Dynamic and Groupwise Statistical Analysis of 3D Faces

Thesis for obtaining the title of
Doctor of Engineering (Dr.-Ing.)
of the Faculty of Natural Sciences and Technology I
of Saarland University

by

Timo Bolkart

Saarbrücken,
June 2016

Datum des Kolloquiums - Date of Defence

14.06.2016

Dekan - Dean

Prof. Dr. Frank-Olaf Schreyer

Prüfungsausschuss - Examination Board

Prof. Dr. Joachim Weickert
Saarland University, Germany
(Vorsitzender - Chairman)

Dr. Edmond Boyer
INRIA Grenoble Rhône-Alpes, France
(Gutachter - Reviewer)

Prof. Dr. Hans-Peter Seidel
Max-Planck-Institute for Informatics, Germany
(Gutachter - Reviewer)

Dr. Stefanie Wuhrer
INRIA Grenoble Rhône-Alpes, France
(Betreuer - Thesis Supervisor)

Dr. Christian Richardt
Intel Visual Computing Institute, Germany
(Protokollant - Reporter)

Abstract

This thesis proposes several methods to statistically analyze static and dynamic 3D face data. First, we present a fully-automatic method to robustly register entire facial motion sequences. The representation of the 3D facial motion sequences obtained by the registration allows us to perform statistical analysis of 3D face shapes in motion. We then introduce a new localized multilinear model that is able to capture fine-scale details while being robust to noise and partial occlusions. To obtain a suitable registration for multilinearly distributed data, we introduce a groupwise correspondence optimization method that jointly optimizes a multilinear model and the registration of the 3D scans used for training. To robustly learn a multilinear model from 3D face databases with missing data, corrupt data, wrong semantic correspondence, and inaccurate vertex correspondence, we propose a robust model learning framework that jointly learns a multilinear model and fixes the data. Finally, we present one application of our registration methods, namely to obtain a sizing system that incorporates the shape of an identity along with its motion. We introduce a general framework to generate a sizing system for dynamic 3D motion data.

Zusammenfassung

Diese Dissertation stellt mehrere Methoden zur statistischen Analyse statischer und dynamischer 3D Gesichtsdaten vor. Zuerst präsentieren wir eine vollautomatische Methode zur Registrierung kompletter Bewegungsabläufe von Gesichtern. Die Darstellung der 3D Sequenzen durch die Registrierungsmethode ermöglicht die statistische Analyse der bewegten Gesichtsdaten. Anschließend stellen wir ein lokalisiertes multilineares Modell vor, das kleine geometrische Details rekonstruieren kann und robust gegenüber von Störungen und teilweisen Verdeckungen ist. Um eine geeignete Registrierung für multilinear verteilte Daten zu erhalten, präsentieren wir ein gruppenbasiertes Optimierungsverfahren, das gleichzeitig ein multilineares Modell lernt und die Registrierung der 3D Trainingsdaten optimiert. Um ein multilineares Modell robust von 3D Gesichtsdaten mit fehlenden Einträgen, korrupten Daten, fehlerhafter semantischer Korrespondenz und ungenauer Punktkorrespondenz zu lernen, stellen wir ein Verfahren vor, das gleichzeitig ein multilineares Modell lernt und die Daten repariert. Schlussendlich präsentieren wir eine Anwendung unserer Registrierungsmethoden. Wir stellen ein generelles Verfahren vor, um Standard-Größenreihen für dynamische 3D Gesichter zu berechnen, das sowohl die Gestalt als auch die Dynamik der Gesichtsdaten berücksichtigt.

Acknowledgment

First, I want to thank my supervisor Stefanie Wuhrer who gave me the opportunity to do my PhD in a nice research environment. She always had great ideas and never lost patience reading plenty of paper and thesis drafts.

I want to thank Ingmar Steiner for giving me the opportunity to attend the group meetings of the Multimodal Speech Processing group. Special thanks also to Alan Brunton, Augusto Salazar, Prosenjit Bose, Chang Shu, Joachim Weickert, Daniel Pohl, Arnur Nigmetov, and Alexander Hewer for many interesting discussions and collaborations, of which some became part of this dissertation. I also want to thank the members of my thesis committee for helpful discussions.

I thank the Cluster of Excellence on Multimodal Computing and Interaction and the German Research Foundation (WU 786/1-1) for funding and supporting the research that ultimately led to this thesis. I thank the cluster office for handling all travel-related issues. Further, I want to thank the members of the Computer Graphics group in the Max Planck Institute for Informatics for providing me with the opportunity to present and discuss my work in several CG-Lunch talks.

Finally, I want to thank my family and friends for their unlimited support.

Contents

Chapter 1

Introduction

"The face is more honest than the mouth will ever be."

– Daphne Orebaugh

1.1 Motivation

The human face has a major impact on our daily life as it plays an essential role in all kinds of social interactions. For instance, facial expressions reveal much about our feelings and thoughts [47, Chapter 1]. This motivates many different fields, including human-computer interaction, entertainment, medicine, ergonomic design, and security, to investigate the human face. Faces are used to control virtual avatars (e.g. [133]), to generate realistic physical deformable face models (e.g. [14]), to plan surgeries (e.g. [62]), to recognize certain diseases (e.g. [59]), to design best fitting gear (e.g. [134]) or to recognize faces (e.g. [102]). Depending on the application this requires a model that precisely describes the facial variations and achieves a high level of realism.

The facial shape is highly variable as it is affected by e.g. ethnicity, sex, age or facial expression. Overall, the human face can perform more than ten thousand distinct facial expressions [47, Chapter 1]. To get a low-dimensional description of the facial expressions, the Facial Action Coding System (FACS) [48] encodes the expressions as combinations of 44 action units. While these action units give a potential basis to describe facial expressions, modeling the human face by hand is rather difficult. Despite the complex variations allowed by the human face, humans are rather sensitive to recognizing unnatural face shapes, especially if the face is moving. The negative emotional response of humans invoked by small imperfections in simulated faces of humanoid robots or facial renderings is often referred to as uncanny valley [101]. To avoid the uncanny valley, data-driven methods learn a high-quality deformable face model from training data (e.g. [16, 132]). The goal of this thesis is to statistically analyze static and dynamic 3D face data.

Deforming faces are often described by statistical models that decouple the influence of identity and expression variations (e.g. [132]). This separation allows identity or expression to be altered independently, which in turn allows for a compact description of facial dynamics (e.g. [135]). Obtaining a high-quality statistical face model is challenging, since it must capture geometric details present in the data while compactly describing the data variations. To get a descriptive model, large training databases that representatively sample the population of human faces are essential.

Several new methods (e.g. [10]) and commercial systems (e.g. [71]) have been developed in recent last years to acquire static or dynamic 3D faces. With improved ability of capture 3D scans, the number of publicly available 3D face databases has increased (e.g. [139, 116, 138, 40]). These databases aim at capturing a wide variety of facial shapes and expressions, including facial dynamics.

Computing statistics on these databases requires all shapes to be in correspondence [43, Chapter 1]. Computing these correspondences for human face data is a challenging task due to the high variability of the face shape and the large differences in the data quality. Depending on the system used for the data acquisition, the face scans contain noise, holes, or partial occlusions. A suitable registration method is hence required to capture fine-scale facial details while being robust to various kinds of data corruption. For dynamic data the registration method further needs to be robust to fast motions, and the established correspondence must be temporally coherent.

The core idea in this thesis is to leverage redundancy in the data for shape processing. This is done in a groupwise fashion by jointly processing large databases in the case of static data (Chapters 6 and 7), and by processing entire motion sequences in the case of dynamic data (Chapters 4 and 5).

1.2 Thesis outline

This thesis addresses various challenges that arise when static and dynamic 3D face data are statistically analyzed. The organization of the thesis is as follows. Chapter 2 gives an overview about existing related literature. Chapter 3 introduces basic concepts about linear and multilinear face models, the wavelet decomposition of 3D surfaces, and the combination of wavelet decomposition and linear face models.

The statistical analysis of dynamic 3D face data requires all motion sequences to be in full vertex correspondence. Chapter 4 describes a fully-automatic approach to register and statistically analyze facial motion sequences using a multilinear face model as statistical prior.

While existing multilinear face models represent the global face shape well, they are unable to capture fine-scale details. To statistically model the human face including more fine-scale details, Chapter 5 introduces multilinear wavelets, a novel localized multilinear face model that makes it possible to model more fine-scale details while retaining robustness to noise and partial occlusions.

To compute a high-quality multilinear face model, the quality of the registration of the database of 3D face scans used for training is essential. Meanwhile, a multilinear face model can be used as an effective prior to register 3D face scans, which are typically noisy and incomplete. Inspired by the minimum description length approach, Chapter 6 proposes the first method to jointly optimize a multilinear model and the registration of the 3D scans used for training. While most existing methods assume the object to be a closed manifold, our approach handles manifolds with multiple boundaries.

Existing methods to learn a multilinear face model degrade if not every person is captured in every expression, if face scans are noisy or partially occluded, if expressions are erroneously labeled, or if the vertex correspondence is inaccurate. To overcome these limitations, Chapter 7 introduces the first framework to robustly learn a multilinear model from 3D face databases with missing data, corrupt data, wrong semantic correspondence, and inaccurate vertex correspondence.

Once 3D facial motion sequences are successfully registered, they can be used in various applications. Besides the applications of dynamic 3D motion data for the synthesis of new motion sequences and the recognition of dynamic expressions discussed in Chapter 4, Chapter 8 introduces a general framework to generate a sizing system for dynamic 3D motion data for the design of face masks that incorporates the shape of an identity along with its motion.

Finally, Chapter 9 concludes the thesis, summarizes key advantages of our methods, and gives an outlook on open problems and future work.

1.3 Contributions

The novel contributions described in this thesis have either been published [19, 21, 28, 22, 20] or have been accepted for publication [24]. Where not explicitly stated otherwise, I am the main contributor to the work. Our novel contributions described in this dissertation are as follows.

Statistical motion analysis (Chapter 4): Parts of this work were first published in 3DV 2013 [19], and an extended version has been published in CVIU [21]. Our main contributions are:

- a new Markov random field (MRF)-based landmark prediction method for entire motion sequences of 3D faces,
- a fully-automatic approach to register motion sequences of 3D faces both spatially and temporally using a multilinear model as statistical prior that is robust with respect to fast motions,
- a general framework to statistically analyze 3D face shapes in motion, and
- four applications for our framework; namely, we propose different ways to synthesize new motion sequences and recognize dynamic expressions.

Multilinear wavelets (Chapter 5): This work has been published in ECCV 2014 [28]. While I was not the main author, the main contributions were achieved in close collaboration with Alan Brunton. My responsibilities comprised the implementation of the multilinear model and the bi-Laplacian smoothing. Further, I contributed to the model integration, testing, evaluation, and writing the paper. Our main contributions are:

- a statistical shape space based on a wavelet decomposition of 3D face geometry and multilinear analysis of the individual wavelet coefficients,
- an efficient algorithm for learning a statistical shape model of the human face in varying expressions, and
- an efficient algorithm for fitting our model to static and dynamic point cloud data that is robust with respect to highly corrupted scans.

Registration optimization (Chapter 6): This work has been published in ICCV 2015 [22]. Our main contributions are:

- a fully automatic groupwise correspondence optimization approach for multilinearly distributed data, and
- an approach that is computationally significantly more efficient and leads to correspondences of higher quality than existing PCA-based optimization methods.

Robust multilinear model learning (Chapter 7): This work has been accepted for publication in CVPR 2016 [24]. Our main contributions are:

- a data completion technique with similar performance as state-of-the-art tensor completion methods,
- a data reconstruction technique for corrupt data that outperforms the state-of-the-art, and
- a re-labeling technique to improve semantic correspondence.

Motion sizing system (Chapter 8): This work has been published in 3DV 2014 [20]. Our main contributions are:

- a general framework to generate a sizing system for dynamic 3D motion data,
- the generation of a representative 3D model for each size for fabrication, and
- the application of our framework to generate a specific sizing system for facial motion data for face mask design.

Chapter 2

Literature review

> *"Never memorize something that you can look up in books."*
>
> – Albert Einstein

This chapter gives an overview about literature related to statistical analysis of static and dynamic 3D face data. While statistical methods and correspondence computations are applied to various kinds of 2D and 3D objects, we focus the literature review on statistical methods, their applications, and correspondence computation on 3D faces. This allows us to give a more comprehensive overview. For a more general overview on statistical models see e.g. our survey [29], and for a more general view on correspondence computation see e.g. the surveys by Tam et al. [125] or van Kaick et al. [131]. While previous methods on groupwise correspondence optimization methods operate on 1D curves or 2D surfaces, we focus the review of related work on surface-based methods. For a more general overview see e.g. the book by Davies et al. [43].

Shape: In this thesis we define the shape of an object as the geometrical information that remains if the effects of rotation, translation and scale are removed, as proposed by Kendall [45, Chapter 1].

Statistical shape model: In this thesis we use statistical shape models to describe the space of variations of a class of shapes. This space—the shape space—is defined by a low-dimensional basis with an additional statistical prior. This statistical prior is a probability distribution that measures, for a shape, the likelihood that it is a valid instance of the given class of shapes. For statistical face models this prior is often a multivariate Gaussian distribution.

2.1 Data acquisition

The goal of this thesis is to statistically analyze static and dynamic 3D face data, which requires suitable 3D face databases. Several methods and systems exist to digitally capture the 3D surface of human faces, ranging from expensive commercial systems of high quality to cheap consumer depth cameras of low quality. One key goal of our methods is to work for data from various sources.

This section gives a brief overview about existing techniques to capture 3D surfaces of the human face. Existing methods can be grouped into active and passive capture methods. In the following, we only discuss methods that are either used to acquire the databases used throughout this thesis, namely BU-3DFE [139], BU-4DFE [138] or the Bosphorus database [116], or systems used in low-cost consumer products like the Microsoft Kinect [78] or Intel RealSense [113]. For a more detailed overview see e.g. the thesis by Scherbaum [117, Chapter 2.1].

Active methods: Active systems consist of an emitting unit and an imaging sensor. The emitting unit projects a signal in the form of light or radiation onto an object and the imaging sensor measures the signal reflected by the surface of the object. The calibration of the emitting unit and the imaging sensor then makes it possible to determine the 3D position of a surface point.

Structured light scanners are one type of active scanner system. These consist of a light projector and an imaging sensor. They project a light pattern on the object and measure the geometrical deformation of the pattern caused by the surface of the object. The calibration of projector and sensor allow depth information to be computed by triangulation. For more details see the overview by Besl [13].

The Bosphorus database and the BU-3DFE database are captured with commercial structured light scanner systems. The Bosphorus database is captured with the Inspeck Mega Capture II; the BU-3DFE database is captured with the 3DMD digitizer [71]. The 3DMD digitizer projects random light patterns onto the surface of the object. Six digital cameras, three on each side of the face, then capture the images used for reconstruction. The first generations of Microsoft Kinect and Intel RealSense use a structured light scanner that consists of an infrared projector and an infrared sensor. The Microsoft Kinect uses a speckled infrared dot pattern; Intel RealSense uses an infrared grid as its light pattern.

Further active methods include *time-of-flight* (ToF) scanners. ToF scanners consist of an infrared projector and an infrared sensor. They emit a light signal and measure the duration until the signal returns to the sensor. Due to the known constant speed of light ($c = 299792458\frac{m}{s}$) the distance between the camera and the surface of the object can be computed once the elapsed time is known. Instead of directly measuring the elapsed time, current ToF systems use indirect measurements. Most existing methods emit a light signal with modulated intensity. The distance from the surface of the object to the camera is then computed from the phase shift of sent signal and received signal. For more details see Lefloch et al. [87]. The second-generation Microsoft Kinect uses a ToF sensor, for instance.

Passive methods: In contrast to active systems, passive systems use the existing lighting of the object without altering the appearance of the object by emitting light or radiation.

Stereoscopic methods are one widely-used type of passive 3D surface capturing system. Stereoscopic systems usually consist of two horizontally slightly displaced imaging sensors. The slight displacement results in two images of the same object from different perspectives. The displacement of the x-coordinate of corresponding points caused by the different perspective is often referred to as disparity. The disparity along with the known distance of the sensors allow the depth of a point to be computed by triangulation. The principle of stereoscopic systems is similar to the binocular visual system of humans. For more details see Moons et al. [100].

The BU-4DFE is captured with the Di3D [72] from Dimensional Imaging. The Di3D is a commercial scanner system with two stereo cameras and one texture camera. Both stereo cameras produce depth maps using a passive stereo reconstruction. These depth maps are then combined to obtain a full 3D face scan.

2.2 Statistical face models

As the goal of this thesis is the statistical analysis of static and dynamic 3D faces, it relates to previous methods that perform statistical analysis on facial surfaces. This section introduces existing statistical 3D face models.

Databases of shapes are often high-dimensional while the possible variations of the shapes only describe a low-dimensional manifold. While it is difficult to model this low-dimensional description manually, data-driven statistical shape models can be used to describe a low-dimensional shape space. This low-dimensional shape description reduces the search space for various applications and hence allows various underconstrained problems to be solved (see Section 2.5).

