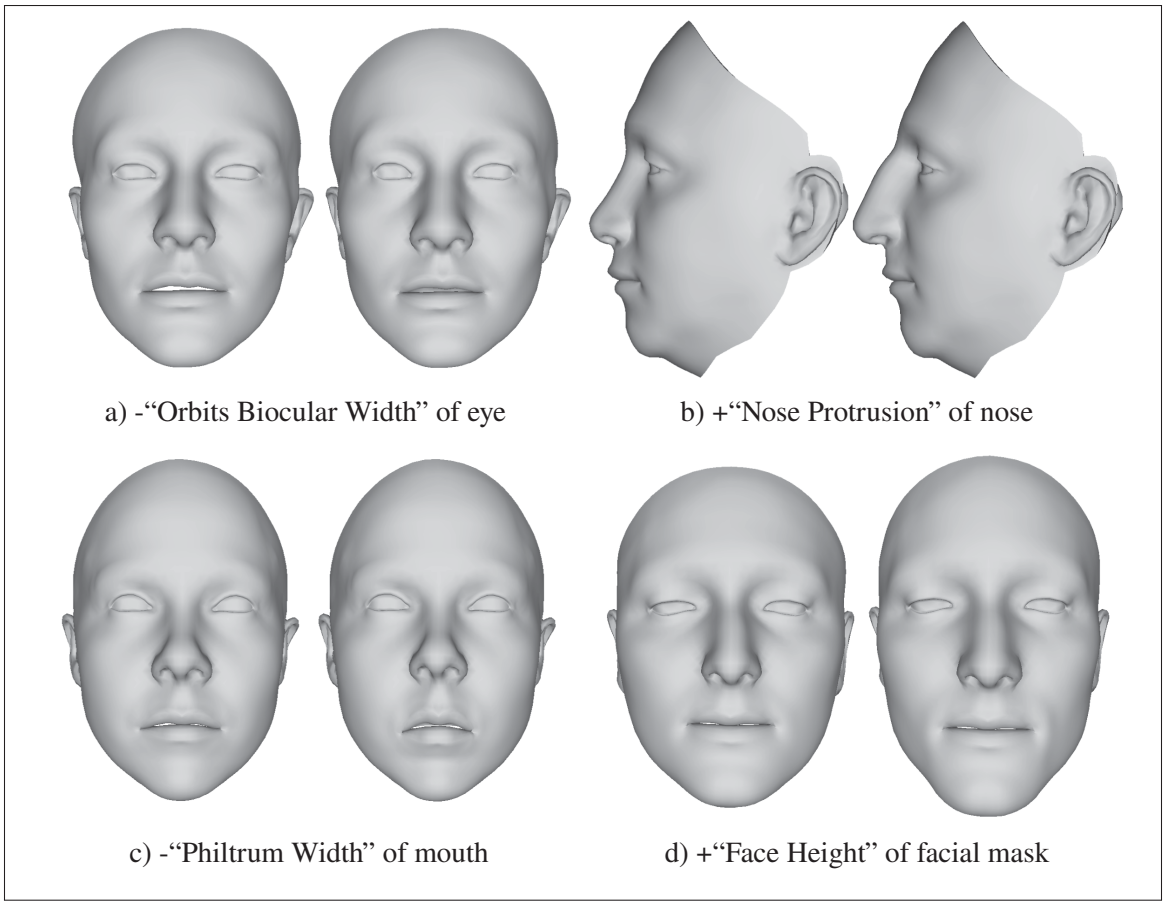


Figure 4.15 We generated random faces (left faces a-d) and edited them by increasing (“+”) or decreasing (“-”) the value of some of the indicated anthropometric measurements

Even though our approach is the one that is closest (on average) to the prescribed measurements, there is a limitation due to the blending of the synthesized parts. This blending sometimes affects the mesh in a way that prevents it from achieving the exact prescribed effect for the editing. Fig. 4.18 shows an example where the blending does not maintain the “Nose Height” of the synthesized nose as it deforms it through the blending process.

4.5.2 Face Decomposition

Our face segmentation was motivated by several facial animation artists with whom we worked, and who strongly prefer having control over the face patches in order to make sure they match

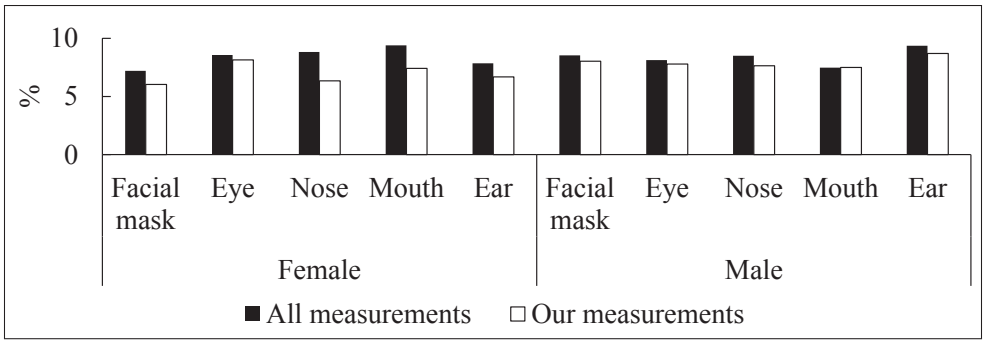


Figure 4.16 Using our subset of measurements on the data set and validation faces leads to lower errors (percentage), as compared to using "all measurements"

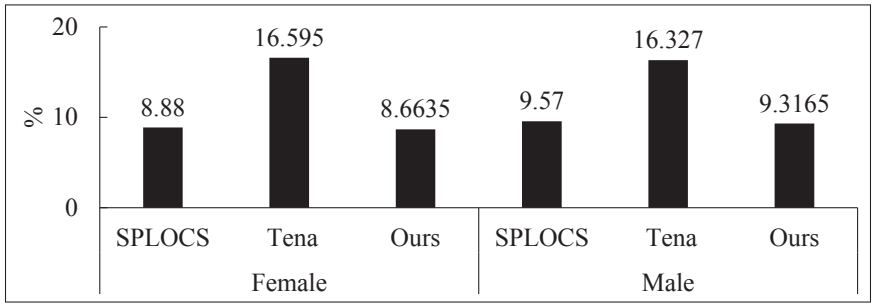


Figure 4.17 1,000 comparisons of different methods to edit measurements. Our approach leads to a smaller error (percentage), as compared to the clustered PCA, and to slightly better results when compared to local measurement SPLOCS

the morphology of the face and muscle locations. This type of control is impossible to achieve with an automatic method, which is typically agnostic to the underlying anatomical structure. It is important to note that this manual way of selecting the regions is no more cumbersome than the current state-of-the-art methods. The state-of-the-art method of Tena *et al.* (2011) requires a post-processing step to fix occasional artifacts in the segmentation method. Furthermore, as illustrated in Fig. 4.10b, segmentation boundaries can occasionally occur across important semantic regions such as the eyes, leading to complications further down the pipeline.