Global models: Given a set of 3D shapes in full correspondence, various methods exist to statistically analyze the shapes. Blanz and Vetter [16] propose the first statistical 3D face model, called a morphable model, that uses principal component analysis (PCA) (see Section 3.2) to analyze 3D shape and texture of registered 3D faces, mainly with neutral expressions. Patel and Smith [110] show simplifications for the morphable model by introducing a multi-resolution fitting. While the morphable model is mainly used to analyze the shape variations of 3D faces of different identities, other works also analyze shape variations caused by different expressions. Yang et al. [136] build several PCA models, one for each expression. Amberg et al. [4] use another statistical model that combines a PCA model for shape and texture of a neutral expression with a PCA model for the expression difference vectors from the neutral expression. The linear separation of identity and expression of this model assumes identity and expression deformations to be independent which enables the direct transfer of expression differences between faces. Tensor-based frameworks instead model the dependency between identity and expression deformations by non-linear projections. Vlasic et al. [132] use a tensor-based method to model 3D faces, named a multilinear model (Section 3.4), that is a higher-order generalization of the PCA model.

Local models: Part-based models are frequently used to increase the data variability captured by the morphable model. Instead of using one global model on the entire shape, the shape is segmented into disjoint parts and separate models per part are computed. Learning models, independently per part, decorrelates the different segments. Due to the decorrelation, these multiple part-based models are more expressive than global models. Due to the better generalization capability, part-based models require less training data than global models do to capture the same data variability.

Blanz and Vetter use a part-based model by manually segmenting the face shape into disjoint segments and learning morphable models on each segment independently. Smet and Van Gool [121] use an automatic segmentation of the face shape to learn a part-based model. Instead of using a global statistical model or part-based statistical models, Brunton et al. [30] learn a localized wavelet model. For this, training faces are transformed into wavelet space, and PCA is performed on the resulting localized wavelet coefficients. For more technical details see Section 3.3. This localized approach preserves local details in the context of model fitting. Another localized method proposed by Neumann et al. [103] takes a facial motion sequence and decomposes the global deformation into localized components using sparse PCA. Ferrari et al. [51] learn a sparse linear basis from a 3D face database of different identities performing multiple expressions. Golovinskiy et al. [57] propose a method based on hierarchical pyramids to reconstruct small facial details.

2.3 Correspondence computations

The statistical analysis of static and dynamic 3D faces requires all shapes to be in full correspondence. This section summarizes previous methods that compute correspondences between 3D face shapes.

Dense point-to-point correspondences between shapes are used e.g. to compute statistics [43, Chapter 1], to morph between shapes [27, Chapter 12], or to transfer textures between shapes [27, Chapter 12]. Tam et al. [125] and van Kaick et al. [131] give an overview of registration techniques for different classes of objects. While it is difficult to register shapes without prior knowledge of the class of objects, we restrict our literature overview to methods that are specifically designed for 3D surfaces of faces. The restriction to 3D faces reduces the search space for the correspondence computation, as it makes it possible to leverage prior knowledge of the face shape and possible deformations.

2.3.1 Sparse face correspondence

Several methods exist to automatically establish sparse correspondence between static face shapes by predicting facial landmarks. Computing a sparse correspondence results in an easier problem compared to computing a full per-vertex correspondence, as salient keypoints—called landmarks—around eyes, nose, and mouth can be leveraged that are easier to detect automatically. While computing facial landmarks has been studied extensively in 2D images (e.g. [38]), we focus on computing landmarks in 3D face scans.

These automatic landmark detection methods learn global or local geometric properties of the landmarks and use these information to infer the same corresponding landmarks on new scans. Guo et al. [58] predict landmarks using a PCA based method learned on a set of salient points together with a geometric and texture-based heuristic. Passalis et al. [108] select possible landmarks using shape index and spin image and validate the possible landmarks using a learned PCA space of facial landmarks. Berretti et al. [11] use curvature together with a scale-invariant feature transform (SIFT) descriptor to predict facial landmarks. Creusot et al. [41] learn the statistical distribution of several descriptors on known landmarks and their optimal combination. In contrast to this method, Salazar et al. [114] learn the statistical distribution of one descriptor on known landmarks and train a MRF to model connections between these landmarks. For an input scan, Salazar et al. predict the landmarks using belief propagation. Gilani et al. [55] use a PCA model fitting to transfer landmarks from the template face to unseen face scans. The survey by Çeliktutan et al. [38] gives further details on facial landmarking.

2.3.2 Dense face correspondence

Computing dense 3D face correspondence is challenging as the facial shape is highly variable and for large facial regions (except eyes, nose, and mouth) the local geometry and texture is less distinctive which impedes dense matching solely based on local facial features. Instead many previous methods use face templates in addition to facial landmarks to reduce the search space for dense registration. To register a 3D face scan, these template-fitting approaches use an initial face shape—the face template—that is deformed locally to closely match the scan. The deformed template is then used as a registration of the scan. All scans that are registered by deforming the same face template are implicitly in full dense per-vertex correspondence. To be more robust to noisy scans or large expression deformations, the possible deformations of the face template are often restricted to match geometric constraints such as minimizing surface bending (e.g. [109]) or by parametric models such as blendshape models (see Section 3.2) or statistical models (see Sections 3.2 and 3.4).

Single expression registration: To register static 3D faces in a single expression, Blanz and Vetter [16] use an optical flow algorithm to match vertices with similar color. They improve the correspondence by bootstrapping; they iteratively learn a model, fit the scans with the model, and update the registration using optical flow. Amberg et al. [6] use a non-rigid iterative closest point (ICP) method to fit a template to the input scan. Passalis et al. [108] fit an annotated face model (AFM) [75] to an input scan. The AFM is an average 3D face from statistical data, segmented into different annotated areas. The deformation of the AFM to fit the scan is done by solving a second-order differential equation with a finite element method. To be robust to missing data they explore the facial symmetry during AFM fitting.

Several methods exist that use a thin-plate spline (TPS) deformation to fit a template to a scan [109, 68, 111, 58, 94]. Given two sets of points, thin-plate splines define the interpolative mapping between both point sets with minimum bending energy [45, Chapter 10]. To register face scans, Patel and Smith [109] compute an interpolative TPS mapping between landmarks of a scan and landmarks in the mean shape, and resample the TPS mapping consistently to get a dense correspondence between the template and the scan. Hu et al. [68] and Qin et al. [111] sample up to 600 points from a template scan and establish an initial correspondence to another scan using ICP. To refine the correspondence, Hu et al. iteratively establish an approximate TPS mapping using the correspondence, deform the template scan according the TPS mapping, and update the correspondence. Qin et al. deform the template scan according the TPS mapping, perform PCA over all TPS based registrations, and use a bootstrapping that iteratively fits the model based on the updated correspondence. Guo et al. [58] and Liu et al. [94] establish a TPS mapping between a face template and the scan using a sparse correspondence, warp the template according the TPS mapping, and establish dense correspondence by projecting the points of the deformed template into the surface of the scan.

Further statistical model based methods exist. Blanz et al. [15] learn a morphable face model and use this model as statistical prior to reconstruct 3D face scans; Brunton et al. [30] learn a linear wavelet face model (Section 3.3) and use this model to reconstruct 3D face scans.

Multiple expression registration: To register static 3D faces in multiple expressions, Mpiperis et al. [102] fit a face template to an input scan using an elastically deformable model. This elastically deformable model consists of face template modeled as a subdivision surface that is deformed based on a non-rigid ICP method to fit the scan. Salazar et al. [114] use a blendshape model to fit the expression of a given input scan, and a template deformation based on a non-rigid ICP method to fit its shape. Ferrari et al. [51] partition each input scan into 28 regions bounded by geodesic paths between facial landmarks and resample the regions consistently to obtain a dense correspondence. Bronstein et al. [27, Chapter 12] establish dense correspondence between faces using generalized multi-dimensional scaling (GMDS) [27, Chapter 9]. GMDS embeds the intrinsic geometry of one shape into another and measures the distance between the shapes in the embedding space defined by the second shape.

Dynamic registration: To register dynamic 3D faces in varying expressions, Fang et al. [50] consecutively fit an AFM to a facial motion sequence where the AFM for each frame is initialized by the result of the previous frame. Huang et al. [69] decompose a face into parts and use displacement mapping, where vertices move along their normal directions combined with point-to-surface mappings to fit the individual face parts to an input face. This is followed by a blending of the separate parts. Breidt et al. [97] use a morphable face model that consists of two PCA models, one for identity and one for expression. They fit the identity PCA to the first frame of a sequence, and for each further frame they only fit the expression PCA initialized by the result of the previous frame while the identity is fixed.

Several methods exist that register facial RGB-D sequences in real time [90, 133, 25, 91, 127, 66, 89]. These methods use linear blendshape bases to model the expression deformations. Li et al. [90], Weise et al. [133], and Bouaziz et al. [25] compute person-specific blendshapes by registering specific example expression scans. For some example expressions, a generic blendshape basis, and blendshape weights that approximately resemble the specific expressions, they optimize for the personalized blendshape basis that best fits the example expressions. Given the personalized basis, registering the RGB-D sequences is performed by optimizing for the blendshape coefficients that best fit the input data. Bouaziz et al. further use a PCA model for identity to robustly fit the identity of a neutral scan. Li et al. [91, 89] and Hsieh et al. [66] perform deformation transfer [123] to a neutral person-specific scan to compute person-specific blendshapes. They learn an adaptive linear PCA model from the person-specific blendshapes to register the RGB-D sequences. To increase the expressiveness of the PCA model during registration, Li et al. [91] and Hsieh et al. add further expressive training shapes to the PCA model. Thies et al. [127] use linear PCA models for identity and albedo, and a linear blendshape model for expression. For sequences registration they estimate the identity on a short sequence and keep the identity fixed afterwards while optimizing for the blendshape coefficients only.

2.4 Groupwise correspondence optimization

Since we leverage redundancy in the data in a groupwise fashion to jointly optimize multilinear correspondence (Chapter 6) and to robustly learn a multilinear face model (Chapter 7), our methods are related to groupwise correspondence optimization methods. This section introduces methods that jointly optimize the registration of a set of 3D shapes and a learned statistical model in a groupwise manner.

Computing these correspondences for human face data is a challenging task that many methods aim to solve (see Section 2.3). Given a good registration, a statistical face model can be learned that can be used to reconstruct the 3D geometry from noisy or partially occluded face scans as discussed in Section 2.2. Such a model is directly applicable for registration as discussed in Section 2.3. Summing up, this is a chicken-and-egg problem: given a good registration, a statistical model can be learned, and given a representative statistical model, a good registration can be computed. This motivates formulating the statistical face model learning as an optimization framework that aims to learn a statistical face model while at the same time optimizing the correspondence of the training data. These optimization frameworks measure the model quality and change the registration such that the quality of the model and the registration improve at the same time. Since the model quality depends on all shapes, these model-based methods to optimize correspondence are called groupwise optimization methods. In the absence of groupwise correspondence optimization methods for 3D faces, this section summarizes related methods that operate on 2D surfaces of any class of shapes.

Linear methods: Most existing groupwise optimization methods describe the data with one linear PCA model. Kotcheff and Taylor [81] propose a groupwise correspondence optimization based on the determinant of the data covariance matrix that explicitly favors compact PCA models. Thodberg [128] uses a simplified version of the information theoretic objective function minimizing the description length of the data [43, Chapter 4]. The basic concept of minimum description length (MDL) approaches is to minimize the length of a message that is transmitted from a sender to a receiver. They encode the data with a PCA model and alter the correspondence such that the number of bits needed to describe the model and the encoded data is minimal. Styner et al. [122] optimize the same simplified MDL objective with additional local curvature constraints. Davies et al. [43, Chapter 4] give a more thorough overview of different objective functions for correspondence optimization. Davies et al. [44] show that MDL outperforms state-of-the-art registration methods for medical datasets. Gollmer et al. [56] compare different objective functions. They show that while the determinant of the covariance matrix is easier to optimize, the results are comparable to results produced by MDL.

Non-linear methods: Burghard et al. [31] use a part-based groupwise linear model. They cut each shape into multiple parts and optimize the correspondence of each part by minimizing the groupwise linear objective function of Kotcheff and Taylor with an additional regularization term. Chen et al. [39] model the data with a non-linear kernel PCA. They embed each shape into a non-linear feature space induced by a non-linear kernel and optimize the groupwise objective function of Thodberg in this feature space. Hirshberg et al. [64] derive a skeleton-based approach specifically for human body shapes to jointly optimize the registration and a statistical model.

2.5 Applications of deformable face models

This section discusses some applications of deformable 3D face models, namely statistical face models and blendshape models. Altering facial shape or pose in 2D images or videos is rather difficult, as parts that were originally occluded might become visible, and lighting conditions change as e.g. shadows cast by the face shape appear or disappear. In contrast to these problems that occur for the 2D face shape, the shape and pose of 3D faces can be changed free of self-occlusions, and lighting changes can be simulated on the surface in 3D. Hence, instead of processing the facial shape directly in 2D, the 3D face shape is frequently used to process and edit faces in 2D images or 2D videos (e.g. [132]). This requires robust methods to reconstruct 3D face shape from 2D images or videos.

Reconstruction: One body of work uses deformable face models to reconstruct 3D face shape from one or more 2D images. Garrido et al. [54] use a personalized blendshape model as prior information to reconstruct 3D face shape from 2D videos. Cao et al. [33] combine a personalized blendshape model with a probability map for person-specific facial features like wrinkles to reconstruct detailed 3D facial performance from 2D videos. Patel and Smith [109] and Aldrian and Smith [1] use a morphable face model to reconstruct 3D face shape from sparse 2D markers. Aldrian and Smith use the 3D reconstruction to estimate object attributes like facial texture, lighting conditions, and camera properties. Brunton et al. [30] use a localized wavelet model (see Section 3.3) to reconstruct 3D face shapes from stereo images. Shi et al. [120] use a multilinear face model to reconstruct 3D face shape from 2D videos.

Further methods exist that use deformable face models to reconstruct 3D face shape from RGB-D images or face scans. Hsieh et al. [66] and Li et al. [89] use a personalized blendshape model as prior information to capture facial performance from RGB-D sequences. The method by Hsieh et al. is robust to various facial occlusions; the method by Li et al. uses additional input of strain sensors to be robust to occlusions caused by a head-mounted display (HMD). These strain sensors are mounted on the HMD to measure the deformations in the foam of the HMD to estimate the expressions under the HMD that are not visible to the RGB-D camera. Kazemi et al. [77] use a generic blendshape model to reconstruct the 3D face shape from RGB-D images. Breidt et al. [97] use identity and expression PCA models to register 3D motion sequences. They use the registration to analyze activated action units of motion sequences. Ferrari et al. [51] use a sparse linear face model to reconstruct the 3D face shape from 2D images and 3D face scans.

Recognition: One area of work uses 3D deformable face models to recognize faces or facial expressions. Face recognition has applications in security, e.g. to automatically recognize identities in surveillance systems. The automatic recognition of expressions can improve human-computer interaction by enabling computers to react in expression-specific ways.

To recognize faces with neutral expressions, Blanz et al. [15] and ter Haar and Veltkamp [126] use a morphable face model to reconstruct the 3D face shape from neutral-expression 3D face scans. The robustness of the morphable face model to scanner noise and lighting conditions allows robust recognition of faces under varying conditions. Blanz et al. use the representation in morphable model space to recognize faces; ter Haar and Veltkamp exploit the surface correspondence induced by the fitted model to recognize faces. Amberg et al. [4] use a combination of a morphable model for shape and a PCA model for expression difference vectors from the mean face to recognize faces with varying expressions. Mpiperis et al. [102] use multilinear face models to recognize faces and facial expressions in 3D face scans.

Animation: Another area of work uses 3D deformable face models to animate digital avatars. The animation of digital avatars has various applications in the gaming or movie areas as it allows non-human avatars to appear realistic by mimicking human facial expressions.

Li et al [90, 91], Bouaziz et al. [25], and Weise et al. [133] use a personalized blendshape model as prior information to capture facial performances from RGB-D sequences and to animate artist-modeled avatars based on the obtained blendshape weights. To obtain a personalized blendshape model, the earlier work of Li et al. [90] and the work by Weise et al. use an initial user-specific calibration. The later work of Li et al. [91] and the work by Bouaziz et al. use a PCA model for identity in neutral expression to alter the identity of the blendshape model. Cao et al. [35, 34] use a multilinear face model to generate personalized blendshapes. They use the personalized blendshape model as prior information to capture facial performance from 2D videos and to animate artist modeled avatars based on the obtained blendshape weights. Ichim et al. [70] use a personalized blendshape model to animate digital avatars from hand-held 2D video input. They use a multi-view stereo method to reconstruct a 3D head model from hand-held video input. To get a personalized blendshape model, they optimize the articulation of the head model for 2D expression recordings.

Editing: Further, many existing methods use 3D deformable face models to change face shape or appearance. Similar to the animation of avatars, face editing has potential applications in movie production e.g. as it allows actors' performance to be altered in a post-processing step.

Thies et al. [127] use linear PCA models for identity and albedo, and a linear blendshape model for expression to register RGB-D sequences to transfer expressions between subjects.

Blanz and Vetter [16] use a morphable face model to reconstruct 3D face shape from 2D images and apply the 3D face shape to alter the images e.g. by editing the pose of the face, or the lighting conditions. Scherbaum et al. [118] use a morphable face model to reconstruct face scans with and without makeup. Then they learn a mapping between facial appearance and facial makeup and automatically suggest makeup for new face scans. Yang et al. [136] learn multiple PCA spaces, one for each expression, to transfer facial parts between images. Amberg et al. [5] use a combination of a morphable model for shape and a PCA model for expression difference vectors from the mean face to alter the 3D face shape. Neumann et al. [103] use a sparse PCA model that allows editing the facial shape locally.