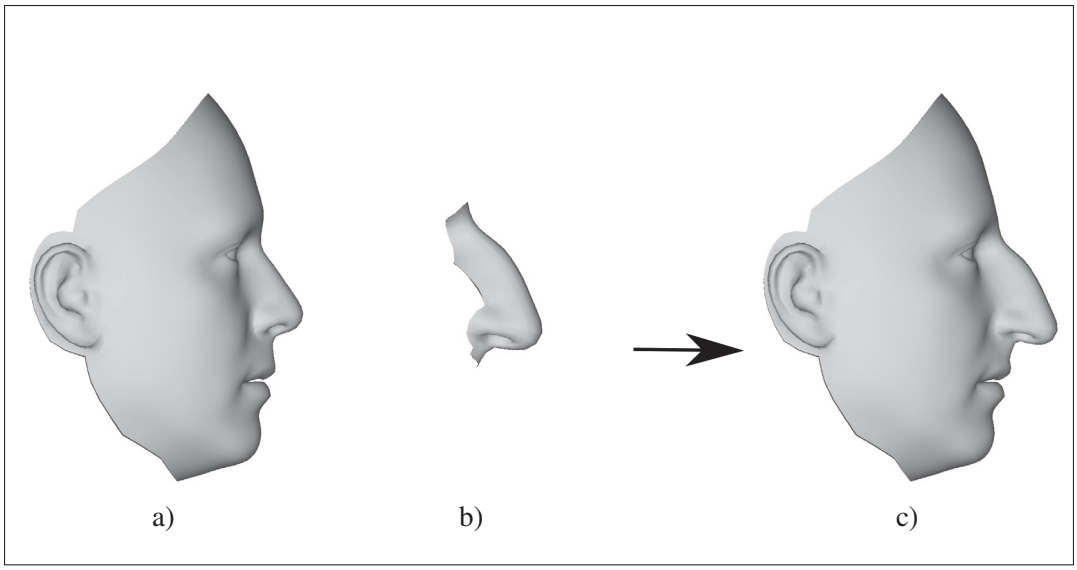


Figure 4.18 a “Nose Height” of the average male head is 45.09 mm. b “Nose Height” of the synthesized nose is 70.11 mm. c “Nose Height” of the blending result is decreased to 58.53 mm

4.5.3 Pose Stabilization

When analyzing the eigenvectors for the different parts, we noticed that some of them were encoding the pose instead of shape. For example, we had parts such as the nose which were rotated or translated, resulting in the eigenvectors encoding the orientation and position. We got a great improvement by doing a rigid alignment of each part to the base mesh. Nevertheless, even with the rigid alignment, one important concern remained; for the mouth, we noticed that some of the eigenvectors were related to the pose of the mouth (slightly opened vs. closed). To address this, we designed a mouth stabilization approach. The idea is to close the mouth by manipulating the mouth inner contour. The manipulation of the mouth contour is propagated to the mandible region (red zone in Fig. 4.19). We apply the Laplacian Surface Editing method (Sorkine *et al.*, 2004) for the mesh deformation. In our energy minimization, we fix the boundary of the mandible zone, we limit the movement of the part of the face which is less affected by the movement of the jaw (yellow zone in Fig. 4.19), we set constraints on the upper and lower contours of the mouth such that they tend to move close to each other, and we solve for the

remaining vertices. Fig. 4.20 compares our proposed approach with a ground truth closed mouth.

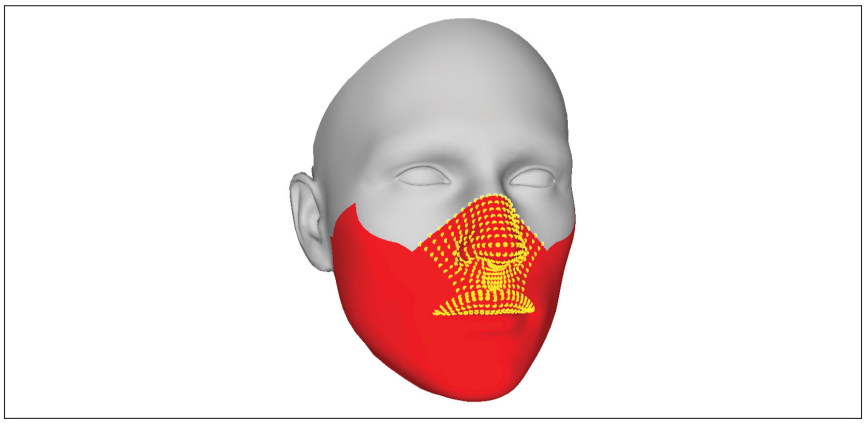


Figure 4.19 Mandible region (highlighted in red) is adjusted by the pose stabilization. We avoid the movement of the part identified in yellow as it tends to be mostly unaffected by jaw movement

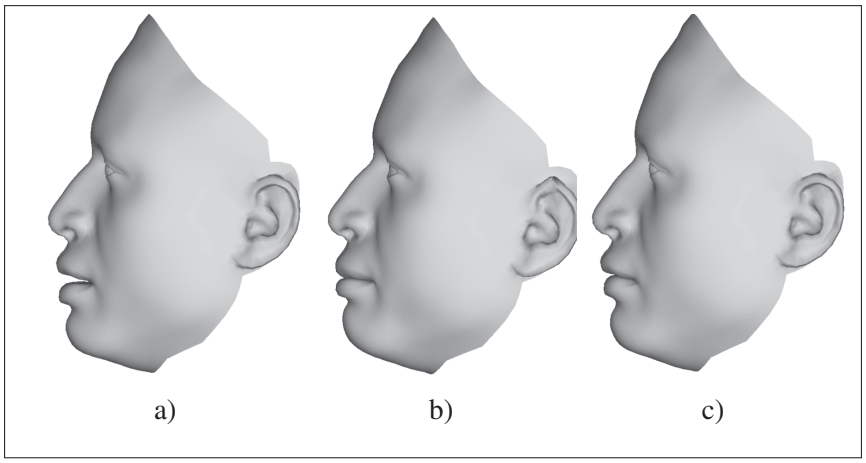


Figure 4.20 a Opened mouth example. We note that b the ground truth closed mouth is quite similar to c the mouth closed by our approach

4.5.4 Data Set

The quality of the input mesh data set is important to the reconstruction of good 3D face models. As many existing methods, we assume that the meshes share a common mesh topology. Mapping the raw 3D scans to a common base mesh is typically done by a surface mapping method (Ramachandran *et al.*, 2018; Zell & Botsch, 2013; Amberg, Romdhani & Vetter, 2007). We established this correspondence with R3DS WRAP.

Our data preparation pipeline involves few steps leading to an automatic surface mapping. Since the 3D scans usually have mesh topology problems such as non-manifold edges, singular vertices, and holes, the first step is to clean the scans. We then seek to automatically find a good frontal camera alignment. As we do not know where the face of the person is in the scan coordinate frame, we generate 42 cameras around the cleaned scan using Fibonacci sphere algorithm.

The frontal pose detector (King, 2009) reads the images from the 42 cameras and scores each detection. After identifying the camera with the best score, this camera is rotated every 18 degrees to find a good camera roll and compensate for the error between the rotation and the detector's trained angle (detector is trained on upright faces). These 42 random cameras provide good landmarks for rigid alignment step but they are not usually good enough for the wrapping step. In practice, there are two problems with this approach; rotation problem and pose problem. In order to improve this step, we use the same random cameras to extract landmarks to align the scan according to the base mesh. We apply a facial alignment step to avoid rotation problem and get a higher accuracy. Regarding the pose problem, we define a camera position based on the frontal pose of the base mesh and use that specific camera to find the angle with highest score. From this frontal camera, we automatically identify a set of 68 facial landmarks utilizing the facial landmark predictor of dlib (Kazemi & Sullivan, 2014). These landmarks are back-projected on the 3D scan meshes (as illustrated in Fig. 4.21) and used to do a rigid alignment to the base mesh coordinate space. In this coordinate space, a frontal pose camera is already set up, providing a very good frontal pose. As the precision of the landmark position depends on the camera orientation, we rerun the facial landmark identification

given the camera frontal pose in the base mesh coordinate space. Widely used facial landmark detectors, such as those of dlib (Kazemi & Sullivan, 2014), do not provide landmarks for the ears. At this point, we rotate the camera 90 degree to left/right for detecting left/right ears' bounding boxes (Castrillón-Santana, Lorenzo-Navarro & Hernández-Sosa, 2011) and we place an initial guess for the ear landmarks (Pflug & Busch, 2012) which we further correct by running an active appearance model, AAM (Alabort-i-Medina & Zafeiriou, 2017) trained on the AMI ear database. Finally, we use the landmarks and aligned scans as inputs to wrap a base mesh with 6014 vertices and 11964 triangles around the high-resolution scans. We also re-project the textures from the scans on the wrapped base mesh.

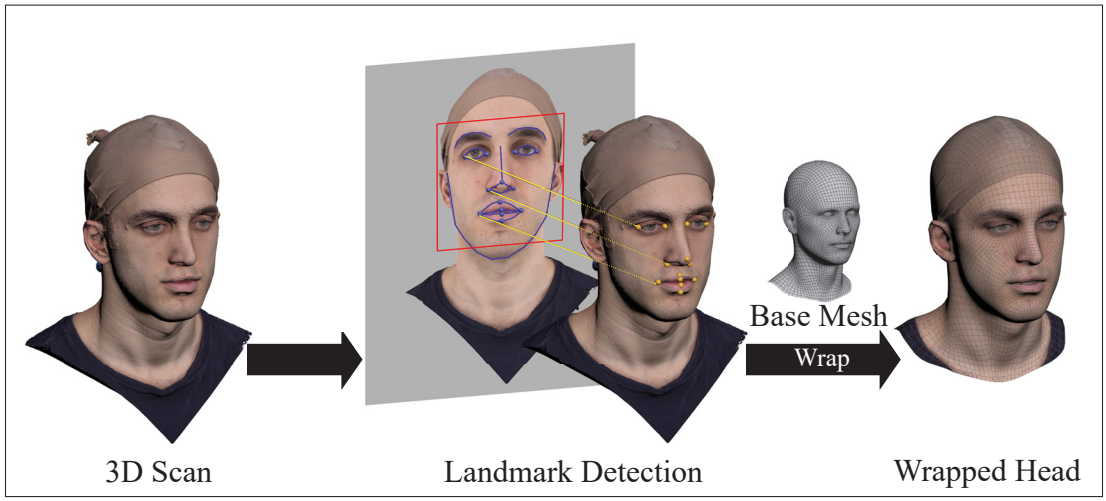


Figure 4.21 We automatically identify landmarks on the scan which will be used to establish the correspondence to the base mesh and create the final wrapped head

Our data set represent a diverse range of ethnicity and ages (Fig. 4.22) which is in contrast to the Basel Face Model (Paysan, Knothe, Amberg, Romdhani & Vetter, 2009).

In this chapter, we designed a new local 3DMM used for face editing. We demonstrated the difficulty of locally editing the face with global 3DMMs; we thus segmented the face into five parts and combined the 3DMMs for each part into a single 3DMM by selecting the best eigenvectors through prediction error measurements. We then proposed the use of established anthropometric measurements as a basis for face editing. We mapped the anthropometric

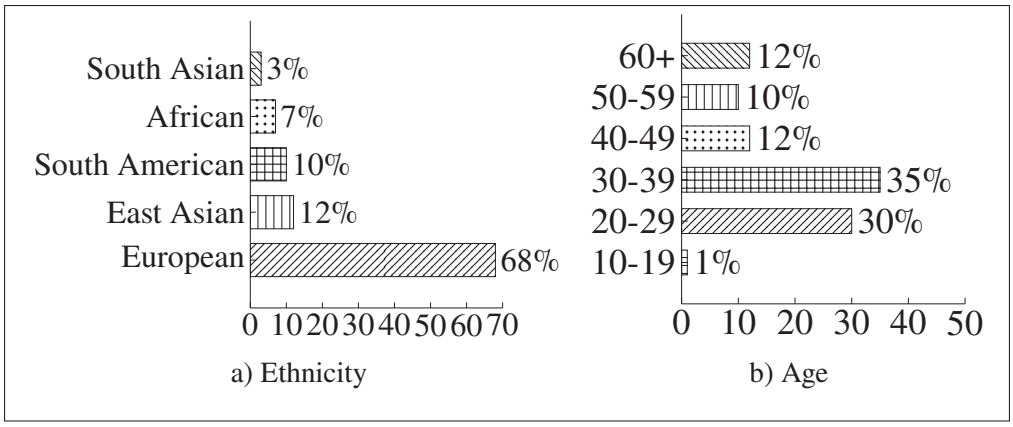


Figure 4.22 Racial and age distribution of our data set. It includes 135 3D scans consisting of 64 females and 71 males. a shows the racial distribution of these heads while b shows different age groups in the data set

measurements to the 3DMM through a mapping matrix. We proposed a process to select the best set of anthropometric measurements, leading to improved reconstruction accuracy and the removal of conflicting measurements. From a list of 33 anthropometric measurements we surveyed from the literature, we identified 31 which lead to an improvement of the reconstruction and rejected 2 as they decreased the quality of the reconstruction. Note that the anthropometric measurement selection process would apply as well even if using a different 3DMM from the one proposed here, as well as when considering a different set of anthropometric measurements. We demonstrated this by applying our set of measurements to both SPLOCS (Neumann *et al.*, 2013) and clustered PCA (Tena *et al.*, 2011). This also demonstrated that our approach produces results superior to those of established methods proposing automatic segmentation and different ways to construct the eigenvector basis. We also presented different bits of experimental evidence to demonstrate the superiority of our approach, especially in terms of local control, as compared to the typical global 3DMM.

CONCLUSION AND RECOMMENDATIONS

In this thesis, we have discussed different objectives to reuse available data instead of creating them from scratch. We have presented an animation setup paper (Avril *et al.*, 2016) to transfer skeletons and skinning weights between characters by deforming the attributes in a way that they fit the new character very well even with distinct mesh topologies. It provides an automatic retargeting pipeline but the results depend greatly on how well the given dense geometric correspondence between the two surfaces is. This dependency led us to propose a novel approach Ramachandran *et al.* (2018) concentrating on a dual flattening geometric correspondence capable of handling a wide range of surfaces but similar to current state-of-the-art methods it does not result in a perfect mapping in every case. Accordingly on our follow-up work, we have introduced and discussed a novel approach that complements the proposed approach by editing the mapping where the user wants it (Ghafourzadeh *et al.*, 2020b). Our approach improves the mapping in the related regions without causing a degradation of the overall mapping. It can greatly improve the results of techniques relying on a geometric correspondence such as animation setup transfer as well as wrapping 3D scans to a common base mesh for 3DMMs. Finally, we have presented 3DMMs paper (Ghafourzadeh *et al.*, 2020a) which provides local deformation to construct realistic faces. The proposed approach has excellent generative properties and allows the user intuitive local control which is one of the main limitation of the current face modeling methods using 3DMM.

5.1 Summary of the contributions

We have presented a new approach for transferring an existing skeleton and skinning weights from a source character to a variety of target character meshes. The approach is flexible as it eliminates typical modeling constraints by handling characters with meshes of various topologies and morphologies. Moreover, this approach provides several benefits compared to creating the skeleton and weights from scratch, which can be tedious, requiring lengthy manual

adjustments. This approach leads to a novel pipeline to quickly and efficiently prepare new characters for animation. A fundamental problem of transferring animation is to establish an accurate geometric correspondence between the two characters since the state of the art methods that exist today are globally adequate, but wrong in localized regions which are unacceptable in a production setting. In fact, no single method results in a perfect mapping in every case and the imperfections can be localized to a specific region. While a lot of research has been done on creating correspondences between characters, comparatively fewer methods have been proposed on mapping editing. Accordingly, we have presented a novel solution that improves the mapping locally and ensures that it does not introduce artifacts elsewhere on the map. The approach is based on a low-distortion region-growing segmentation followed by an independent planar parameterization of each segment. The mapping is then optimized based on an alignment of the user-prescribed landmarks in the parameterization space of each segment. Our approach has also opened the door to a new family of deformation methods that optimize the deformation by iterative conformal energy minimization. We have presented a novel joint planar parameterization deformation for the segments that is robust, and results in low distortion. Our new iterative LSCM approach can be reused in several contexts where a deformation with low distortion is required.