Vlasic et al. [132], Dale et al. [42], and Yang et al. [135] use a multilinear model to reconstruct 3D face shape from 2D motion sequences. Vlasic et al. and Dale et al. use the reconstruction to transfer expressions between images and videos; Yang et al. use the reconstruction to alter face shapes or expressions in videos.

Chapter 3

Basic definitions

> *"Trying to analyze a situation without enough data was like looking at a photograph of a ball in flight and trying to gauge its direction. Is it going up, down, sideways? Is it about to collide with a baseball bat? Is it moving at all, or is something on the blind side holding it in place? A single frame didn't mean a thing. Patterns were based on data. With enough datapoints, you could predict just about anything."*
>
> – Marcus Sakey

This chapter gives some background on data and techniques used throughout this thesis. First, we describe some 3D face databases used for training and evaluation of our techniques. Further, we give some technical details of statistical face models, namely linear face models and multilinear face models. The multilinear face models in particular are heavily used in the remainder of the thesis to model human faces in varying expressions. Further, this chapter introduces the wavelet transformation of surfaces and the combination of wavelet transformations and linear face models to obtain a localized linear face model.
Notation: The notation for the wavelet decomposition and the linear wavelet face model is similar to the notation used in our survey [29] and the notation by Brunton et al. [30]. The notation for the multilinear face models resembles the notation used by Vlasic et al. [132].

3.1 Face databases

This section describes some 3D face databases used throughout this thesis, namely the Bosphorus database [116], the BU-3DFE database [139], and the BU-4DFE database [138].
Bosporus: The Bosphorus database contains scans of 105 subjects, 45 female and 60 male, mostly Caucasians, with up to 35 expressions, 4 variations of facial occlusions, and up to 13 head poses. The expression scans consist of a neutral expression, the six prototypical expressions anger, disgust, fear, happiness, sadness, and surprise, and 28 FACS. The occlusions contain occlusions of the mouth and eye, and occlusions by glasses and hair. For each face scan 24 facial landmarks are manually selected.
BU-3DFE: The BU-3DFE database contains scans of 100 subjects, 56 female and 44 male, of different ethnicities in neutral expressions and the six prototypical expressions. Each of the

expressions occurs in four intensity levels, ranging from low intensity to high intensity. For each face scan 83 facial landmarks are manually selected.
BU-4DFE: The BU-4DFE database contains motion data of 101 subjects, 58 female and 43 male, of different ethnicities, each performing the prototypical expressions. Each motion sequence starts with a neutral expression, then goes to high intensity, and back to the neutral expression. Each motion sequence consists of about 100 frames.

3.2 Linear face model

Blendshape model: Blendshape models [88] are widely used to generate facial animations. Blendshape models are linear face models defined by a set of $m + 1$ face meshes $\mathbf{b}_i \in \mathbb{R}^{3n}$ all in full vertex correspondence. The mesh $\mathbf{b}_0$ typically represents a neutral face shape, while each of the $\mathbf{b}_i$ with $i > 0$ represent a semantically meaningful expression.

The blendshape model represents a 3D face $\mathbf{f} \in \mathbb{R}^{3n}$ by the affine transformation

$$\mathbf{f} = \mathbf{b}_0 + \mathbf{B}\mathbf{w}, \tag{3.1}$$

where the i-th column of $\mathbf{B}$ is $\mathbf{b}_i - \mathbf{b}_0$, and $\mathbf{w} \in \mathbb{R}^m$ contains the blendshape coefficients. Note that blendshape models are not statistical shape models as usually no statistical prior is used. Unlike statistical methods, each coefficient of the blendshape model corresponds to a semantically meaningful expression. Statistical models, in contrast, aim to describe the data variability with a low number of coefficients, and hence lack an intuitive mapping to semantically meaningful expressions. The mapping to semantic information causes blendshape models to be overcomplete, where different coefficients produce the same expression. Further, the combination of different expressions may produce unlikely facial expressions.
Principal component analysis: The most common linear model for statistical analysis is PCA [45, Chapter 5]. Given some data, PCA is an unsupervised method that learns a linear subspace of the data. For 3D face modeling, given a set of d registered and spatially aligned 3D face scans, each face is represented by a vector $\mathbf{x} = (x_1, y_1, z_1, \cdots, x_n, y_n, z_n)^T$ that consists of n vertices $(x_i, y_i, z_i)^T$. PCA is an orthogonal linear basis transformation from $\mathbb{R}^{3n}$ to $\mathbb{R}^m$ with $m \leq 3n$ that maximizes the variance of the projection along each axis $\mathbf{u}_a$, $a \in \{1, ..., m\}$ in the projected space. Hence, the axes $\mathbf{u}_a$ are chosen to maximize $\sum_{a=1}^{m} \sum_{i=1}^{d} \left(\left(\mathbf{x}_i - \overline{\mathbf{x}} \right) \mathbf{u}_a \right)^2$, where $\overline{\mathbf{x}} = \frac{1}{d} \sum_{i=1}^{d} \mathbf{x}_i$ denotes the mean over all training faces, and the $\mathbf{u}_a$ are constrained to be orthogonal. The axes $\mathbf{u}_a$ are the principal axes or principal components of the data. The subspace spanned by the $\mathbf{u}_a$ is called shape space or model space.

The basis of the shape space can be computed as the eigenvectors corresponding to the first m non-increasing eigenvalues λ_a of the data covariance matrix

$$\mathbf{D} = \frac{1}{d} \sum_{i=1}^{d} \left(\mathbf{x}_i - \overline{\mathbf{x}} \right) \left(\mathbf{x}_i - \overline{\mathbf{x}} \right)^T. \tag{3.2}$$

PCA reduces the dimensionality of the data if $m < 3n$. If the original data contain redundancies, the dimensionality reduction is lossless if $m = rank(\mathbf{D})$, where $rank(\mathbf{D})$ is the rank

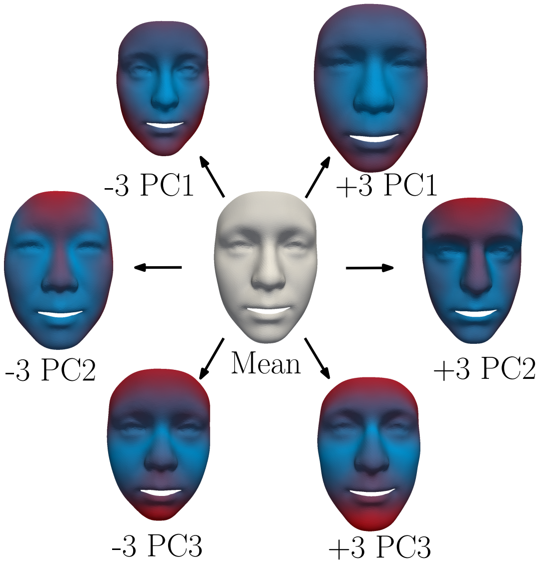


Figure 3.1: Variations of three principal components of the PCA model learned from all neutral scans of the registered BU-3DFE database. The magnitude of the vertex displacements from the mean face is color coded from blue (zero) to red (maximum).

of **D**. Note that due to the centering of the data, the rank of **D** is at most $min(3n, d-1)$. If $m < rank(\mathbf{D})$, information is lost by the dimensionality reduction. The amount of lost information can be estimated with prior knowledge about the multivariate distribution of the training face. Assuming a multivariate Gaussian distribution, each eigenvalue λ_a of **D** measures the variability of the training data captured by $\mathbf{u}_a$. The PCA model captures $100 \cdot \frac{\sum_{a=1}^{m} \lambda_a}{\sum_{a=1}^{rank(\mathbf{D})} \lambda_a} \%$ of the variability of the training data, while the rest is lost.

Assuming a multivariate Gaussian distribution, the PCA model is a generative statistical model that represents a registered 3D face $\mathbf{f} \in \mathbb{R}^{3n}$ by the affine transformation

$$\mathbf{f} = \overline{\mathbf{x}} + \mathbf{U}\mathbf{w}, \tag{3.3}$$

where the columns of the projection matrix $\mathbf{U} \in \mathbb{R}^{3n \times m}$ are the $\mathbf{u}_a$, and $\mathbf{w} \in \mathbb{R}^m$ contains the coefficients in parameter space, namely the principal components.

The PCA model represents the high-dimensional differences of each shape and the mean shape in a low-dimensional shape space. Within this shape space, each principal component potentially affects each vertex coordinate. Statistical models with this global influence are referred to as global models.

Figure 3.1 visualizes the variations of three principal components of a PCA model learned from 100 registered 3D face scans of the BU-3DFE with neutral expressions. The variation along each principal component is in the range of $\pm 3\sigma$, where σ denotes the standard deviation of the principal component.

Quality measures: The quality of linear statistical models is widely evaluated by the measures compactness, generalization, and specificity [43, Chapter 9.2]: the model should ideally be compact, general and specific.

Compactness measures the amount of variability of the training data captured by the statistical model. The compactness of the model for m components is computed by $C\left(m\right) = \frac{\sum_{a=1}^{m} \lambda_a}{\sum_{a=1}^{rank(\mathbf{D})} \lambda_a}$, where λ_a denotes the a-th eigenvalue of the data covariance matrix $\mathbf{D}$, computed as described above.

Generalization measures the ability of the statistical model to represent shapes of the same class that are not part of the training. The generalization error is measured in a leave-one-out fashion, where each shape is excluded once from training and the resulting model is used to reconstruct the excluded shape. The error is then measured as the distance between the reconstruction and the excluded shape. A high generalization error indicates that the statistical model overfits the training data.

Specificity measures the ability of the statistical model to represent only valid shapes of the object class. The model space is randomly sampled and the sample shapes are reconstructed using Equation 3.3. The specificity error is then computed as the distance of the sample shape from the closest training shape.

3.3 Linear wavelet face model

Wavelet transform: The wavelet transform is a basis transformation into a set of spaces spanned by scaled and shifted versions of a scaling function and a wavelet function. The wavelet decomposition is local in frequency and due to the local support of the scaling and wavelet functions, variations of coefficients only affect the surface locally and hence the wavelet decomposition is local in space or time [96, Chapter 2.2]. Originally the wavelet transform was defined on regularly sampled 1D signals or 2D images [98]. For 2D surfaces imbedded in 3D the wavelet transform can be computed on a subdivision representation of the surface that defines a hierarchical multi-resolution representation of the surface [96]. Subdivision techniques for polyhedral surfaces such as Loop subdivision [95], butterfly subdivision [46], or Catmull-Clark subdivision [37] recursively refine the surface according to a specific scheme by inserting new vertices and possibly moving existing ones.

The wavelet transform decomposes a surface into multiple levels of scale, with low-frequency parts represented by scaling coefficients, and higher-frequency parts represented by wavelet coefficients. This makes it possible to denoise [67] and compress [12] geometry by using only coefficients up to a certain scale, while the coefficients of higher scales are discarded. The multiresolution nature of wavelets makes it possible to define the basis functions of a certain level by a finite linear combination of the basis functions of a higher (finer) level. This allows the wavelet transform for biorthogonal wavelets to be computed efficiently in linear time using a lifting scheme [124, 119]. The lifting scheme reduces the number of algebraic operations by predicting coefficients from neighbors in the subdivision grid. Bertram et al. [12] extend the lifting scheme for generalized B-spline wavelets for surfaces with quadrilateral subdivision hierarchy obtained by Catmull-Clark subdivision.

The inverse wavelet transform reconstructs the surface **f** from scaling coefficients $\mathbf{v}^k \in \mathbb{R}^3$ and wavelet coefficients $\mathbf{w}^k \in \mathbb{R}^3$ at the grid point $\mathbf{t}_b \in \mathbb{R}^2$ by

$$\mathbf{v}_b\left(\mathbf{f}\right) = \sum_{k \in V(0)} \phi_k^0(\mathbf{t}_b)\mathbf{v}^k + \sum_{j=0}^{J-1} \sum_{k \in W(j)} \psi_k^j(\mathbf{t}_b)\mathbf{w}^k, \tag{3.4}$$

where $\mathbf{v}_b\left(\mathbf{f}\right) \in \mathbb{R}^3$ denotes the b-th vertex of **f** corresponding to $\mathbf{t}_b$, J is the number of subdivision levels, $\phi_k^0(\mathbf{t})$ is the scaling function of the coarsest resolution level centered at the k-th vertex, $\psi_k^j(\mathbf{t})$ denotes the wavelet function of level j centered at the k-th vertex, $V(j)$ is the set of vertices in the j-th subdivision step, and $W(j)$ is the set of vertices added in the j-th subdivision step; therefore, $V(j+1) = V(j) \cup W(j)$.

Linear wavelet face model: The linear wavelet face model proposed by Brunton et al. [30] combines wavelet transform and PCA to obtain a statistical model that better models fine-scale details than a global PCA model. Given a set of d registered and spatially aligned 3D face scans **x**, each face is decomposed into its scaling coefficients $\mathbf{v}^k$ and wavelet coefficients $\mathbf{w}^k$. For the scaling function and the wavelet function Brunton et al. use linear B-spline basis functions, and the coefficients are computed with the lifting scheme of Bertram et al. [12]. Let $\mathbf{c}_i^k \in \mathbb{R}^3$ denote the scaling or wavelet coefficient indexed by k of face i. For each coefficient we compute PCA independently over all training faces, more formally on the sets of coefficients $\left\{\mathbf{c}_i^k | 1 \leq i \leq d\right\}$ for all k, equivalent to Section 3.2.

Each of these wavelet PCA spaces indexed by k can be computed as the eigenvectors corresponding to the three non-increasing eigenvalues of the covariance matrix

$$\mathbf{D}^k = \frac{1}{d}\sum_{i=1}^{d} \left(\mathbf{c}_i^k - \overline{\mathbf{c}}^k\right)\left(\mathbf{c}_i^k - \overline{\mathbf{c}}^k\right)^T, \tag{3.5}$$

where $\overline{\mathbf{c}}^k = \frac{1}{d}\sum_{i=1}^{d} \mathbf{c}_i^k$ denotes the mean of coefficient k over all training faces.

Let $\mathbf{u}_a^k$ denote the a-th eigenvector of $\mathbf{D}^k$. From the wavelet PCA spaces scaling or wavelet coefficients can be reconstructed equivalent to Equation 3.3 by the affine transformation

$$\mathbf{c}^k = \overline{\mathbf{c}}^k + \mathbf{U}^k\mathbf{r}^k, \tag{3.6}$$

with the 3×3 projection matrix $\mathbf{U}^k = \left(\mathbf{u}_1^k, \mathbf{u}_2^k, \mathbf{u}_3^k\right)$ and the coefficients $\mathbf{r}^k \in \mathbb{R}^3$ in wavelet PCA space.

To reconstruct the 3D face shape $\mathbf{f} \in \mathbb{R}^{3n}$ from wavelet PCA space coefficients $\mathbf{r}^k$ for all k, first all scaling and wavelet coefficients must be reconstructed using Equation 3.6. The vertices of **f** can then be reconstructed with Equation 3.4.

The linear wavelet face model represents the high-dimensional shape vectors by sets of 3-dimensional shape spaces. Within these shape spaces, each principal component potentially affects localized regions defined by the support of the corresponding basis function. Statistical models with this localized influence are referred to as local models.

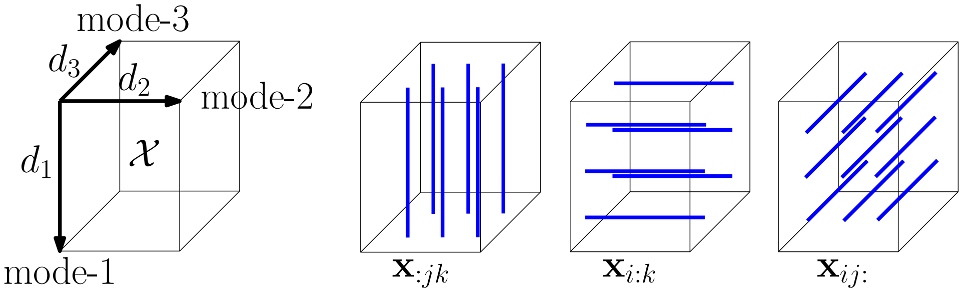


Figure 3.2: Tensor fibers for a 3-mode tensor (figure adapted from Vlasic et al. [132]). From left to right: 3-mode tensor $\mathcal{X}$, mode-1 fibers $\mathbf{x}_{:jk}$, mode-2 fibers $\mathbf{x}_{i:k}$, and mode-3 fibers $\mathbf{x}_{ij:}$.

3.4 Multilinear face model

This section introduces the basic concepts of higher-order tensors and tensor decompositions. For a more comprehensive overview of tensors and tensor decompositions, see the thesis of De Lathauwer [83] or the survey by Kolda and Bader [80].

Tensor algebra: Higher-order tensors are multidimensional arrays that generalize vectors (one-dimensional arrays) and matrices (two-dimensional arrays) to higher dimensions. The order of a tensor denotes the number of tensor dimensions. N^{th}-order tensors are also called N-mode or N-way tensors. Hence, tensor algebra denotes the generalization of linear algebra to N-mode tensors with $N \geq 3$.

Let $\mathcal{X} \in \mathbb{R}^{d_1 \times d_2 \times \ldots \times d_N}$ denote a real-valued N-mode tensor with elements $\{x_{i_1 i_2 \ldots i_N}\}$ indexed by N indices $i_j \in \{1, \ldots, d_j\}$, one index per mode. Varying only one index while keeping all other indices fixed extracts the data of the tensor along this mode. These mode vectors are known as mode fibers, which generalize the concept of rows and columns of matrices to higher dimensions. Figure 3.2 visualizes the fibers of a 3-mode tensor for each mode. We denote a fiber of mode n by $\mathbf{x}_{i_1 \ldots i_{n-1} : i_{n+1} \ldots i_N} \in \mathbb{R}^{d_n}$, where the n-th index is replaced by a colon. Mode fibers extracted from a tensor are by definition column vectors in $\mathbb{R}^{d_n}$.

The mode fibers allow for a straightforward description of the reordering of the elements of an N-mode tensor into a matrix. This tensor-to-matrix transformation is known as unfolding, flattening or matricization of a tensor. The $d_n \times \prod_{k \neq n} d_k$ matrix $\mathbf{X}_{(n)}$ denotes the mode-n unfolding of the N-mode tensor $\mathcal{X}$. There, all mode-n fibers of $\mathcal{X}$ form the columns of $\mathbf{X}_{(n)}$. Formally, the tensor element $x_{i_1 i_2 \ldots i_N}$ maps to row i_n and column j of $\mathbf{X}_{(n)}$, where $j = 1 + \sum_{k \neq n} (i_k - 1) j_k$, with $j_k = \prod_{m \in \{1, \ldots, k-1\}; m \neq n} d_m$.

The n-mode product $\times_n$ defines the multiplication of a tensor with a matrix in mode n. The n-mode product $\mathcal{Y} = \mathcal{X} \times_n \mathbf{U}_n$ of tensor $\mathcal{X}$ with a matrix $\mathbf{U}_n \in \mathbb{R}^{m_n \times d_n}$ left multiplies each n-mode fiber of $\mathcal{X}$ with $\mathbf{U}_n$. All transformed n-mode fibers of $\mathcal{X}$ form the n-mode fibers of the resulting tensor $\mathcal{Y} \in \mathbb{R}^{d_1 \times \ldots \times d_{n-1} \times m_n \times d_{n+1} \times \ldots \times d_N}$. Hence, the n-mode product relates to the matrix multiplication of the tensor unfoldings as

$$\mathcal{Y} = \mathcal{X} \times_n \mathbf{U}_n \;\Leftrightarrow\; \mathbf{Y}_{(n)} = \mathbf{U}_n \mathbf{X}_{(n)}. \tag{3.7}$$

Formally, the n-mode product is defined by the elementwise multiplication

$$(\mathcal{X} \times_n \mathbf{U}_n)_{i_1 \ldots i_{n-1} j i_{n+1} \ldots i_N} = \sum_{i_n=1}^{d_n} x_{i_1 \ldots i_N} u_{j i_n}, \tag{3.8}$$

where u_{ji_n} denotes the element of $\mathbf{U}_n$ in the j-th row and i_n-th column. For multiple mode products of distinct modes the multiplication order is arbitrary, i.e.

$$(\mathcal{X} \times_m \mathbf{U}_m) \times_n \mathbf{U}_n = (\mathcal{X} \times_n \mathbf{U}_n) \times_m \mathbf{U}_m \quad (m \neq n), \tag{3.9}$$

while for mode products of the same mode, the mode matrices are multiplied before processing the mode product, i.e.

$$\mathcal{X} \times_n \mathbf{U}_n \times_n \mathbf{V}_n = \mathcal{X} \times_n (\mathbf{V}_n \cdot \mathbf{U}_n)\,. \tag{3.10}$$

Tensor decompositions: Multidimensional data are modeled in tensors to analyze the structure of the data. These multidimensional data are usually represented in a high-dimensional space, but form only a low-dimensional subspace due to a high amount of redundancies within the data. Tensor decompositions are common methods to reduce the dimensionality of multidimensional data while preserving the structure of the data. Two common tensor decompositions are the canonical polyadic decomposition (CP decomposition) [65] and the Tucker decomposition [130, 76]. The CP decomposition is also often referred to as canonical decomposition (CANDECOMP) [36] or parallel factors (PARAFAC) decomposition [61].

The CP decomposition decomposes an N-mode tensor $\mathcal{X}$ into a sum of rank-1 tensors as

$$\mathcal{X} \approx \sum_{r=1}^{R} \mathbf{u}_r^{(1)} \circ \mathbf{u}_r^{(2)} \circ \ldots \circ \mathbf{u}_r^{(N)}, \tag{3.11}$$

where the operator "$\circ$" denotes the outer product of the vectors $\mathbf{u}_r^{(j)} \in \mathbb{R}^{d_j}$. As for matrices, tensors spanned by a set of vectors are defined to be of rank one. In general, the rank of a tensor is defined as the smallest R of all exact CP decompositions (the left and right sides of Equation 3.11 are equal). The rank of $\mathcal{X}$ is denoted by $rank(\mathcal{X})$. While for matrices it is easy to compute the rank, for N-mode tensors ($N \geq 3$), the rank computation is NP-hard [63].

The Tucker decomposition decomposes an N-mode tensor $\mathcal{X}$ into a product of a potentially lower-dimensional tensor and N matrices as

$$\mathcal{X} = \mathcal{C} \times_1 \mathbf{U}_1 \times_2 \mathbf{U}_2 \times \ldots \times_N \mathbf{U}_N, \tag{3.12}$$

with the so-called core tensor $\mathcal{C} \in \mathbb{R}^{m_1 \times \ldots \times m_N}$ and the factor matrices $\mathbf{U}_n \in \mathbb{R}^{d_n \times m_n}$. Each $\mathbf{U}_n$ defines a transformation of $\mathcal{C}$ along mode n. The columns of the $\mathbf{U}_n$ can be interpreted as the basis vectors of mode n, and the elements of $\mathcal{C}$ describe the influence of the basis vectors. This becomes clear when writing the Tucker decomposition as linear combination of rank-1 tensors as

$$\mathcal{X} = \sum_{i_1=1}^{m_1} \sum_{i_2=1}^{m_2} \cdots \sum_{i_N=1}^{d_N} c_{i_1 i_2 \ldots i_N} \mathbf{u}_{i_1}^{(1)} \circ \mathbf{u}_{i_2}^{(2)} \circ \ldots \circ \mathbf{u}_{i_N}^{(N)}, \tag{3.13}$$

where $c_{i_1 i_2 \dots i_N}$ is an element of tensor $\mathcal{C}$, and $\mathbf{u}_i^{(j)}$ denotes the i-th column of $\mathbf{U}_j$. This also reveals that the CP decomposition (Equation 3.11) is a special case of the Tucker decomposition if $\mathcal{C}$ is superdiagonal. The Tucker2 decomposition used in this thesis is another special case of the Tucker decomposition, where $\mathbf{U}_1$ is the identity matrix.

By the rules of the n-mode product (Equations 3.9 and 3.10) it follows that the Tucker decomposition is not unique as a linear transformation of the n-mode factor matrix along with mode-multiplying the core tensor with the inverse linear transformation results in another valid Tucker decomposition. Formally, this is

$$\mathcal{C} \times_n \mathbf{U}_n = \left(\mathcal{C} \times_n \mathbf{T}^{-1}\right) \times_n (\mathbf{U}_n \mathbf{T}) . \tag{3.14}$$

The $\mathbf{U}_n$ are often enforced to be orthogonal. To compute a Tucker decomposition with imposed orthogonality constraints, Kolda and Bader describe three methods, namely higher-order singular value decomposition (HOSVD) [83], higher-order orthogonal iteration (HOOI) [83], and a Newton-Grassmann optimization approach [49]. All these methods compute a Tucker decomposition for given maximum mode ranks m_2 and m_3.

HOSVD extends the matrix singular value decomposition (SVD) to N-mode tensors. For each mode n, $\mathcal{X}$ is unfolded and a matrix SVD is computed as $\mathbf{X}_{(n)} = \mathbf{U}_n \mathbf{S}_n \mathbf{V}_n^T$. HOSVD allows the data in mode n to be compressed by using only the first m_n columns of $\mathbf{U}_n$ to compute $\mathcal{C}$. This dimensionality reduction is called truncated HOSVD. HOSVD is exact if $m_n = rank(\mathbf{X}_{(n)})$ for all n; otherwise HOSVD approximates the data. In contrast to matrix SVD, HOSVD does not give the best approximation of the data.

HOOI iteratively optimizes the Tucker decomposition initialized by HOSVD. Within each iteration, both factor matrices are updated by fixing one and updating the other. That is, for a fixed mode-2 factor matrix, a tensor $\mathcal{X} = \mathcal{X} \times_2 \mathbf{U}_2^T$ is computed, and $\mathbf{U}_3$ is updated by the m_3 left singular vectors of $\mathbf{X}_{(3)}$. A similar computation is performed for a fixed mode-3 factor matrix. While HOOI gives a better approximation of $\mathcal{X}$ than HOSVD, it does not necessarily find a stationary point.

The Newton-Grassmann optimization approach iteratively optimizes the Tucker decomposition initialized by HOSVD. The Newton-Grassmann optimization approach constrains each factor matrix to a Grassmannian manifold, an equivalence class of orthogonal matrices. The Tucker decomposition is then computed by a non-linear Newton method on the product of two Grassmannian manifolds. This method converges to a stationary point.

Multilinear face model: This section introduces the multilinear face model [132] as we use it throughout this thesis. The multilinear face model is a statistical 3D face model that generalizes the concepts of PCA to 3D faces with multiple sources of variation. Possible sources of variation are shape differences of different identities and facial deformations e.g. from facial expressions or speech-related articulations. In the following we use a set of registered and spatially aligned 3D face scans of d_2 identities, each with d_3 expressions, where every face $\mathbf{x} = (x_1, y_1, z_1, \cdots, x_n, y_n, z_n)^T$ consists of n vertices $(x_i, y_i, z_i)^T$. We center each face by subtracting the mean over all training faces $\bar{\mathbf{x}}$. Let $\mathbf{x}_{ie}$ denote face i in expression e. We arrange the centered faces as mode-1 fibers in a 3-mode tensor $\mathcal{X} \in \mathbb{R}^{3n \times d_2 \times d_3}$ such that the

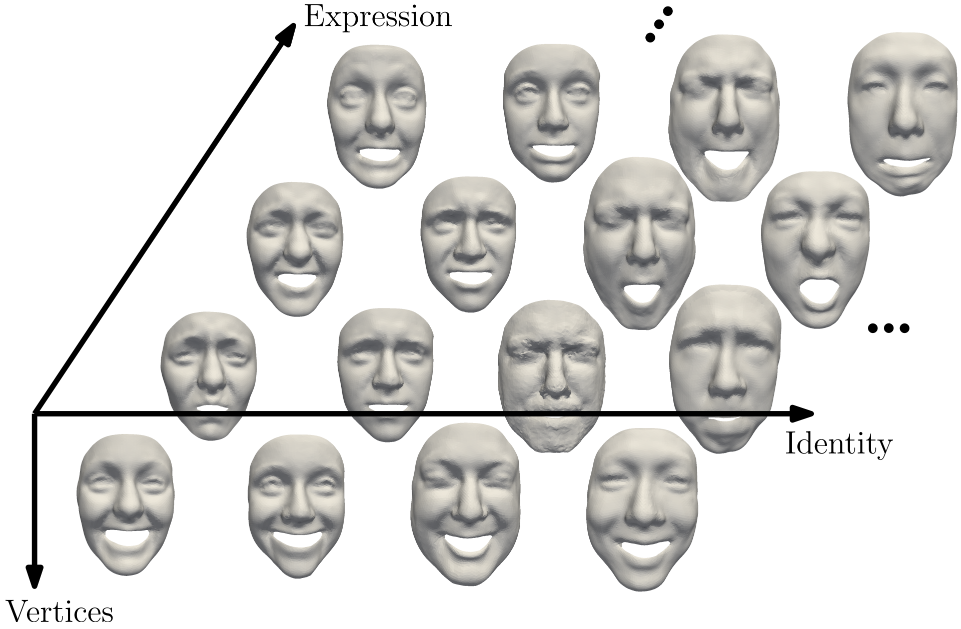


Figure 3.3: Data tensor for registered face scans of the BU-3DFE database, where the different identities align with mode 2, and the different expressions with mode 3.

different identities align with mode 2, and the different expressions with mode 3. The Tucker2 decomposition with orthogonality constraints of $\mathcal{X}$ with HOSVD

$$\mathcal{X} \approx \mathcal{M} \times_2 \mathbf{U}_2 \times_3 \mathbf{U}_3 \tag{3.15}$$

results in a tensor $\mathcal{M} \in \mathbb{R}^{3n \times m_2 \times m_3}$ called a *multilinear model*, which is the mode-1 multiplication of the core tensor and the identity matrix. There, m_2 and m_3 denote the number of columns of $\mathbf{U}_2$ and $\mathbf{U}_3$, respectively. The columns of $\mathbf{U}_2$ span the mode-2 subspace, called identity space, and the columns of $\mathbf{U}_3$ span the mode-3 subspace, called expression space. Each row of $\mathbf{U}_2$ represents an identity in identity space, and each row of $\mathbf{U}_3$ represents an expression in expression space.

Assuming a multivariate Gaussian distribution in identity and expression space, the multilinear model is a generative statistical model, as is the PCA model. It represents a registered 3D face $\mathbf{f} \in \mathbb{R}^{3n}$ by

$$\mathbf{f} = \bar{\mathbf{x}} + \mathcal{M} \times_2 \mathbf{w}_2^T \times_3 \mathbf{w}_3^T, \tag{3.16}$$

where $\mathbf{w}_2 \in \mathbb{R}^{m_2}$ and $\mathbf{w}_3 \in \mathbb{R}^{m_3}$ are the identity and expression coefficients.

Figure 3.3 shows an example of a 3-mode tensor for registered 3D scans of the BU-3DFE database, where the faces are arranged as mode-1 fibers, the different identities align with mode 2, and the different expressions with mode 3. Note that this is only a symbolic visualization, as we arrange the centered faces in the tensor rather than the registered face scans.

Figure 3.4 visualizes the variations of three principal components of identity mode (left) and expression mode (right) for a multilinear model learned from the registered BU-3DFE

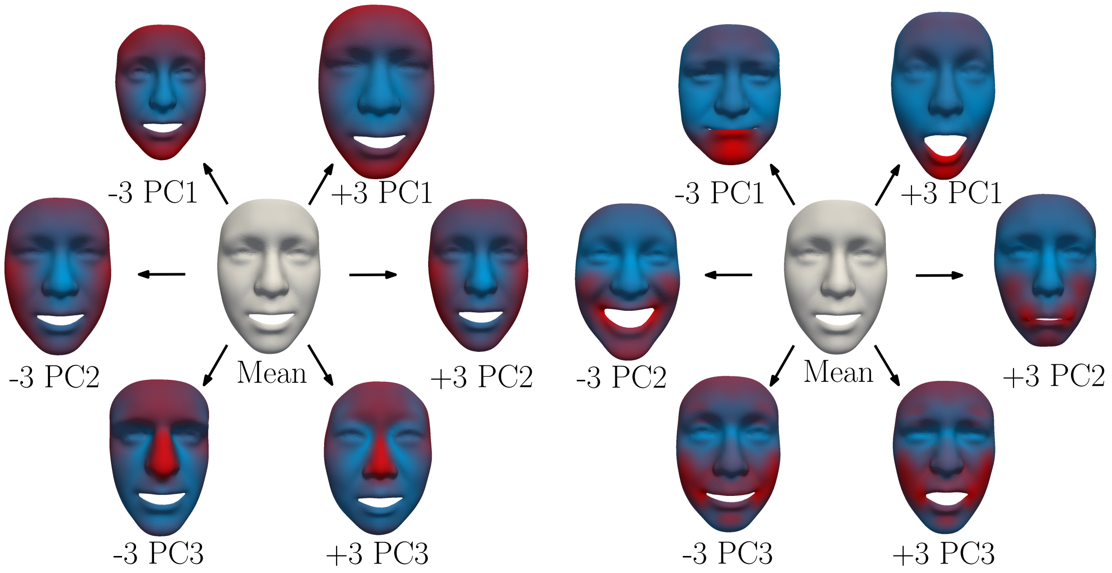


Figure 3.4: Variations of three principal components of the multilinear model learned from the registered BU-3DFE database. The magnitude of the vertex displacements from the mean face is color coded from blue (zero) to red (maximum). Left: Identity mode. Right: Expression mode.

database. The variation along each principal component for both modes is in the range of $\pm 3\sigma$, where σ denotes the standard deviation of the principal component.

Chapter 4

Statistical motion analysis

“Nothing happens until something moves.”

– Albert Einstein

This chapter describes how to statistically analyze dynamic 3D face data. As discussed in Chapter 2, statistical methods require the data to be in correspondence. Performing statistical analysis of 3D motion data is a challenging problem, since it requires a robust registration method that establishes spatial and temporal correspondence for motion sequences of different identities performing different expressions. This is difficult since different identities have different face shapes and each face undergoes strong geometric deformations in the course of different expressions. While it is possible to apply the previously mentioned facial registration methods (see Section 2.3) for each frame of the sequence individually, these methods do not preserve the temporal coherence of the motion.