Besides animating a character, modeling a character's face is particularly time-consuming due to the complexity of facial features. Our presented ideas of local editing and reusing available data are both well suited for modeling 3D faces as well. We have presented a new approach that aids modeling by providing high-level controls, such as anthropometric measurements, to edit locally available human-like character heads. We thus segmented the face in five parts and the main idea is to select the best eigenvectors for each part. After editing, we combine the parts into a single 3DMM. We then proposed the use of established anthropometric measurements as a basis for the face editing. We proposed a process to select the best set of anthropometric measurements, leading to improved reconstruction accuracy and the removal of conflicting measurements.

We demonstrated that our approach produces results superior to those of established methods proposing automatic segmentation and different ways to construct the eigenvector basis.

5.2 Future work

The results of animation transfer suggest several future directions. One direction for future work is to deal with facial bones. As such, after designing a new face with the 3DMM, the facial joints needed for animation would be automatically positioned and afterwards the skin weights would be transferred. Accordingly, this would lead to an automatic pipeline that intelligently outputs new facial models ready for animation. Another direction is to work on transferring other elements of the animation setup, such as blend shapes that are often used in facial animation.

Our proposed dense correspondence and local editing approaches lead to interesting questions which open many avenues for future work. One such prospective area is higher-level landmarks such as lines. This will lead to challenges in terms of easing the interactive placement of these lines on both meshes, but will provide a better set of constraints for the deformation. The cage deformation approach aligns interior vertices more naturally but it does not guarantee that the deformed mesh will be free of flipped faces which would destroy the injectivity of the mapping. On the other hand, it does not provide a low-distortion deformation since it was not minimizing a distortion energy. Accordingly, applying a deformation energy that guarantee injectivity as well as having low conformal distortion such as our proposed ILSCM or SLIM could be an interesting avenue for future work. Another avenue would be to extend the scope to editing deformation transfer. This would combine deformation with editing and enable the user to control animation retargeting.

About 3DMMs approach, one of its main limitation is the mapping matrices that assume a linear relationship between anthropometric measurements and the eigenvector weights. An interesting avenue for future work would be to apply machine learning to identify non-linear mappings.

Also, our measurements are based on distances between points on the surface. Future work could consider measurements based on the curvature over the face, such as measurements specifying the angle formed at the tip of the chin. On the other hand, although anthropometric measurements generate plausible facial geometric variations, they do not consider fine-scale nor coarse-scale features. Regarding the fine-scale details, our approach does not model realistic variations of wrinkles that can be an interesting direction for future research. Regarding coarse-scale features, we could reconstruct a skull based on the anthropometric measurements, and then generate the facial mask based on an energy minimization of the skin thickness considering the skull and the measurements.

APPENDIX I

PROOF

To ensure that there is always a single solution, even if λ is arbitrarily small, we add a new term $E_{B\star}$ to Eq. 3.1:

$$E(V) = E_L(V) + E_B(V) + E_{B\star} + E_D(V) \tag{A I-1a}$$

$$E_{B\star}(V) = \xi \sum_{j \in \Omega(\widetilde{B}_i)} \left\| v_j - v_j^{old}, \right\|^2 \tag{A I-1b}$$

where v_j^{old} denotes the position of vertex v_j at the previous iteration. The energy E_B pulls the vertices of the boundary to where they correspond given the mapping. The term $E_{B\star}$ pulls the vertices on the boundary of $\widetilde{B}_i$ to their position at the previous iteration. Eq. A I-1b is weighted by a small constant $\xi = 0.001$ such that in practice the vertices will converge to map(v_j). The previous position v_j^{old} is initialized with the position on the boundary of the ABF of $\widetilde{B}_i$.

Our deformation method proceeds iteratively by finding a sequence of 2D embeddings V_i of a given patch. We show that if the initial embedding of the mesh V_0 has no fold-overs, then the resulting embedding at every iteration V_i also has no fold-overs. We prove this by induction. The base case for $i = 0$ is given by the hypothesis, and thus, we are showing that if V_i has no fold-overs, our procedure will yield a configuration V_{i+1} that also has no fold-overs.

At every iteration, the new set of vertex positions V_{i+1} is obtained by solving Eq. 3.1:

$$\arg\min_{V_{i+1}}(E(V_i, V_{i+1}, \bar{\lambda})$$

where V_i is the embedding in the current iteration, V_{i+1} is the new embedding we are computing, and $\bar{\lambda} > 0$ is a parameter of the algorithm, constant w.r.t. this minimization. We select $\bar{\lambda}$ as follows: we create a monotonically decreasing positive sequence $\lambda_j > 0$ such that $\lim_{j\to\infty}(\lambda_j) = 0$; we solve the optimization problem for the λ_j in the sequence and stop at the first element in the sequence $\bar{\lambda} = \lambda_k$ that yields a fold-over free configuration, and we now show that such a λ_k always exists.

Let $B(x) \in \mathbb{R}^{n\times n}$, $x \in \mathbb{R}$ and $B_{ij}(x)$ is continuous in x $\forall i, j$.

Lemma 0.1. *$det(B(x))$ is a continuous function in x.*

Proof: We prove by induction on n. If $n = 1$, $B(x)$ is a continuous real function. $det(B(x)) = B_{1\,1}(x)$ is also continuous. We assume that the statement is true for $n - 1$ and we prove for n. We write $det(B(x))$ using the Laplace formula:

$$det(B(x)) = \sum_{i=1}^{n} (-1)^{i+j} \cdot B_{ij}(x) \cdot M_{ij}(x) \tag{A I-2}$$

where M_{ij} is the minor of the entry (i, j) defined as the determinant of the sub-matrix obtained by removing row i and column j from B. As each element of this matrix is also continuous in x and this reduced matrix is $n - 1 \times n - 1$, it follows from the inductive hypothesis that M_{ij} is also continuous in x. As $det(B(x))$ is obtained by using addition and multiplication of continuous functions, it follows that $det(B(x))$ is continuous in x. □

Corollary 0.1.1. *if $det(B(x)) \neq 0$ $\forall x$ then $\frac{1}{det(B(x))}$ is also continuous in x.*

Lemma 0.2. *if $det(B(x)) \neq 0$ $\forall x$ then $B_{ij}^{-1}(x)$ is a continuous function in x $\forall i, j$.*

Proof: If $det(B(x)) \neq 0$ $\forall x$ then $B(x)$ is invertible $\forall x$ the inverse of a matrix has the following analytic expression:

$$B^{-1}(x) = \frac{1}{det(B(x)))} \cdot C^{\intercal} \tag{A I-3}$$

where C is the matrix of co-factors: $C_{ij} = (-1)^{i+j} \cdot M_{ij}$ and M_{ij} is the minor of the entry (i, j). $M_{ij}(x)$ is continuous in x from Lemma 0.1. It follows trivially that C_{ij} is continuous in x. Since $det(B(x)) \neq 0$ $\forall x$ it follows from Corollary 1 that $\frac{1}{det(B(x)))}$ is continuous in x. Since $B^{-1}(x)$ is obtained by multiplying a scalar function continuous in x by a matrix whose entries are all continuous in x, it follows that $B_{ij}^{-1}(x)$ is continuous in x $\forall i, j$. □