To robustly compute a spatial and temporal registration, we jointly process entire motion sequences. Figure 4.1 shows an overview of our method. To be robust to fast motions, we need a good initialization for our motion registration. For this, we fully automatically predict landmarks for an entire motion sequence using a MRF-based method. We then use a learned multilinear model as statistical prior for a fully automatic dense registration of the motion sequences. To be independent of illumination changes, our overall approach depends only on geometric information, but texture information could be added using a higher-dimensional multilinear model.

After registration, each motion sequence is represented by a vector of coefficients for identity and a high-dimensional curve for expression. This representation allows to use standard techniques to perform statistical analysis on 3D faces in motion. We apply our framework to four applications. Namely, we propose different ways to synthesize new motion sequences, and we recognize dynamic expressions.

4.1 Landmark prediction for sequence data

This section describes a MRF-based method that predicts facial landmarks for entire motion sequences. Given a sequence $\mathbf{s}_1, \cdots \mathbf{s}_F$ of F face scans, the method predicts for each frame of the sequence L facial landmarks that are in correspondence across the entire sequence.

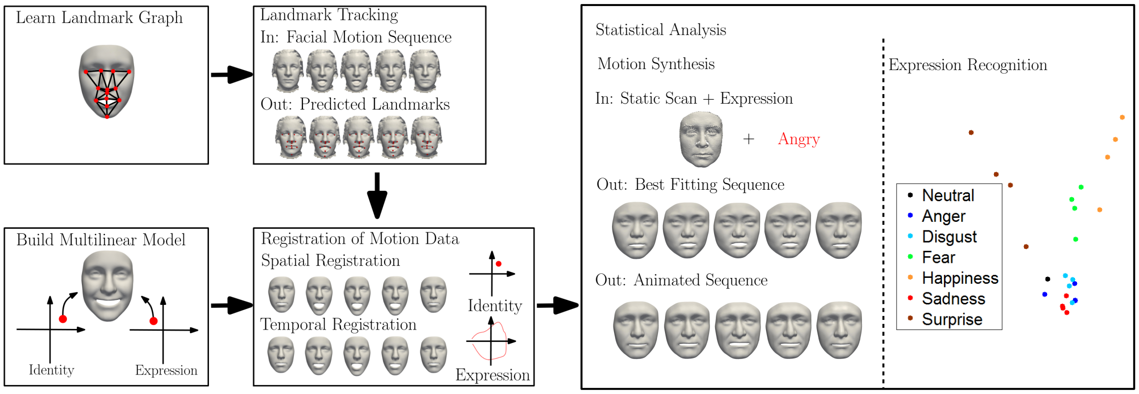


Figure 4.1: Overview of our proposed method. Left: training of landmark graph (top) and multilinear model (bottom). Middle: landmark prediction (top) and motion sequence registration (bottom). Right: statistical analysis.

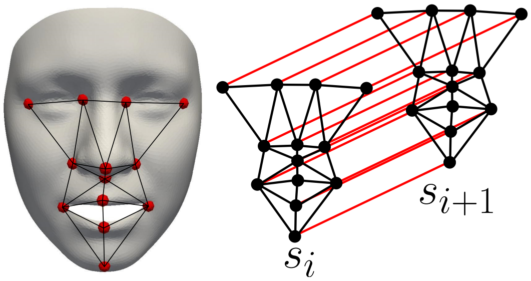


Figure 4.2: Markov networks. Left: Selected landmarks (red) and landmark graph (black) for single frame. Right: Temporal edges (red) between corresponding landmarks of consecutive frames.

A MRF consists of a set of random variables $\mathbf{l}_j$ with probability distributions $\phi_j(\mathbf{l}_j)$ and pairwise connections between random variables $\mathbf{l}_j$ and $\mathbf{l}_k$ with pairwise probability distributions $\psi_{j,k}(\mathbf{l}_j, \mathbf{l}_k)$. Within a MRF, the random variables are represented by nodes and the pairwise connections between random variables by edges. The landmark prediction method of Salazar et al. [114] makes it possible to find landmarks on static 3D face shapes by learning the statistical distributions of a descriptor on known landmarks and by training a MRF to learn geometric properties of these landmarks. We extended this approach to motion sequences, as described in the following.

4.1.1 Learning of landmark graph

We manually define an anatomically meaningful MRF for all landmarks across the entire sequence. For each of the F frames, we predict L landmarks. Let $\mathbf{l}_j^i$ denote the j-th landmark of i-th frame of the sequence. Each landmark $\mathbf{l}_j^i$ is represented by a node and each connection between two landmarks by an edge within the MRF. Figure 4.2 (left) shows the landmark

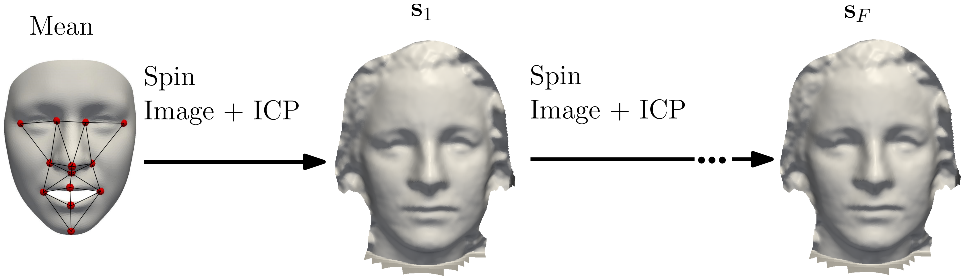


Figure 4.3: Initial alignment computation.

graph for one frame; Figure 4.2 (right) shows the temporal edges between corresponding landmarks of consecutive frames. During training, we learn the node potentials ϕ_j and the edge potentials $\psi_{j,k}$ for edges between nodes of one frame, and the edge potentials $\psi_{j,j}$ for temporal edges between corresponding nodes of consecutive frames. The joint probability of all nodes and edges is

$$p(\mathbf{l}_1^1, ..., \mathbf{l}_L^F) = \frac{1}{Z}\prod_i \prod_j \phi_j(\mathbf{l}_j^i) \prod_{j,k} \psi_{j,k}(\mathbf{l}_j^i, \mathbf{l}_k^i) \prod_{j,j} \psi_{j,j}(\mathbf{l}_j^i, \mathbf{l}_j^{i+1}), \tag{4.1}$$

where Z is a normalization factor. We assume all node and edge potentials to be multivariate Gaussian distributed. We use the mean curvature, Gaussian curvature, and shape index to compute the multivariate Gaussian distribution $\phi_j = \mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{l}_j}, \boldsymbol{\Sigma}_{\mathbf{l}_j}\right)$ for the node potential, where $\boldsymbol{\mu}_{\mathbf{l}_j}$ is the mean vector and $\boldsymbol{\Sigma}_{\mathbf{l}_j}$ the covariance matrix computed over landmark $\mathbf{l}_j$ on the training data. Here, we compute over all training faces for landmark $\mathbf{l}_j$ the vector $(H_{\mathbf{l}_j}, K_{\mathbf{l}_j}, SI_{\mathbf{l}_j})^T$, where $H_{\mathbf{l}_j}$ denotes the mean curvature, $K_{\mathbf{l}_j}$ denotes the Gaussian curvature, and $SI_{\mathbf{l}_j}$ denotes the shape index at $\mathbf{l}_j$. For the edge potentials, we compute two multivariate Gaussian distributions $\psi_{j,k} = \mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{l}_j\mathbf{l}_k}, \boldsymbol{\Sigma}_{\mathbf{l}_j\mathbf{l}_k}\right)$ and $\psi_{j,j} = \mathcal{N}\left(\mathbf{0}, \boldsymbol{\Sigma}_{\mathbf{l}_j}\right)$. Here, $\boldsymbol{\mu}_{\mathbf{l}_j\mathbf{l}_k}$ and $\boldsymbol{\Sigma}_{\mathbf{l}_j\mathbf{l}_k}$ are the mean vector and the covariance matrix of edge lengths and orientations on edge $(\mathbf{l}_j, \mathbf{l}_k)$ over all training faces.

4.1.2 Landmark tracking

We want to predict facial landmarks for a sequence of scanned frames, showing a face in motion. We assume that expressions change smoothly, and hence adjacent frames are similar. Our landmark prediction method for entire motion sequences consists of three parts. First, we compute a rigid transformation that aligns $\mathbf{s}_i$ with the landmark graph (Figure 4.3). Second, we select for each node a possible set of labels within each frame (Figure 4.4). Third, we predict landmarks for an entire sequence using the selected label sets.

To compute a rigid alignment, we compute correspondences between the mean face $\bar{\mathbf{f}}$ of the training data and the first frame of every sequence using the spin-image-based method of Johnson and Hebert [74]. A spin image describes the local neighborhood of an oriented point

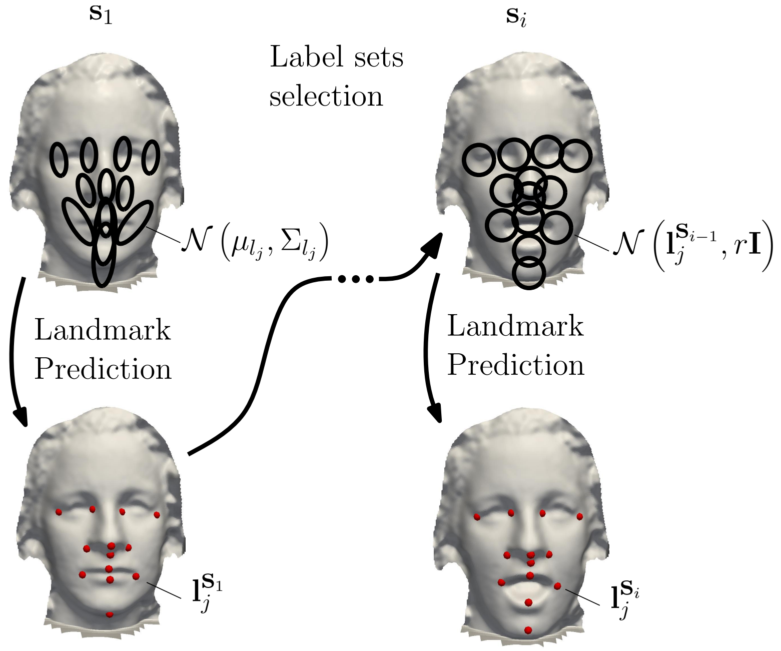

$\mathbf{s}_1$
Label sets selection
$\mathbf{s}_i$
$\mathcal{N}\left(\mu_{l_j}, \Sigma_{l_j}\right)$
$\mathcal{N}\left(\mathbf{l}_j^{\mathbf{s}_{i-1}}, r\mathbf{I}\right)$
Landmark Prediction
Landmark Prediction
$\mathbf{l}_j^{\mathbf{s}_1}$
$\mathbf{l}_j^{\mathbf{s}_i}$

Figure 4.4: Consecutive selection of the label sets for each node. For the first frame we select label sets based on the learned Gaussian distributions of the node potentials. For all other frames, we select label sets based on a sphere around the predicted landmarks of the previous frame.

with respect to the local coordinate system of the point. For an oriented point $\mathbf{x}$, each nearby vertex is assigned two parameters, which encode its relative position in the local coordinate system of $\mathbf{x}$. The spin image of $\mathbf{x}$ collects all assigned 2D values of vertices within a specified neighborhood and is represented by an image. Spin images of different oriented points can be compared, grouped, and finally used to establish correspondences between two meshes. The use of local coordinate systems ensures that spin images are invariant under rigid transformations. While the correspondence we determine this way can contain incorrect matches and outliers, we use RANdom SAmple Consensus (RANSAC) [52] to get a good rigid alignment. RANSAC randomly selects four points as a minimum point set defining a valid rigid alignment with the assumption that these points are inliers. This initial set is extended by all consistent points. The solution computed by RANSAC is based on only one of the consistent point sets with few outliers. We refine the resulting rigid transformation using ICP.

We aim to find landmark positions $\mathbf{l}_j^i$ that maximize Equation 4.1. For this we need to select a set of possible labels for each landmark. To select this label set, we process a sequence in consecutive order and independently predict the landmarks for each $\mathbf{s}_i$ with respect to the landmarks predicted for the last frame. For the first frame, we select as a label set all vertices $\mathbf{x}_{\mathbf{l}_j}$ that are within one standard deviation of the mean of $\mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{l}_j}, \boldsymbol{\Sigma}_{\mathbf{l}_j}\right)$. To predict the landmarks of a single frame, we maximize Equation 4.1 without temporal edges using loopy belief propagation [137]. This belief propagation iteratively finds a maximum of Equation 4.1 by passing messages between connected nodes. Since expression changes between consecutive frames are small, predicted landmarks of adjacent frames need to be close. Therefore, we select all points within a sphere of radius r centered at the predicted landmarks of the previous frame as the label set of the current frame.

With the selected label sets of the entire motion sequence, we perform a loopy belief propagation for the entire sequence. The temporal edges keep the landmarks of adjacent frames close.

4.2 Multilinear space of face identity and expression

This section describes how the multilinear model can be used as statistical prior for model fitting, and introduces appropriate error measurements to evaluate the trained model.

4.2.1 Multilinear model as statistical prior

If we have shapes of only one identity (or one expression), the multilinear model reduces to PCA. For PCA, the data are modeled by a multivariate Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$. That is, all shapes are centered and the centered shape space is rotated such that the major axes of $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$ are aligned with the directions of maximal variance. Using a Gaussian distribution as statistical prior to constrain the shape in parameter space is described in [43, Chapter 2.2]. The data are then normalized, such that $\mathbf{\Sigma} = \mathbf{I}$. This allows the use of $\mathcal{N}(\mathbf{0}, \mathbf{I})$ as a statistical prior.

A face is represented as $\mathbf{f}(\mathbf{w})$, where $\mathbf{w}$ is the set of coefficients in PCA space. The PCA model can be fitted to a new face scan $\mathbf{s}$ by finding $\mathbf{w}$, such that $\mathbf{f}(\mathbf{w})$ is close to $\mathbf{s}$. This problem is commonly solved using two energy terms that are optimized simultaneously. The first term measures how closely $\mathbf{f}(\mathbf{w})$ resembles $\mathbf{s}$. The second term measures the negative log-probability of $\mathbf{w}$ with respect to $\mathcal{N}(\mathbf{0}, \mathbf{I})$. This choice has the disadvantage of introducing a bias towards the model mean. One way to avoid this bias is to optimize the first energy term only while restricting $\mathbf{w}$ to stay within the learned probability distribution. Ideally, this restriction would find the best $\mathbf{w}$ inside a hypersphere of radius c centered at the origin. Here, the parameter c controls the amount of variability. In practice, a simpler restriction is to find the best $\mathbf{w}$ inside a centered axis-aligned hypercube of side length $2c$. This restricts each component of $\mathbf{w}$ independently, which allows the use of efficient constrained optimization algorithms.

If we have multiple identities in multiple expressions, we search for coefficients $\mathbf{w}_2$ and $\mathbf{w}_3$, such that $\mathbf{f}(\mathbf{w}_2, \mathbf{w}_3)$ is close to $\mathbf{s}$. We outline how the previously discussed method can be extended to this scenario. Note that unlike in the case of PCA, this is a non-linear model that treats identity and expression spaces independently. In the following, we focus on identity space, and similar arguments apply to expression space. If $\bar{\mathbf{f}}$ were equal to the mean of all identities, the multilinear model would model identity space by a standard normal distribution.

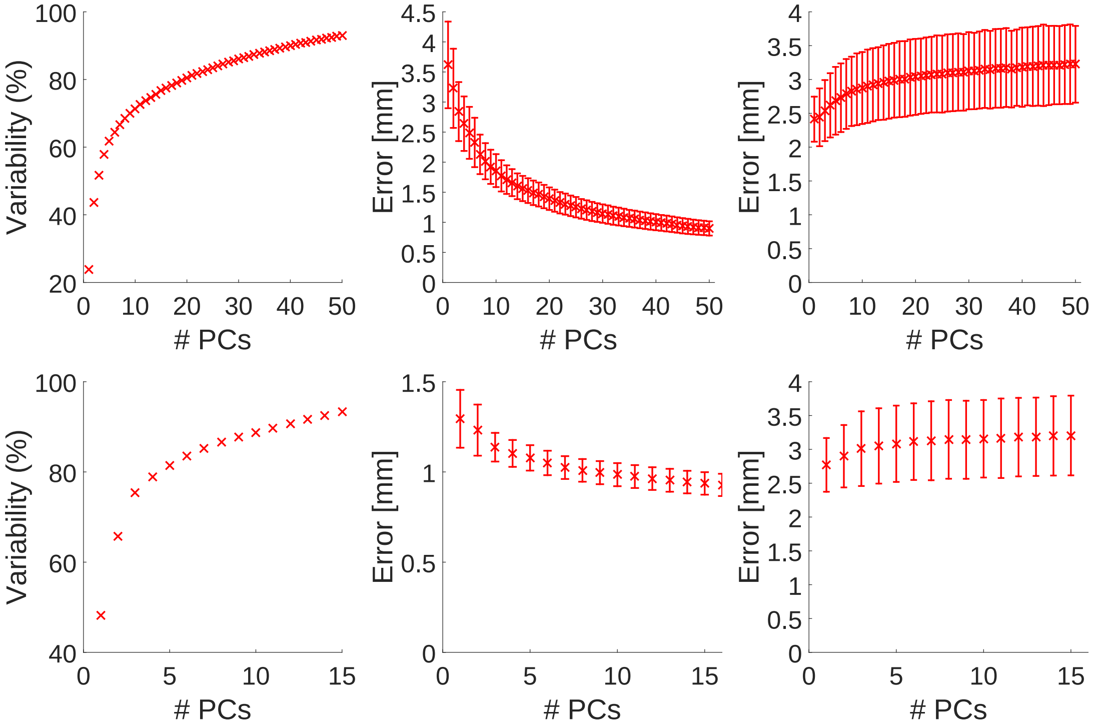


Figure 4.5: Compactness, generalization, and specificity of identity mode (top) and expression mode (bottom).