$V_{i+1}(\lambda)$ is the minimizer of a quadratic energy function, and therefore, it has the standard least squares analytical solution:

$$V_{i+1}(\lambda) = (A(\lambda)^\intercal \cdot A(\lambda))^{-1} \cdot A(\lambda)^\intercal \cdot b \tag{A I-4}$$

where the matrix A and vector b are computed from Eq. 3.1 in a standard way for a least squares solution. The matrix $A(\lambda)$ has the following structure:

$$A(\lambda) = \begin{bmatrix} A_1(\lambda) \\ A_2(\lambda) \\ A_3(\lambda) \\ A_4(\lambda) \end{bmatrix} \tag{A I-5}$$

where:

1. A_1 corresponds to E_L and is a $k \times n$ matrix that encodes the landmark constraints of the patch. These constraints are weighted by λ (Eq. 3.2a).
2. A_2 corresponds to E_B and is a $b \times n$ matrix that encodes the boundary constraints of the patch. These constraints are weighted by λ (Eq. 3.2b).
3. A_3 corresponds to $E_{B\star}$, and is a $2 \times n$ matrix that constrains two boundary vertices to their positions in the previous iteration. These constraints are weighted by a small constant ξ independent of λ (Eq. A I-1b).
4. A_4 corresponds to E_D and is the $m \times n$ matrix from the original LSCM formulation (Lévy *et al.*, 2002), where n is the number of vertices in the patch and $m > n$.

Lemma 0.3. *$A(\lambda)(i, j)$ is a continuous function of λ $\forall i, j$.*

Proof: The matrix $A(\lambda)$ is constructed by stacking four matrices: A_1, A_2, A_3 and A_4 shown above. A_3 and A_4 are independent of λ and therefore all their entries are continuous w.r.t. λ. The entries of A_1 and A_2 are either 0, and therefore continuous in λ, or a linear function of λ, as in Eq. 3.2a and 3.2b, and therefore also continuous in λ. □

Corollary 0.3.1. *$A^\intercal(\lambda)(i, j)$ is a continuous function of λ $\forall i, j$.*

Proof: As the entries of $A^\intercal(\lambda)(i,j)$ are the same as for $A(\lambda)(i,j)$ this follows trivially from Lemma 0.3. □

Corollary 0.3.2. $(A^\intercal(\lambda) \cdot A(\lambda))(i,j)$ *is a continuous function of* $\lambda\ \forall i,j$.

Proof: As $(A^\intercal(\lambda) \cdot A(\lambda))(i,j)$ is obtained by multiplying and adding elements of A and $A^\intercal$ that are all continuous in λ, it follows that $(A^\intercal(\lambda) \cdot A(\lambda))(i,j)$ is a continuous function of $\lambda\ \forall i,j$. □

Corollary 0.3.3. $(A^\intercal(\lambda) \cdot b)(i)$ *is a continuous function of* λ, $\forall i$.

Proof: $b(i)$ is constant in $\lambda\ \forall i$. $(A^\intercal(\lambda) \cdot b)(i)$ is obtained by multiplying and adding elements of b and $A^\intercal$ that are all continuous w.r.t. λ, and it therefore follows that $(A^\intercal(\lambda) \cdot b)(i)$ is a continuous function of $\lambda\ \forall i$. □

Lemma 0.4. $det(A^\intercal(\lambda) \cdot A(\lambda)) \neq 0\ \forall\lambda$

Proof: The LSCM paper (Lévy *et al.*, 2002) shows that if we constrain exactly 2 vertices, we obtain a unique solution, which means that:

$$A_{34} = \begin{bmatrix} A_3(\lambda) \\ A_4(\lambda) \end{bmatrix} \tag{A I-6}$$

has rank n. Since the matrix A_{34} does not depend on λ, it follows that $rank(A_{34}) = n\ \forall\lambda$. Since the rank of A_{34} cannot be larger than n, and the rank of the resulting matrix does not decrease by adding rows to the matrix, it follows that when stacking A_1 and A_2 to A_{34} to form the final matrix A, $rank(A(\lambda)) = n\ \forall\lambda$. Since the rank of the Gramm matrix is the same as that of the matrix, it follows that $rank(A^\intercal(\lambda) \cdot A(\lambda)) = n\ \forall\lambda$. Since $A^\intercal(\lambda) \cdot A(\lambda)$ is a $n \times n$ matrix, this means that A has full rank, therefore $det(A(\lambda)) \neq 0\ \forall\lambda$. □

Lemma 0.5. $V_{i+1}(\lambda)$ *is continuous in* λ.

Proof: $V_{i+1}(\lambda) = (A(\lambda)^\intercal \cdot A(\lambda))^{-1} \cdot A(\lambda)^\intercal \cdot b$. From Lemmas 0.2, 0.3.2, and 0.4, it follows that $(A(\lambda)^\intercal \cdot A(\lambda))^{-1}(i,j)$ is continuous in $\lambda\ \forall i,j$ and from Corollary 0.3.3 $(A^\intercal(\lambda) \cdot b)(i)$ is a

continuous function of λ $\forall i$. As each element of $V_{i+1}(\lambda)$ is computed as a sum or product of functions continuous in λ, it follows that $V_{i+1}(\lambda)$ is continuous in λ. $\square$

Lemma 0.6. *if* $\lambda = 0$*, then* $V_{i+1} = V_i$

Proof: $\lambda = 0$ reduces the linear system to only A_3 and A_4. $\square$

Theorem 0.7. $\exists k > 0$ *s.t.* $V_{i+1}(\lambda_k)$ *has no fold-overs.*

Proof: From Lemma 0.6, if $\lambda = 0$, then V_{i+1} has no fold-overs. Since V_{i+1} is continuous in λ (Lemma 0.5), it follows that for all vertex positions $\exists \hat{\lambda} > 0$ *s.t.* $\forall \lambda, 0 < \lambda < \hat{\lambda}, V_{i+1}(\lambda)$ has no fold-overs. Since the sequence λ_j is monotonically decreasing and $\lim_{j \to \infty}(\lambda_j) = 0$, it follows that $\exists k$ *s.t.*$0 < \lambda_k < \hat{\lambda}$. It follows that $V_{i+1}(\lambda_k)$ has no fold-overs. $\square$

By proving Theorem 0.7, we show that at every iteration, our embedding V_{i+1} has no fold-overs and thus yields an injective map.

APPENDIX II

FAILURE CASES

Our deformation method guarantees progress toward meeting the landmark constraints while being free of flipped faces, but it cannot guarantee that the user constraints will be satisfied. In fact, there are cases where it is impossible to meet these constraints, such as the example in Fig. II-1. There are also "hard" cases (Fig. II-2) where, while it might be possible to find a deformation that meets the constraints, deformation methods, such as LIM, SLIM, KP-Newton and our approach, are not able to find it.