However, since this is not the case in general, letting $\mathcal{N}(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$ denote the Gaussian fitted to identity space, $\boldsymbol{\mu}_2 \neq \mathbf{0}$ and $\boldsymbol{\Sigma}_2 \neq \mathbf{I}$. In practice, we expect the distribution not to deviate too far from a standard normal distribution. Hence, for simplicity, we set $\boldsymbol{\Sigma}_2 = \mathbf{I}$. However, setting $\boldsymbol{\mu}_2 = \mathbf{0}$ is problematic, as $\mathbf{0}$ is a singularity in identity space: if $\mathbf{w}_2 = \mathbf{0}$, then $\mathbf{f}(\mathbf{w}_2, \mathbf{w}_3) = \bar{\mathbf{f}}$, independently of the value of $\mathbf{w}_3$. For this reason, we use the correct mean in our fitting approach. As each row of the matrix $\mathbf{U}_2$ represents one identity of the training data, the mean identity $\boldsymbol{\mu}_2 = \bar{\mathbf{w}}_2$ is computed as the average of all rows of $\mathbf{U}_2$. This allows us to fit the model to the data while restricting $\mathbf{w}_2$ to lie in the hypercube of side length $2c_2$ centered at $\bar{\mathbf{w}}_2$. Similarly, $\mathbf{w}_3$ is restricted to lie in the hypercube of side length $2c_3$ centered at $\bar{\mathbf{w}}_3$.

4.2.2 Evaluation of multilinear model

We use a multilinear model to separate identity and expression for human faces. To ensure that the multilinear model is applicable for our face data, we evaluate it for the registered training database. For details regarding the training data used, see Section 4.5.

We quantitatively evaluate the quality of the optimization with the widely-used measures compactness, generalization, and specificity (see Section 3.2) that we extend to the multilinear case. For a good multilinear model, the identity and expression spaces should ideally be compact, general, and specific. This evaluation also allows us to pick a number of components for identity (m_2) and expression (m_3) that preserves a high amount of variability without overfitting the training data. Fig. 4.5 visualizes the results.

Compactness: We independently measure the compactness of the model for identity and expression space as $C(k) = \frac{\sum_{i=1}^{k} \lambda_i}{\sum_{i=1}^{rank(\mathbf{D})} \lambda_i}$, where $k \in \{1, \ldots, d_2\}$ or $k \in \{1, \ldots, d_3\}$, and λ_i denotes for each mode the i-th eigenvalue of $\mathbf{D} = \frac{1}{d_3}\mathbf{X}_{(2)}\mathbf{X}_{(2)}^T$ or $\mathbf{D} = \frac{1}{d_2}\mathbf{X}_{(3)}\mathbf{X}_{(3)}^T$, respectively.

Generalization: The generalization error is measured in a leave-one-out fashion. For the identity mode, each subject is once fully excluded from training and the resulting model is used to reconstruct all excluded scans. The error is then measured as the average Euclidean vertex distance between all corresponding vertices. We perform this measurement for all subjects, and report mean and standard deviation of the distances. The error for the expression mode is computed accordingly by excluding once each expression.

Specificity: To measure the specificity of the model, we randomly choose 10000 Gaussian distributed samples in identity and expression space, and reconstruct a face $\mathbf{f}$ for each sample using Eq. 3.16. For each sample, we compute the minimum of the average Euclidean vertex distance over the training data. We then consider the mean and standard deviation over all samples.

To evaluate generalization and specificity of the model for identity mode, the number of expression components is fixed to 7, which gives 85% compactness. Similarly, while evaluating the expression mode, the number of identity components is fixed to 30, which gives 86% compactness.

Our identity and expression space should ideally be compact, general and specific. Based on the analysis shown in Fig. 4.5, we choose $m_2 = 30$ and $m_3 = 7$.

4.3 Registration of motion data

In this section, we discuss how to register motion sequences of faces. Our method uses a learned multilinear model as statistical prior. We make some assumptions about the motion data for the proposed registration method. First, the identity stays fixed for an entire sequence. Second, every motion sequence starts and ends in a neutral expression. Third, expressions change smoothly, and hence are similar in adjacent frames. To statistically analyze faces in motion, the motion sequences need to be spatially and temporally registered.

4.3.1 Spatial registration

To fit the multilinear model to a sequence $\mathbf{s}_1, \cdots \mathbf{s}_F$ of F face scans, we minimize the energy $E : \mathbb{R}^{m_2+Fm_3} \rightarrow \mathbb{R}$

$$E = E_D + w_L E_L + w_T E_T, \tag{4.2}$$

with respect to the coefficients $\mathbf{w}_2$ for identity, and $\mathbf{w}_{3,1}, ..., \mathbf{w}_{3,F}$ for expression. The energy E is composed of the energy E_D to fit the model to the scan geometry, E_L to fit the model to given landmarks, and E_T to keep the changes between consecutive coefficients in expression space small. The parameter w_L controls the influence of the given landmarks, and the parameter w_T controls the trade-off between the accuracy of the geometric fitting and the temporal smoothness of the m_3-dimensional curve in expression space.

Data: The data term measures the distance between the model and the data for each frame of the sequence. The data term is

$$E_D = \sum_{i=1}^{F} \frac{1}{\sum\limits_{j=1}^{n} b_{ij}} \sum_{j=1}^{n} b_{ij} \left\| \mathbf{v}_j \left(\mathbf{f}_i \right) - \mathbf{nn}_j \right\|^2, \tag{4.3}$$

where $\mathbf{f}_i = \bar{\mathbf{x}} + \mathcal{M} \times_2 \mathbf{w}_2^T \times_3 \mathbf{w}_{3,i}^T$ denotes the reconstruction of frame i (Eq. 3.16), $\mathbf{v}_j \left(\mathbf{f}_i \right)$ denotes the j-th vertex of $\mathbf{f}_i$, and $\mathbf{nn}_j$ is the nearest neighbor of $\mathbf{v}_j \left(\mathbf{f}_i \right)$ in $\mathbf{s}_i$ computed using a point-to-plane distance measure. We use binary weights $b_{ij} \in \{0, 1\}$ to control whether a point is considered for fitting. To lower the influence of outliers, we only consider nearest neighbors that are closer than 10mm and with an angle between the normals smaller than 45 degrees.

Landmarks: The landmark energy for L given landmarks is defined as

$$E_L = \frac{1}{L} \sum_{i=1}^{F} \sum_{j=1}^{L} \left\| \mathbf{v}_{r_j} \left(\mathbf{f}_i \right) - \mathbf{l}_j \right\|^2, \tag{4.4}$$

where $\mathbf{l}_j \in \mathbb{R}^3$ is the j-th landmark and r_j the index of corresponding vertex on the statistical face model.

Temporal smoothness: The temporal smoothness term measures the similarity of adjacent frames and the distance of the start and endpoint of the expression curve to the neutral expression. The temporal smoothness term is

$$E_T = \frac{1}{m_3} \left(\left\| \mathbf{w}_{3,1} - \mathbf{w}_3^{ne} \right\|^2 + \left\| \mathbf{w}_{3,F} - \mathbf{w}_3^{ne} \right\|^2 + \sum_{i=1}^{F-1} \left\| \mathbf{w}_{3,i} - \mathbf{w}_{3,i+1} \right\|^2 \right), \tag{4.5}$$

where $\mathbf{w}_3^{ne}$ is the vector describing the training data in the neutral expression (in expression space).

4.3.2 Optimization

The energy E in Equation 4.2 is non-linear. One way to solve this system is by linearizing the problem. This can be done by fixing the coefficients of all but one mode and solving for the remaining mode [132, 42, 135]. Since this linearization does not consider identity and expression simultaneously, it can lead to a solution that is not a local minimum over combined identity and expression space. As the objective function E is analytically differentiable with respect to the coefficients $\mathbf{w}_2$ and $\mathbf{w}_{3,1}, ..., \mathbf{w}_{3,F}$, to remedy this, we solve the non-linear problem using L-BFGS [92], a quasi-Newton method with linear constraints.

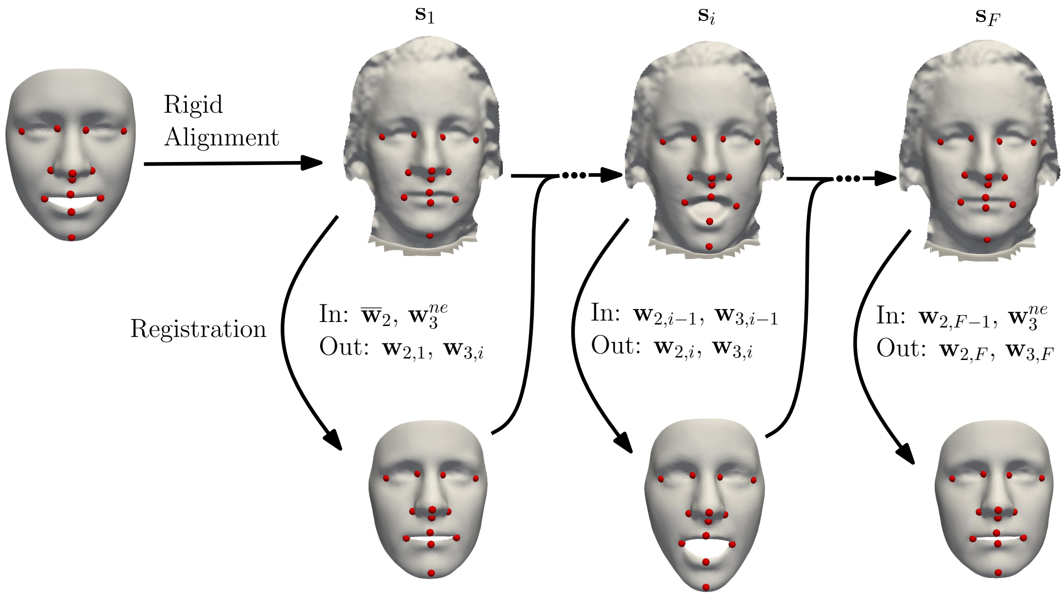

$\mathbf{s}_1$
$\mathbf{s}_i$
$\mathbf{s}_F$
Rigid Alignment
Registration
In: $\overline{\mathbf{w}}_2$, $\mathbf{w}_3^{ne}$
Out: $\mathbf{w}_{2,1}$, $\mathbf{w}_{3,i}$
In: $\mathbf{w}_{2,i-1}$, $\mathbf{w}_{3,i-1}$
Out: $\mathbf{w}_{2,i}$, $\mathbf{w}_{3,i}$
In: $\mathbf{w}_{2,F-1}$, $\mathbf{w}_3^{ne}$
Out: $\mathbf{w}_{2,F}$, $\mathbf{w}_{3,F}$

Figure 4.6: Overview of the initialization process.

Computational complexity: We evaluate the computational complexity for one iteration step of our spatial registration method. We build a k-d tree for each frame of the target sequence with m vertices. The complexity of building a k-d tree is $O(m \log m)$ [85]. Computing the nearest neighbors for all n template vertices takes $O(nm^{\frac{2}{3}})$ time. A single evaluation step of Equation 4.3 takes $O(Fn)$, and a single evaluation of its gradient $O((m_2 + Fm_3)n)$ time. Evaluating Equation 4.4 takes time $O(L)$, and a single evaluation of its gradient takes $O((m_2 + Fm_3)L))$ time. Evaluating Equation 4.5 and its gradient takes $O(Fm_3)$ time.

Let t_c denote the number of optimization steps required to reach a local minimum. Assuming L to be a small constant with $L \ll n$ and $L \ll m$, the overall time complexity is $O(F(m \log m + nm^{\frac{2}{3}}) + t_c(m_2 + Fm_3)n)$.

Initialization: Since E is non-linear, we need a good initialization for the optimization. To fit a multilinear model to a sequence of 3D faces, a spatial rigid alignment and initial coefficients $\mathbf{w}_2$ and $\mathbf{w}_3$ are needed. While other methods manually initialize the spatial alignment or the coefficients [132, 42], our method is fully automatic. Figure 4.6 gives an overview of our initialization approach.

We start by computing the transformation from the local coordinate system of each scan of the sequence into the local coordinate system of the multilinear model. To compute the rigid transformation, we use the automatically predicted landmarks. To be less affected by expression changes, we just use the landmarks placed at eyes and nose to compute the rigid alignment. To minimize the influence caused by noise at the landmarks, rigid ICP is performed. After initialization, the rigid alignment computed for each $\mathbf{s}_i$ is fixed.

We compute initial coefficients $\mathbf{w}_{2,i}$ and $\mathbf{w}_{3,i}$ by fitting the multilinear model to each frame of the motion sequence via minimizing E. For these fitting steps, all available landmarks are used. To register a single frame, for the first frame $\mathbf{w}_{2,1}$ is initialized to the mean of the identity $\overline{\mathbf{w}}_2$, and for the first and last frames, $\mathbf{w}_{3,1}$ and $\mathbf{w}_{3,F}$ are initialized to the neutral expression $\mathbf{w}_3^{ne}$. For all other frames, we use the result of the previous frame to initialize the coefficients, since we assume adjacent frames to be similar. The initial $\mathbf{w}_2$ are computed by averaging all $\mathbf{w}_{2,i}$, since the identity stays fixed across the sequence.

Multi-resolution optimization: To register an entire motion sequence, we perform several iterations of minimizing E. To increase the computational performance, a multi-resolution approach that iteratively optimizes E is employed (Equation 4.2) in different resolution levels. The low-resolution steps aim to establish the rough overall shape together with a good initialization of the performed expression. The high-resolution step aims to pick up finer mesh details. This step leads to a significant improvement in the running time of the method.

4.3.3 Temporal registration

After spatial registration, a motion sequence is represented by identity coefficients $\mathbf{w}_2$ and an ordered set of coefficients for expression $\mathbf{w}_{3,i}$. The ordered set of coefficients for an expression can be seen either as a point ($\in \mathbb{R}^{Fm_3}$) or as a high-dimensional curve ($\in \mathbb{R}^{m_3}$). To perform statistics on registered motion sequences, they need to be in correspondence. While all faces already spatially correspond, we also need to establish a temporal coherence. Since the motion sequences differ in frame number and speed of performed expression, the maximum expression magnitude is reached at different times, and resampling with respect to number of frames does not yield a good registration.

One method to temporally register motion sequences is to use dynamic time warping (DTW) [112]. DTW uses dynamic programming to align temporal sequences by computing a mapping between both sequences that minimizes the dissimilarity. While DTW could be used to align pairs of registered facial motion sequences, it is computationally expensive. Zhou and De la Torre [140] extend DTW to minimize the sum of pairwise distances between multiple sequences. In contrast to DTW, this generalized time warping (GTW) is of linear computational complexity as it is optimized using a Gauss-Newton method. While GTW is computationally efficient, it requires solving a non-convex optimization, and hence our proposed method is much simpler.

Since we temporally register the entire registered motion database, we use a resampling method instead. Specifically, the expression curve $\mathbf{w}_{3,i}$ is resampled according to its arc length. The resampling of the expression curve leads to a good temporal correspondence, since E_T forces large expression changes to be represented by large changes in expression space, and since each motion sequence starts and ends neutral. In the following, $\mathbf{w}_{3,i}$ denotes the coefficients of the resampled expression curve.

4.4 Statistical analysis of motion data

This section outlines how to perform statistical analysis on registered motion data and shows four applications. Namely, different ways to synthesize new motion sequences are discussed, by morphing between existing expressions, by exploring learned PCA spaces of identity coefficients and expression curves, and by animating static face scans. Furthermore, we outline how to perform expression recognition.

4.4.1 Expression morphing

One way to generate new motion sequences is to morph between a start and an end frame of the same subject. For this, we select two arbitrary frames of the same subject, possibly from different (registered) motion sequences. These frames are represented by one identity and one expression coefficient each. Let $\mathbf{w}_2^s$, $\mathbf{w}_3^s$ and $\mathbf{w}_2^e$, $\mathbf{w}_3^e$ denote the coefficients of the start and end frames, respectively. Since the identity is the same for both sequences, the identity coefficients $\mathbf{w}_2^s$ and $\mathbf{w}_2^e$ are similar. Hence, the identity coefficient of the new sequence is chosen as the average of $\mathbf{w}_2^s$ and $\mathbf{w}_2^e$ and the expression coefficients of the new motion sequence linearly interpolate between $\mathbf{w}_3^s$ and $\mathbf{w}_3^e$.