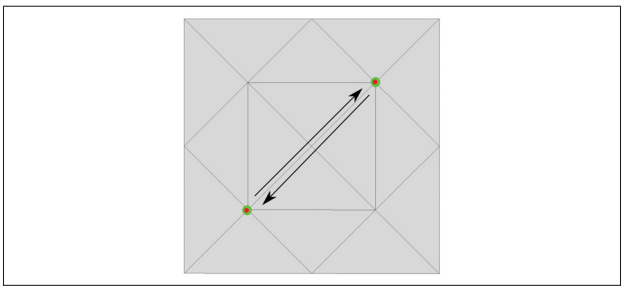


Figure-A II-1 Example of cases where it is impossible to meet the constraints without flipping triangles

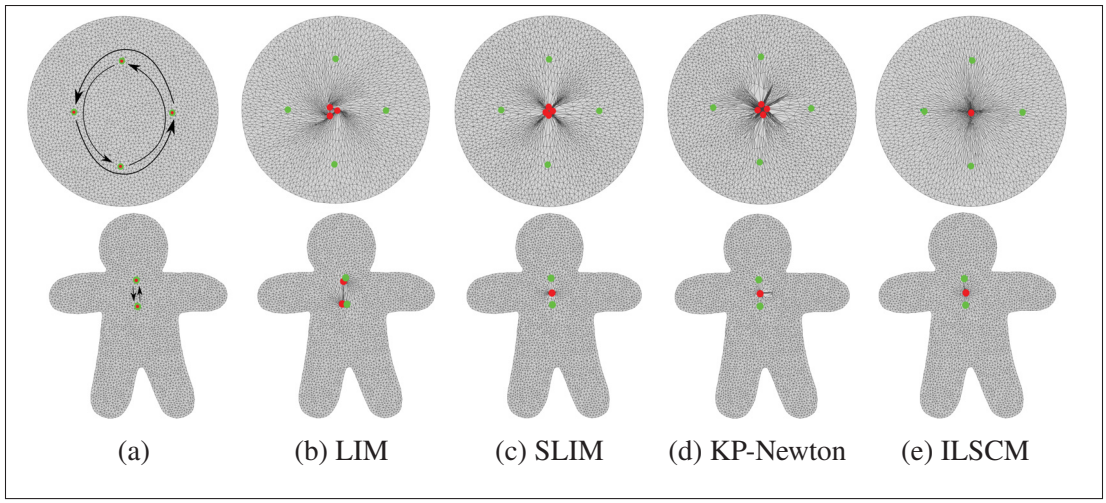


Figure-A II-2 With the design of these landmark configurations, moving the landmarks (red dots) to their final positions (green dots) makes it hard to converge without any flipped faces

BIBLIOGRAPHY

Aigerman, N. & Lipman, Y. (2015). Orbifold Tutte Embeddings. *ACM Trans. Graph.*, 34(6), 190:1–190:12.

Aigerman, N. & Lipman, Y. (2016). Hyperbolic Orbifold Tutte Embeddings. *ACM Trans. Graph.*, 35(6), 217:1–217:14.

Aigerman, N., Poranne, R. & Lipman, Y. (2014). Lifted Bijections for Low Distortion Surface Mappings. *ACM Trans. Graph.*, 33(4), 69.

Aigerman, N., Poranne, R. & Lipman, Y. (2015). Seamless Surface Mappings. *ACM Trans. Graph.*, 34(4), 72:1–72:13.

Aigerman, N., Kovalsky, S. Z. & Lipman, Y. (2017). Spherical Orbifold Tutte Embeddings. *ACM Trans. Graph.*, 36(4), 90.

Alabort-i-Medina, J. & Zafeiriou, S. (2017). A Unified Framework for Compositional Fitting of Active Appearance Models. *International Journal of Computer Vision*, 121(1), 26–64.

Alexa, M. (1999). Merging Polyhedral Shapes with Scattered Features. *International Conference on Shape Modeling and Applications*, pp. 202–210.

Alexa, M. (2002). Recent Advances in Mesh Morphing. *Computer Graphics Forum*, 21(2), 173–198.

Allen, B., Curless, B. & Popović, Z. (2003). The Space of Human Body Shapes: Reconstruction and Parameterization from Range Scans. *ACM Trans. Graph.*, 22(3), 587–594.

Amberg, B., Romdhani, S. & Vetter, T. (2007). Optimal Step Nonrigid ICP Algorithms for Surface Registration. *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8.

Anguelov, D., Srinivasan, P., Koller, D., Thrun, S., Rodgers, J. & Davis, J. (2005). SCAPE: Shape Completion and Animation of People. *ACM Trans. Graph.*, 24, 408–416.

Athanasiadis, T., Fudos, I., Nikou, C. & Stamati, V. (2012). Feature-Based 3D Morphing Based on Geometrically Constrained Spherical Parameterization. *Computer Aided Geometric Design*, 29, 2–17.

Au, O. K.-C., Tai, C.-L., Chu, H.-K., Cohen-Or, D. & Lee, T.-Y. (2008). Skeleton Extraction by Mesh Contraction. *ACM Trans. Graph.*, 27(3), 1–10.

Avril, Q., Ghafourzadeh, D., Ramachandran, S., Fallahdoust, S., Ribet, S., Dionne, O., de Lasa, M. & Paquette, E. (2016). Animation Setup Transfer for 3D Characters. *Computer Graphics Forum*, 35(2), 115–126.

Baran, I. & Popović, J. (2007). Automatic Rigging and Animation of 3D Characters. *ACM Trans. Graph.*, (3), 72.

Blanz, V. & Vetter, T. (1999). A Morphable Model for the Synthesis of 3D Faces. *In Proceedings of SIGGRAPH 99*, (Annual Conference Series), 187–194.

Bogo, F., Romero, J., Loper, M. & Black, M. J. (2014). FAUST: Dataset and Evaluation for 3D Mesh Registration. *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3794-3801.

Booth, J., Roussos, A., Zafeiriou, S., Ponniah, A. & Dunaway, D. (2016). A 3D Morphable Model Learnt from 10,000 Faces. *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5543–5552.

Booth, J., Roussos, A., Ponniah, A., Dunaway, D. & Zafeiriou, S. (2018). Large Scale 3D Morphable Models. *International Journal of Computer Vision*, 126(2-4), 233–254.

Botsch, M. & Sorkine, O. (2008). On Linear Variational Surface Deformation Methods. *IEEE Transactions on Visualization and Computer Graphics*, 14(1), 213-230.

Bronstein, A. M., Bronstein, M. M. & Kimmel, R. (2006). Generalized Multidimensional Scaling: A Framework for Isometry-invariant Partial Surface Matching. *Proceedings of the National Academy of Sciences*, 103(5), 1168–1172.

Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A. & Vandergheynst, P. (2017). Geometric Deep Learning: Going beyond Euclidean Data. *IEEE Signal Processing Magazine*, 34(4), 18–42.

Cao, C., Weng, Y., Zhou, S., Tong, Y. & Zhou, K. (2014). FaceWarehouse: a 3D Facial Expression Database for Visual Computing. *IEEE Transactions on Visualization and Computer Graphics*, 20(3), 413–425.

Cao, C., Bradley, D., Zhou, K. & Beeler, T. (2015). Real-Time High-fidelity Facial Performance Capture. *ACM Trans. Graph.*, 34(4), 46.

Cao, C., Chai, M., Woodford, O. & Luo, L. (2018). Stabilized Real-time Face Tracking via a Learned Dynamic Rigidity Prior. *ACM Trans. Graph.*, 37(6), 233:1–233:11.

Castrillón-Santana, M., Lorenzo-Navarro, J. & Hernández-Sosa, D. (2011). An Study on Ear Detection and Its Applications to Face Detection. *Proceedings of the 14th International Con-*

ference on Advances in Artificial Intelligence: Spanish Association for Artificial Intelligence, (CAEPIA'11), 313–322.

Chen, R. & Weber, O. (2015). Bounded Distortion Harmonic Mappings in the Plane. *ACM Trans. Graph.*, 34(4), 73.

Chi, J., Gao, S. & Zhang, C. (2017). Interactive Facial Expression Editing based on Spatio-temporal Coherency. *The Visual Computer*, 33(6), 981–991.

De Aguiar, E., Theobalt, C., Thrun, S. & Seidel, H.-P. (2008). Automatic Conversion of Mesh Animations into Skeleton-based Animations. *Computer Graphics Forum*, (2), 389–397.

Dicko, A. H., Liu, T., Gilles, B., Kavan, L., Faure, F., Palombi, O. & Cani, M.-P. (2013). Anatomy Transfer. *ACM Trans. Graph.*, 32(6), 1–8.

Dionne, O. & de Lasa, M. (2013). Geodesic Voxel Binding for Production Character Meshes. *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, (SCA '13), 173–180.

Dionne, O. & de Lasa, M. (2014). Geodesic Binding for Degenerate Character Geometry Using Sparse Voxelization. *IEEE Transactions on Visualization and Computer Graphics*, 20(10), 1367–1378.

Etöz, A. & Ercan, İ. (2012). Anthropometric Analysis of the Nose. In *Handbook of Anthropometry* (pp. 919–926). Springer.

Ezuz, D. & Ben-Chen, M. (2017). Deblurring and Denoising of Maps between Shapes. *Computer Graphics Forum*, 36(5), 165–174.

Farkas, L. G., Katic, M. J. & Forrest, C. R. (2005). International Anthropometric Study of Facial Morphology in Various Ethnic Groups/Races. *Journal of Craniofacial Surgery*, 16(4), 615–646.

Garrido, P., Zollhöfer, M., Casas, D., Valgaerts, L., Varanasi, K., Pérez, P. & Theobalt, C. (2016). Reconstruction of Personalized 3D Face Rigs from Monocular Video. *ACM Trans. Graph.*, 35(3), 28.

Gehre, A., Bronstein, M., Kobbelt, L. & Solomon, J. (2018). Interactive Curve Constrained Functional Maps. *Computer Graphics Forum*, 37, 1-12.

Ghafourzadeh, D., Rahgoshay, C., Fallahdoust, S., Beauchamp, A., Aubame, A., Popa, T. & Paquette, E. (2020a). Part-Based 3D Face Morphable Model with Anthropometric Local Control. *Proceedings of Graphics Interface 2020*, (GI 2020), 7 – 16.

Ghafourzadeh, D., Ramachandran, S., Lasa, M. D., Popa, T. & Paquette, E. (2020b). Local Editing of Cross-Surface Mappings with Iterative Least Squares Conformal Maps. *Proceedings of Graphics Interface 2020*, (GI 2020), 192 – 205.

Golla, B., Seidel, H.-P. & Chen, R. (2018). Piecewise Linear Mapping Optimization based on the Complex View. *Computer Graphics Forum*, 37(7), 233-243.

He, Y., Xiao, X. & Seah, H. S. (2009). Harmonic 1-form based Skeleton Extraction from Examples. *Graphical Models*, 71(2), 49–62.

Huang, Q.-X., Adams, B., Wicke, M. & Guibas, L. J. (2008). Non-Rigid Registration Under Isometric Deformations. *Computer Graphics Forum*, 27(5), 1449–1457.

Ichim, A.-E., Kadleček, P., Kavan, L. & Pauly, M. (2017). Phace: Physics-based Face Modeling and Animation. *ACM Trans. Graph.*, 36(4), 1–14.

Jacobson, A., Baran, I., Popovic, J. & Sorkine, O. (2011). Bounded Biharmonic Weights for Real-Time Deformation. *ACM Trans. Graph.*, 30(4), 78–1.

Jain, V., Zhang, H. & Van Kaick, O. (2007). Non-Rigid Spectral Correspondence of Triangle Meshes. *International Journal of Shape Modeling*, 13, 101–124.

Kavan, L. & Sorkine, O. (2012). Elasticity-Inspired Deformers for Character Articulation. *ACM Trans. Graph.*, 31(6), 196:1–196:8.

Kazemi, V. & Sullivan, J. (2014). One Millisecond Face Alignment with an Ensemble of Regression Trees. *IEEE Conference on Computer Vision and Pattern Recognition*.

Kim, V. G., Lipman, Y., Chen, X. & Funkhouser, T. (2010). Möbius Transformations for Global Intrinsic Symmetry Analysis. *Computer Graphics Forum*, 29(5), 1689–1700.

Kim, V. G., Lipman, Y. & Funkhouser, T. (2011). Blended Intrinsic Maps. *ACM Trans. Graph.*, 30(4), 79:1–79:12.

King, D. E. (2009). Dlib-ml: A Machine Learning Toolkit. *Journal of Machine Learning Research*, 10(Jul), 1755–1758.

Kraevoy, V. & Sheffer, A. (2004). Cross-Parameterization and Compatible Remeshing of 3D Models. *ACM Trans. Graph.*, 23(3), 861–869.

Kraevoy, V., Sheffer, A. & Gotsman, C. (2003). Matchmaker: Constructing Constrained Texture Maps. *ACM Trans. Graph.*, 22(3), 326–333.

Lacko, D., Huysmans, T., Parizel, P. M., De Bruyne, G., Verwulgen, S., Van Hulle, M. M. & Sijbers, J. (2015). Evaluation of an Anthropometric Shape Model of the Human Scalp. *Applied ergonomics*, 48, 70–85.

Le, B. H. & Deng, Z. (2012). Smooth Skinning Decomposition with Rigid Bones. *ACM Trans. Graph.*, 31(6), 1–10.

Le, B. H. & Deng, Z. (2014). Robust and Accurate Skeletal Rigging from Mesh Sequences. *ACM Trans. Graph.*, 33(4), 10.

Lee, J.-h., Hwang, S., Istook, C. L. et al. (2006). Analysis of Human Head Shapes in the United States. *International Journal of Human Ecology*, 7(1), 77–83.

Lévy, B., Petitjean, S., Ray, N. & Maillot, J. (2002). Least Squares Conformal Maps for Automatic Texture Atlas Generation. *ACM Trans. Graph.*, 21(3), 362–371.

Li, H., Sumner, R. W. & Pauly, M. (2008). Global Correspondence Optimization for Non-Rigid Registration of Depth Scans. *Computer Graphics Forum*, 27(5), 1421–1430.

Li, J. & Lu, G. (2011). Skeleton Driven Animation based on Implicit Skinning. *Computers & Graphics*, 35(5), 945–954.

Li, T., Bolkart, T., Black, M. J., Li, H. & Romero, J. (2017). Learning a Model of Facial Shape and Expression from 4D Scans. *ACM Trans. Graph.*, 36(6), 194.

Lipman, Y. (2012). Bounded Distortion Mapping Spaces for Triangular Meshes. *ACM Trans. Graph.*, 31(4), 108:1–108:13.

Lipman, Y. & Funkhouser, T. (2009). Möbius Voting for Surface Correspondence. *ACM Trans. Graph.*, 28(3), 72:1–72:12.

Liu, R., Zhang, H., Shamir, A. & Cohen-Or, D. (2009). A Part-aware Surface Metric for Shape Analysis. *Computer Graphics Forum*, 28(2), 397–406.

Liu, T., Kim, V. G. & Funkhouser, T. (2012). Finding Surface Correspondences Using Symmetry Axis Curves. *Computer Graphics Forum*, 31(5), 1607–1616.

Lüthi, M., Gerig, T., Jud, C. & Vetter, T. (2018). Gaussian Process Morphable Models. *IEEE Transactions on pattern analysis and machine intelligence*, 40(8), 1860–1873.

Miller, C., Arikan, O. & Fussell, D. S. (2011). Frankenrigs: Building Character Rigs From Multiple Sources. *IEEE Transactions on Visualization and Computer Graphics*, 17(8), 1060–1070.

Mocanu, B. & Zaharia, T. (2012). A Complete Framework for 3D Mesh Morphing. *Proceedings of the 11th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and Its Applications in Industry*, (VRCAI '12), 161–170.

Molla, E., Debarba, H. G. & Boulic, R. (2017). Egocentric Mapping of Body Surface Constraints. *IEEE Transactions on Visualization and Computer Graphics*, 24(7), 2089–2102.