4.4.2 Combined PCA of identity and expression for synthesis

To synthesize new motion sequences of one expression, we learn a PCA space of all identity coefficients $\in \mathbb{R}^{m_2}$ and a PCA space on all expression curves $\in \mathbb{R}^{Fm_3}$of a particular expression. To obtain new motion sequences, we combine samples from both learned PCA spaces. Choosing a sample from the identity coefficients PCA space gives a new identity coefficient within the identity space of the learned multilinear model. A sample from the expression curve PCA space gives a new expression curve within the expression space of the multilinear model. This allows the generation of new motion sequences by combining the sampled identity coefficients and expression curve.

4.4.3 Static scan animation

A more challenging problem is to animate a static (unregistered) scan $\mathbf{s}$ in a neutral expression to perform a specified motion sequence. This application is related to the problem of transferring a given motion from one given subject to another, which is considered in the literature [132, 42]. Note, however, that our application of animating a given input scan from scratch is more challenging than performing motion transfer, as we need to find the best subject to transfer the motion from in a fully automatic way.

To synthesize a motion sequence for $\mathbf{s}$, we find the subject in the registered database that performs the specified motion sequence and that best matches $\mathbf{s}$. Let $\mathbf{w}_2$, $\mathbf{w}_{3,i}$ denote the weights of said motion sequence. To animate $\mathbf{s}$, we fix the expression coefficient $\mathbf{w}_{3,1}^s$ of $\mathbf{s}$

to $\mathbf{w}_{3,1}$, initialize the identity coefficient $\mathbf{w}_2^s$ of $\mathbf{s}$ to $\mathbf{w}_2$, and fit the multilinear model to $\mathbf{s}$ by minimizing E_D (Eq. 4.3). The resulting $\mathbf{w}_2^s$, together with $\mathbf{w}_{3,i}$, represent $\mathbf{s}$ in motion.

It remains to discuss how to find the sequence that best matches $\mathbf{s}$ automatically. We perform the fitting described above for each sequence with the specified motion in the database and measure the dissimilarity of the sequence and $\mathbf{s}$ as the distance between $\mathbf{w}_2$ and $\mathbf{w}_2^s$. To compute the distance, we weigh each component of identity space by the amount of variability captured by said component (i.e. the singular value of the mode covariance matrix). The best match is the sequence that has the lowest dissimilarity.

4.4.4 Expression recognition

Since the multilinear model separates variations due to different identity from variations due to expression changes, expression recognition is a natural application of our shape space. The right side of Figure 4.1 shows a plot of the expression space obtained by performing multi-dimensional scaling (MDS). Note that different expressions form clusters.

We use a method to perform expression recognition of motion sequences of faces that is designed to evaluate the quality of the spatial and temporal registration of the motion sequences. To this end, we classify the motion sequences using a method to perform static 3D facial expression recognition that is based on landmarks. More specifically, we use a sparse set of landmark positions to measure the distance between two faces as the sum of the squared Euclidean distances between corresponding landmarks. This distance measure is then used in a maximum likelihood classification framework to estimate the likelihood of each expression class, as in Mpiperis et al. [102].

This method first needs to find the frame of the sequence that exhibits the highest level of expression, and second uses landmark positions on this frame for the classification. Since each motion sequence is registered temporally, the frame with the highest expression level can be found as the midpoint of the expression curve. Furthermore, since each frame is registered spatially, the extraction of a predefined set of landmarks is straightforward.

Note that while this simple method is designed to evaluate the quality of the spatial and temporal registration, we will show that it leads to results comparable to those of state-of-the-art dynamic expression recognition techniques.

4.5 Evaluation

This section evaluates our registration pipeline.

Training data: For training the landmark graph and for training and evaluation of the multilinear model, we use models of the BU-3DFE database [139]. For details on the database, see Section 3.1. We use the template fitting method of Salazar et al. [114], based on provided ground truth landmarks of the database, to register all models. The template we use for registration consists of 5996 vertices.

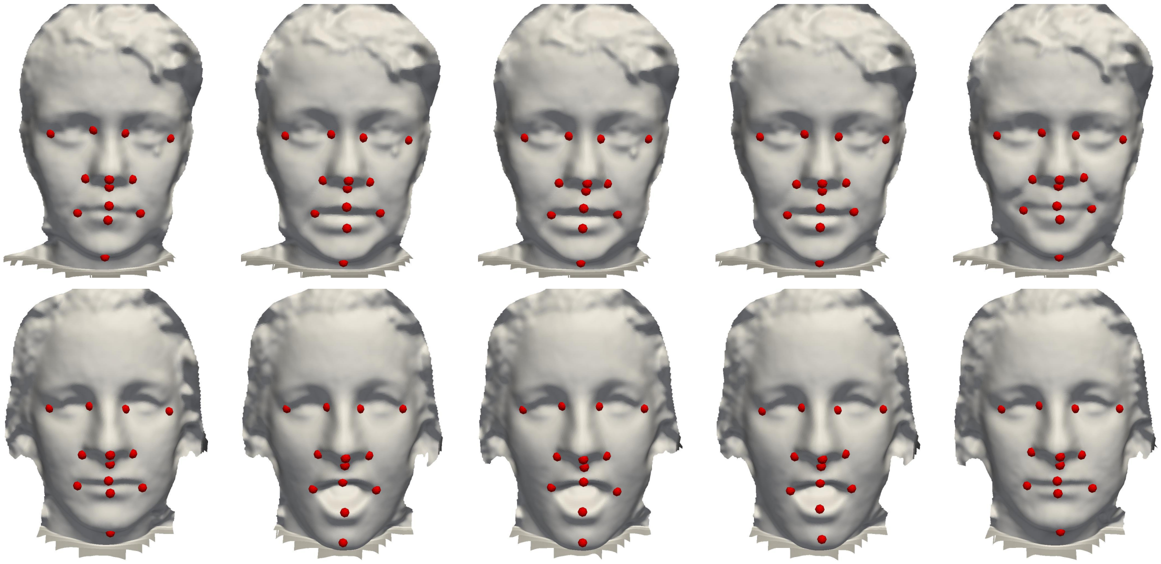


Figure 4.7: Result of landmark prediction on sequences.

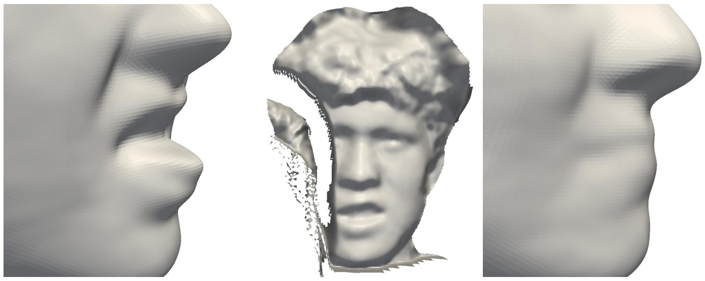


Figure 4.8: Challenging models of the BU-4DFE database. Left: Visible tongue. Middle: Scanner noise. Right: Smooth geometry.

Test data: To evaluate our registration framework we use motion sequences of the BU-4DFE database [138]. A more detailed explanation of the data is given in Section 3.1.
Reproducibility: Our approach is implemented in C++, using OpenCV [107], ANN [8], and LBFGSB [92]. We publish the statistical multilinear face model learned from the registered BU-3DFE database and code to fit the multilinear model to static input face scans [18].

4.5.1 Landmark prediction

We predict landmarks for all 606 motion sequences. The initial alignment computation using spin images is successful for 599 sequences (98.8%). Strong geometric differences between consecutive frames of a motion sequence, caused by scanner noise (middle of Figure 4.8), are one reason for failure. Due to the absence of ground truth landmarks, to evaluate the landmark prediction, we visually inspect the predicted landmark positions. The landmarks are

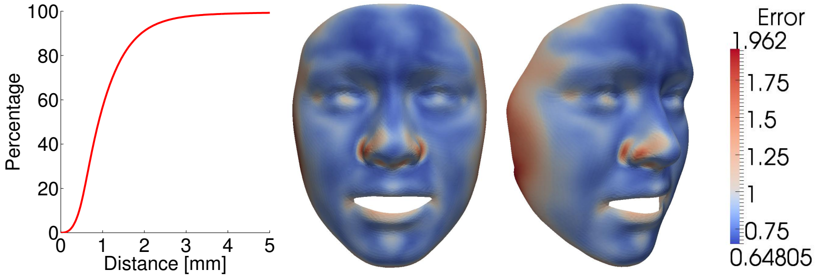


Figure 4.9: Cumulative error plot (left) and color-coded face of median distance per vertex (right) in mm.

successfully predicted for 561 sequences (93.7%). Cases where the landmark prediction fails are where the lip is geometrically not discriminative (Figure 4.8, right), or sequences where the tongue is tracked instead of the lip due to similar curvature (Figure 4.8, left). Figure 4.7 shows frames of sequences where the landmarks are successfully tracked.

4.5.2 Spatial registration

Since some of the motion sequences violate the assumption that motions start and end in neutral expression, we remove them manually. We use the remaining 501 sequences for our further experiments. To minimize E, we choose $w_L = 0.2$ during initialization, and $w_L = 0.0$ and $w_T = 10000$ while registering the motion sequence. Two resolution levels are used to register the motion sequences. The optimization performs 6 low-resolution steps (using about 10% of the vertices), and 3 high-resolution steps (using the full mesh resolution).

To evaluate the spatial registration, we compare the registration result to the scanned motion sequences. For 470 sequences (93.8%), the spatial registration is successfully computed. Reasons for failure are erroneously predicted landmarks, or problems with tracking the lips due to a non-descriptive geometry. To measure the quality of the spatial registration, the nearest neighbor distance between the registration result and the data is computed for each registered face. Figure 4.9 shows the cumulative error for all vertices of all 470 successfully registered faces. Furthermore, Figure 4.9 shows the median of all errors per vertex. Note that 56% of all vertices have a distance of less than 1 mm to the data, and the per-vertex median error is lower than 1 mm for 73% of the vertices. Reasons for facial parts with lower accuracy are the smoothness of the scanned motion sequences (e.g. left and right subnosal), or noise near the facial border.

Additionally, Figure 4.10 visualizes scanned motion sequences and registration results. The sequences are chosen to show the performance of different expressions. Note that the overall shape of the registration result and the face scans is similar and the expressions are well captured.

We also compare the result of our spatial registration to the template-fitting method of Salazar et al. [114], applied to motion sequences frame by frame using our predicted land-

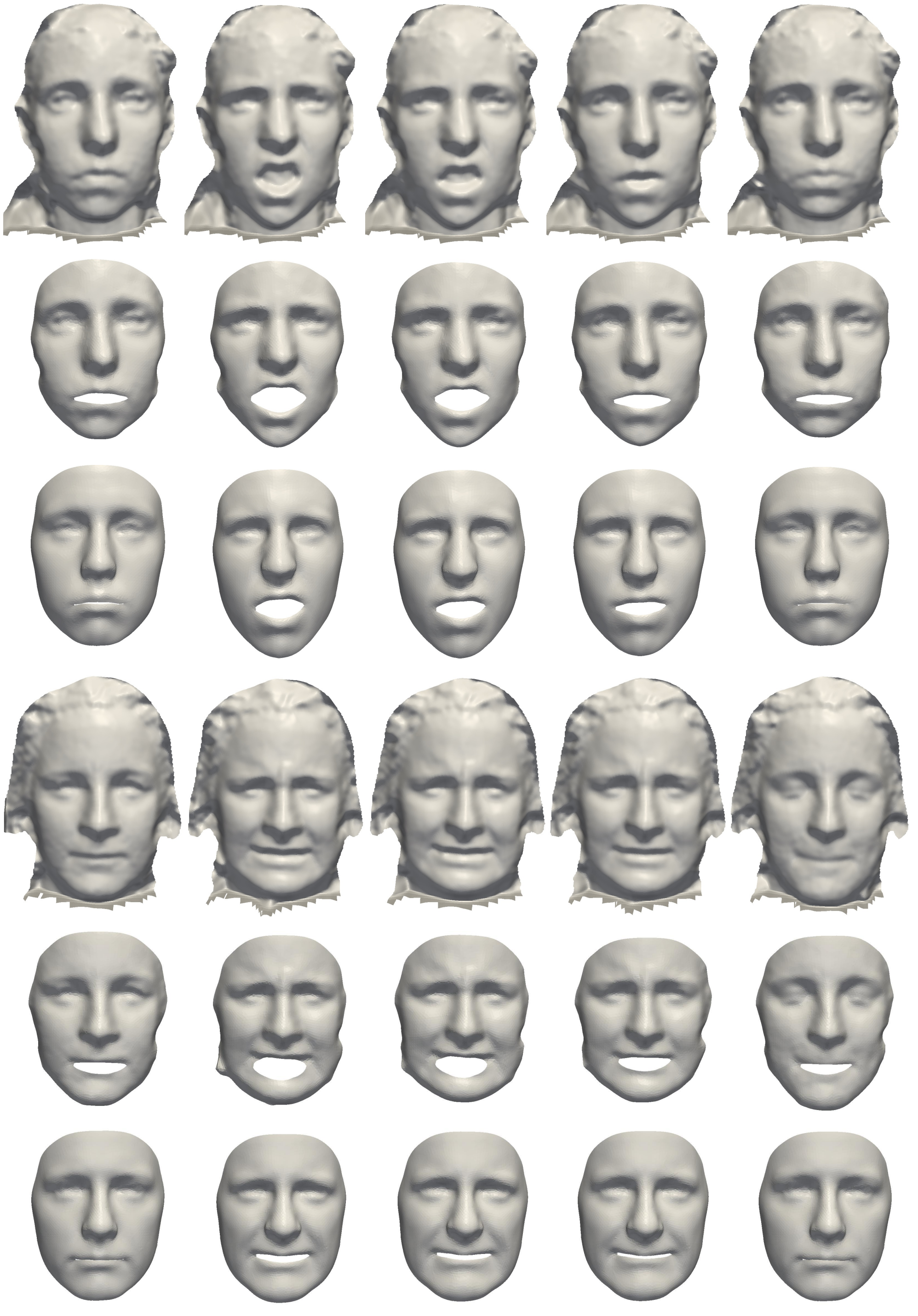


Figure 4.10: Comparison to a template-fitting method [114] applied to each frame individually for two motion sequences. Rows 1 and 4: Frames of motion sequences. Rows 2 and 5: Template-fitting result. Rows 3 and 6: Our result.

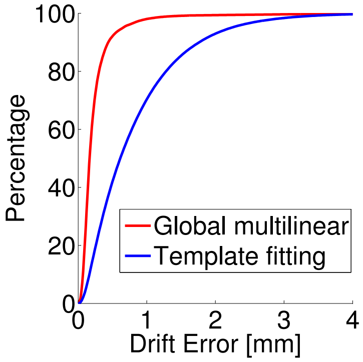


Figure 4.11: Comparison of a template-fitting method [114] applied to each frame individually, vs. our method. Cumulative point movements between consecutive frames are computed over six motion sequences.

marks. Figure 4.10 shows the result of the template-fitting method for two sequences. While for the upper sequence, the shape of the mouth is fitted well, the noise close to the border of the face is reconstructed. The registration for the same sequence by our registration approach looks more realistic. For the second row of Figure 4.10, the template-fitting method fails, while our method gives a good registration result. Furthermore, fitting each frame individually breaks the temporal coherence of the motion sequence, which causes drift. To get a quantitative measurement for this drift, we measure the distance of corresponding vertices of consecutive frames, since differences due to expression changes of consecutive frames are small. Figure 4.11 shows a cumulative plot for all differences for 6 randomly chosen motion sequences (which include the two sequences shown in the top rows of Figure 4.10.), registered with the template-fitting method and our method. For our method, 98% of the distances are below 1 mm, while for the template fitting method less than 70% of the distances are below 1 mm. This indicates that our method better preserves the temporal coherence.

The spatial registration is forced to start and end neutral due to the terms of E_T pulling towards $\mathbf{w}_3^{ne}$ for first and last frames, and the initialization of $\mathbf{w}_{3,1}$ and $\mathbf{w}_{3,F}$ to $\mathbf{w}_3^{ne}$. Without these terms of the temporal smoothness energy and without initializing to the neutral expression, the sequence registration can be used for sequences without neutral start and end frames.

4.5.3 Temporal registration

To evaluate the quality of the temporal registration, we compare the temporal correspondence of different motion sequences before and after temporal registration. The left of Figure 4.12 shows spatially registered motion sequences, resampled according to the number of frames (left). These motion sequences do not reach their maximum intensity of the performed expression at the same time. After temporal registration, the motion sequences reach the maximum intensity of the performed expression at the middle of the sequence.