Moutafidou, A. & Fudos, I. (2019). A Mesh Correspondence Approach for Efficient Animation Transfer. *Computer Graphics and Visual Computing (CGVC)*.

Neumann, T., Varanasi, K., Wenger, S., Wacker, M., Magnor, M. & Theobalt, C. (2013). Sparse Localized Deformation Components. *ACM Trans. Graph.*, 32(6), 179:1–179:10.

Ng, A. Y., Jordan, M. I. & Weiss, Y. (2002). On Spectral Clustering: Analysis and an Algorithm. *Advances in Neural Information Processing Systems*, pp. 849–856.

Nguyen, A., Ben-Chen, M., Welnicka, K., Ye, Y. & Guibas, L. (2011). An Optimization Approach to Improving Collections of Shape Maps. *Computer Graphics Forum*, 30(5), 1481–1491.

Nogneng, D. & Ovsjanikov, M. (2017). Informative Descriptor Preservation via Commutativity for Shape Matching. *Computer Graphics Forum*, 36(2), 259–267.

Ovsjanikov, M., Ben-Chen, M., Solomon, J., Butscher, A. & Guibas, L. (2012). Functional Maps: A Flexible Representation of Maps Between Shapes. *ACM Trans. Graph.*, 31(4), 30:1–30:11.

Ovsjanikov, M., Corman, E., Bronstein, M., Rodolà, E., Ben-Chen, M., Guibas, L., Chazal, F. & Bronstein, A. (2017). Computing and Processing Correspondences with Functional Maps. *ACM SIGGRAPH 2017 Courses*, pp. 9.

Panozzo, D., Baran, I., Diamanti, O. & Sorkine-Hornung, O. (2013). Weighted Averages on Surfaces. *ACM Trans. Graph.*, 32(4), 60:1–60:12.

Paysan, P., Knothe, R., Amberg, B., Romdhani, S. & Vetter, T. (2009). A 3D Face Model for Pose and Illumination Invariant Face Recognition. *Proceedings of the 2009 Sixth IEEE International Conference on Advanced Video and Signal Based Surveillance*, (AVSS '09), 296–301.

Pflug, A. & Busch, C. (2012). Ear biometrics: a survey of detection, feature extraction and recognition methods. *IET biometrics*, 1(2), 114–129.

Poirier, M. & Paquette, E. (2009). Rig Retargeting for 3D Animation. *Proceedings of Graphics Interface 2009*, (GI 2009), 103-110.

Praun, E. & Hoppe, H. (2003). Spherical Parametrization and Remeshing. *ACM Trans. Graph.*, 22(3), 340–349.

Rabinovich, M., Poranne, R., Panozzo, D. & Sorkine-Hornung, O. (2017). Scalable Locally Injective Mappings. *ACM Trans. Graph.*, 36, 16:1–16:16.

Ramachandran, S., Ghafourzadeh, D., de Lasa, M., Popa, T. & Paquette, E. (2018). Joint Planar Parameterization of Segmented Parts and Cage Deformation for Dense Correspondence. *Computers & Graphics*, 74, 202 - 212.

Ren, J., Poulenard, A., Wonka, P. & Ovsjanikov, M. (2018). Continuous and Orientation-Preserving Correspondences via Functional Maps. *ACM Trans. Graph.*, 37(6), 1–16.

Sahillioğlu, Y. (2020). Recent Advances in Shape Correspondence. *The Visual Computer*, 36(8), 1705–1721.

Sander, P. V., Snyder, J., Gortler, S. J. & Hoppe, H. (2001). Texture Mapping Progressive Meshes. *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, pp. 409–416.

Schendel, S. A. (1995). Anthropometry of the Head and Face. *Plastic and Reconstructive Surgery*, 96(2), 480.

Schüller, C., Kavan, L., Panozzo, D. & Sorkine-Hornung, O. (2013). Locally Injective Mappings. *Computer Graphics Forum*, 32, 125–135.

Seo, J., Seol, Y., Wi, D., Kim, Y. & Noh, J. (2010). Rigging Transfer. *Computer Animation and Virtual Worlds*, 21(3-4), 375–386.

Sheffer, A., Lévy, B., Mogilnitsky, M. & Bogomyakov, A. (2005). ABF++: Fast and Robust Angle Based Flattening. *ACM Trans. Graph.*, 24(2), 311–330.

Sheffer, A., Praun, E. & Rose, K. (2007). Mesh Parameterization Methods and Their Applications. *Foundations and Trends in Computer Graphics and Vision*, 2(2), 105-171.

Shi, F., Wu, H.-T., Tong, X. & Chai, J. (2014). Automatic Acquisition of High-fidelity Facial Performances Using Monocular Videos. *ACM Trans. Graph.*, 33(6), 222.

Smith, J. & Schaefer, S. (2015). Bijective Parameterization with Free Boundaries. *ACM Trans. Graph.*, 34(4), 70.

Sorkine, O. & Alexa, M. (2007). As-Rigid-As-Possible Surface Modeling. *Symposium on Geometry processing*, 4, 109–116.

Sorkine, O., Cohen-Or, D., Lipman, Y., Alexa, M., Rössl, C. & Seidel, H.-P. (2004). Laplacian Surface Editing. *Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry processing*, pp. 175–184.

Streuber, S., Quiros-Ramirez, M. A., Hill, M. Q., Hahn, C. A., Zuffi, S., O'Toole, A. & Black, M. J. (2016). Body Talk: Crowdshaping Realistic 3D Avatars with Words. *ACM Trans. Graph.*, 35(4), 54.

Sumner, R. W. & Popović, J. (2004). Deformation Transfer for Triangle Meshes. *ACM Trans. Graph.*, 23(3), 399–405.

Tena, J. R., De la Torre, F. & Matthews, I. (2011). Interactive Region-Based Linear 3D Face Models. *ACM Trans. Graph.*, 30(4), 1–10.

Van Kaick, O., Zhang, H., Hamarneh, G. & Cohen-Or, D. (2011). A Survey on Shape Correspondence. *Computer Graphics Forum*, 30(6), 1681–1707.

Veltkamp, R. & ter Haar, F. (2007). SHREC 2007-Shape Retrieval Contest.

Vestner, M., Litman, R., Rodolà, E., Bronstein, A. M. & Cremers, D. (2017). Product Manifold Filter: Non-rigid Shape Correspondence via Kernel Density Estimation in the Product Space. *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6681-6690.

Wang, X. C. & Phillips, C. (2002). Multi-Weight Enveloping: Least-Squares Approximation Techniques for Skin Animation. *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, (SCA '02), 129–138.

Wareham, R. & Lasenby, J. (2008). Bone Glow: An Improved Method for the Assignment of Weights for Mesh Deformation. In *Articulated Motion and Deformable Objects* (pp. 63–71). Springer.

Wu, C., Bradley, D., Gross, M. & Beeler, T. (2016). An Anatomically-constrained Local Deformation Model for Monocular Face Capture. *ACM Trans. Graph.*, 35(4), 115:1–115:12.

Xu, Z., Zhou, Y., Kalogerakis, E., Landreth, C. & Singh, K. (2020). RigNet: Neural Rigging for Articulated Characters. *ACM Trans. Graph.*, 39(4), 58–1.

Zell, E. & Botsch, M. (2013). ElastiFace: Matching and Blending Textured Faces. *Proceedings of the Symposium on Non-Photorealistic Animation and Rendering*, (NPAR '13), 15–24.

Zhuang, Z., Landsittel, D., Benson, S., Roberge, R. & Shaffer, R. (2010). Facial Anthropometric Differences among Gender, Ethnicity, and Age Groups. *Annals of occupational hygiene*, 54(4), 391–402.