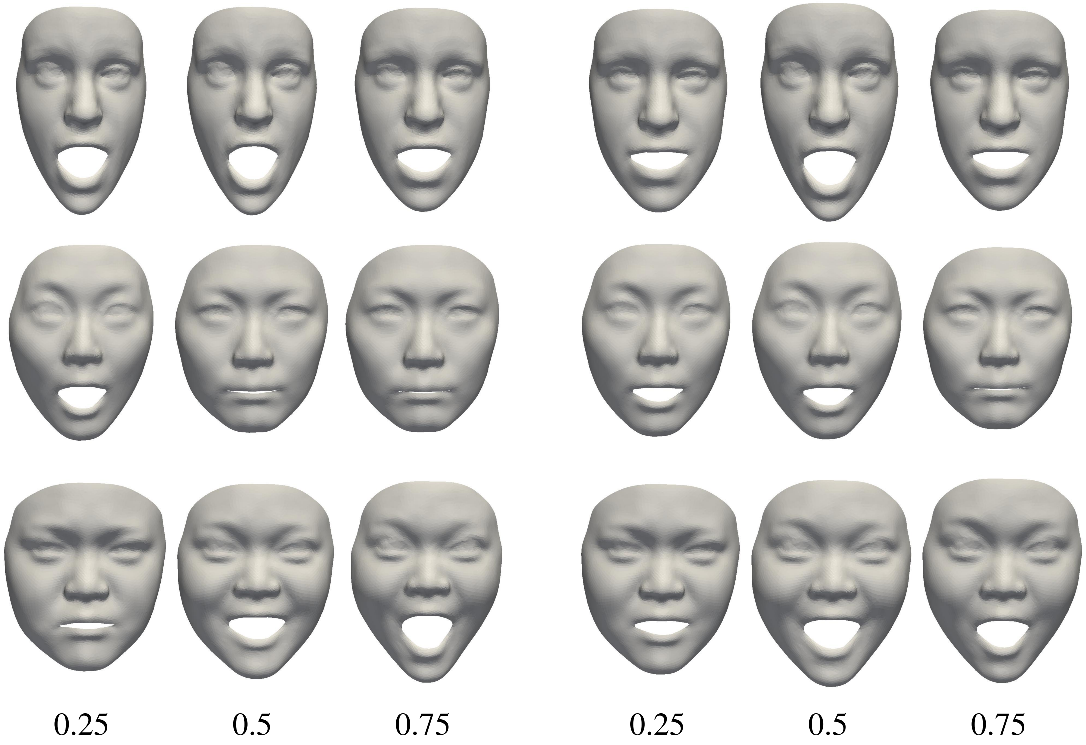


Figure 4.12: Uniformly sampled expression curve (parametrized between 0 and 1) with respect to frame number (left) and with respect to arc length of expression curve (right).

4.5.4 Expression morphing

For the synthesis of new motion sequences, we first show results for the expression morphing. While for one subject, any pair of frames can be used for the expression morphing, we choose two frames with a high-intensity expression from different motion sequences. This ensures that the new motion sequence has a significant expression change. Figure 4.13 shows selected starting (left) and ending (right) key frames, and uniformly sampled frames of the resulting motion sequences (middle). For both sequences, the originally selected key frames look similar to starting and ending frames of resulting sequences, and the deformation over time looks realistic.

4.5.5 Combined PCA of identity and expression for synthesis

To generate new motion sequences for one particular expression, we obtain new identity coefficients by sampling the PCA space learned over all identity coefficients. To obtain new expression curves, we sample the PCA space learned over all expression curves of a particular expression. Combining new identity coefficients with new expression curves produces new motion sequences. To obtain the happy motion sequences shown in Figure 4.14, we combine the mean of the identity coefficients PCA space with variations of the expression curve along

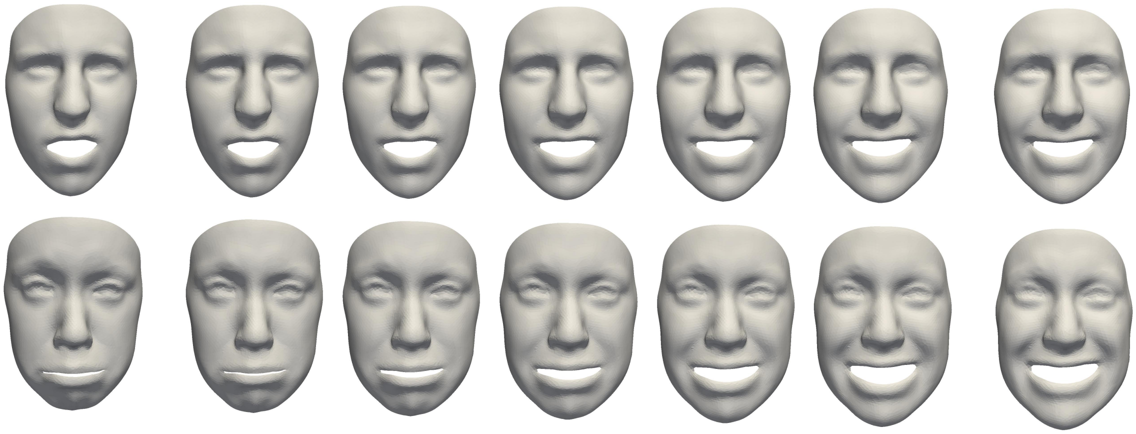


Figure 4.13: Expression morphing between frames of different motion sequences. Left/Right: Resulting frame of registration. Middle: Synthesized motion sequence. Top: disgusted to happy. Bottom: sad to happy.

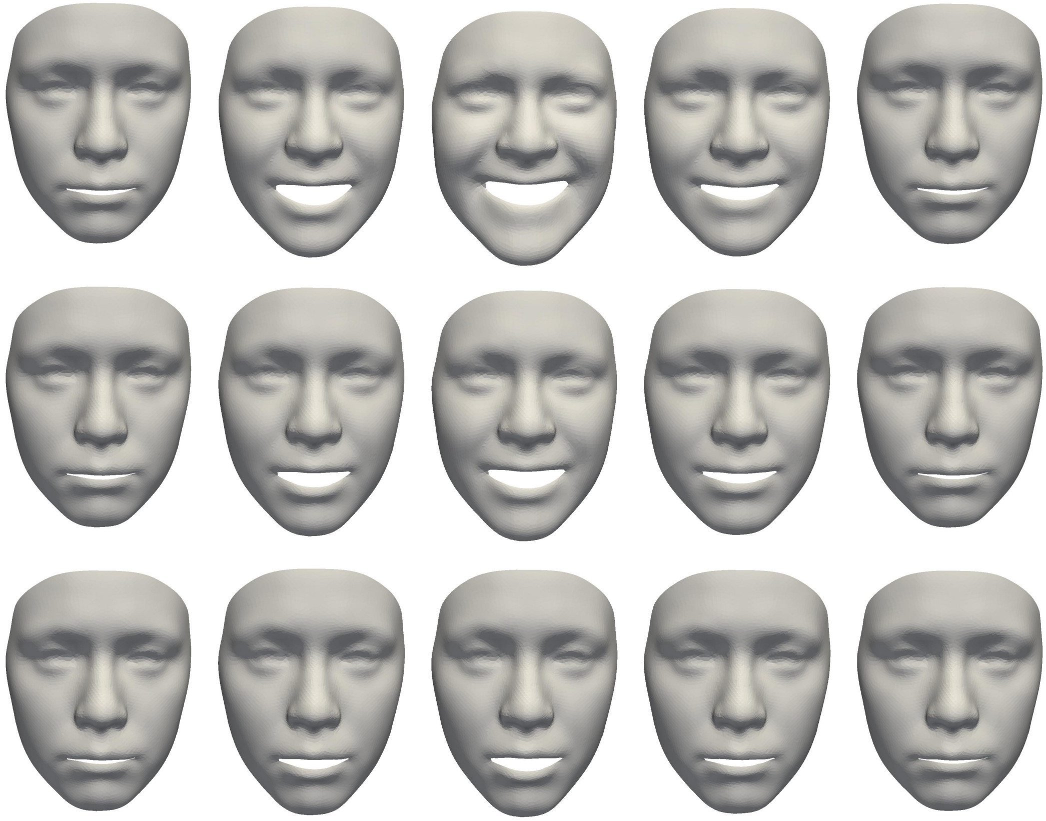


Figure 4.14: New happy motion sequences for average identity, generated by varying the expression curves along the first principal component within the PCA space of all happy expression curves. Variation: Top: $+3\sigma$. Middle: 0. Bottom: -3σ.

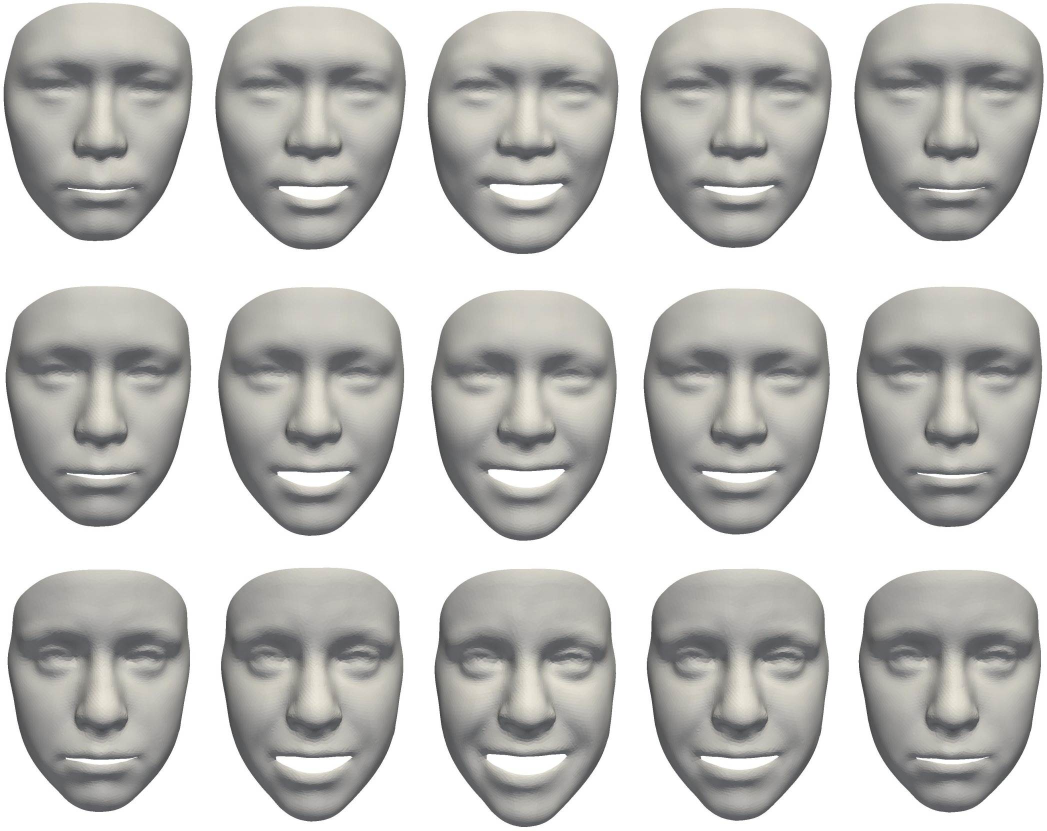


Figure 4.15: New identities in average happy motion, generated by varying the identity coefficients along the first principal component within the PCA space of all identities. Variation: Top: $+3\sigma$. Middle: 0. Bottom: -3σ.

the first principal component of the learned expression curves PCA space. The variation along the first principal component is within -3σ and $+3\sigma$, where σ is the singular value of the happy expression curves covariance matrix, associated with the first principal component. In this case, the variation along the first principal component controls the intensity of the performed happy expression.

To generate happy motion sequences for different identities, we combine new identity coefficients with the average expression curve. Figure 4.15 shows new identities that are obtained by variation along the first principal component of the PCA space, learned over the identity coefficients of all motion sequences. The variation along the first principal component is within -3σ and $+3\sigma$, where σ is the first singular value of the covariance matrix of all motion sequence identity coefficients. In this case, all rows show happy motion sequences for different face shapes. While the overall face shape changes, the variation along the first principal component especially affects the nose shape and the shape of the forehead.

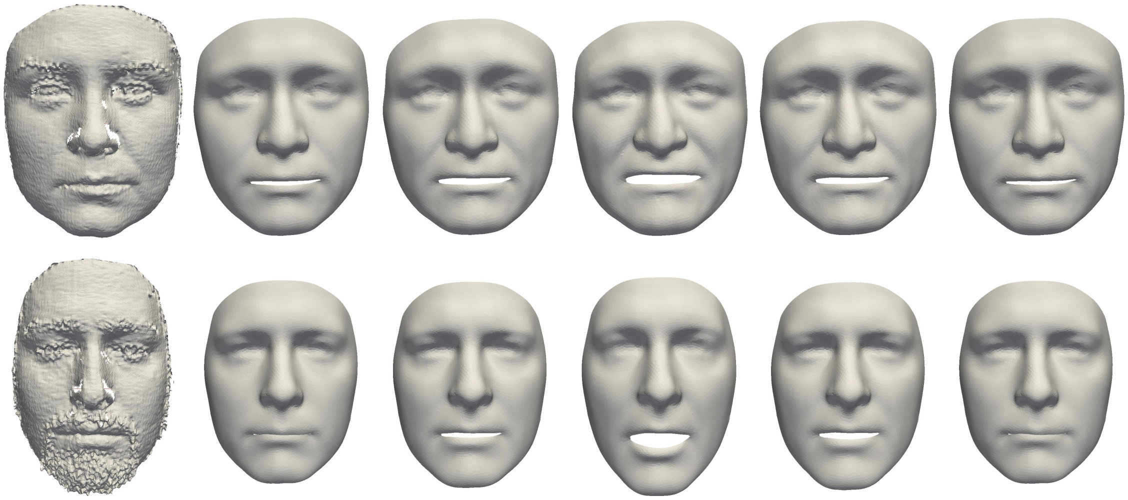


Figure 4.16: Motion synthesis. Left: scan. Right: synthesized motion. Top: angry motion. Bottom: surprised motion.

4.5.6 Static scan animation

We show results for synthesizing motion sequences for a static input scan from scratch. As input, we use scans of different subjects of the Bosphorus database [116], which captures static scans of different subjects performing different facial expressions. While it would be possible to use the method described in Section 4.1 to establish the initial alignment, we use the provided landmarks to remove one possible source of error. Figure 4.16 shows the target faces of two identities (left) and uniformly sampled frames of the synthesized motion for the angry and surprised expressions. Since we use a global multilinear model for synthesis, the result resembles the global shape of the input scan, but does not contain all fine-scale details. Nevertheless, for all examples, the fitting result is similar to the target face and the synthesized motion looks realistic. We furthermore compare the result of the motion sequence with the recorded sequence present in the BU-4DFE database. Figure 4.17 shows a registered motion sequence (top) and a synthesized motion sequence (bottom). The expression of the motion sequence that is selected to transfer the motion from is more expressive than the acquired sequence, which results in an expressive synthesized motion sequence. Note that while the result of the motion synthesis differs from the acquired motion sequence, both performed expressions look realistic.

4.5.7 Expression recognition

For expression recognition, we use the expression subsets anger, happiness, surprise and happiness, sadness, surprise to get values comparable to the ones in prior works [115, 50, 84]. We use the registered BU-3DFE database for training, and perform expression recognition for registered motion sequences of the BU-4DFE database. Our classification rate for the expressions

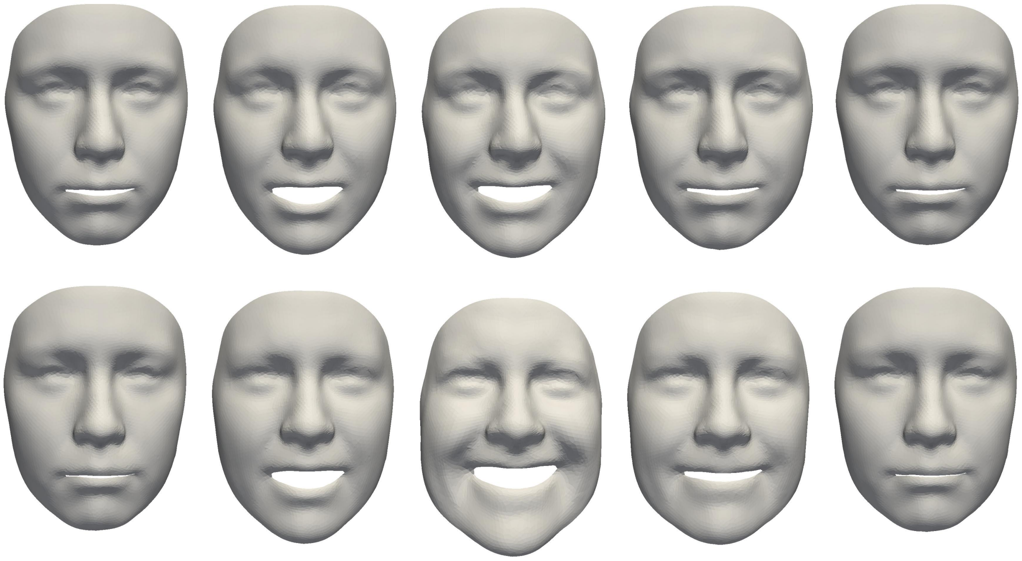


Figure 4.17: Motion synthesis and acquired sequence. Top: Original registered motion sequence. Bottom: Synthesized motion sequence for start frame of original motion sequence.

Ours	AN	HA	SU
AN	**90.14**	4.23	5.63
HA	3.95	**89.47**	6.58
SU	3.80	3.80	**92.41**

[50]	AN	HA	SU
AN	**97.32**	2.68	0.00
HA	2.00	**96.33**	1.67
SU	2.54	1.00	**96.46**

Table 4.1: Expression recognition for the expressions anger, happiness, and surprise. Left: our method with classification rate of 90.71%. Right: method of Fang et al.[50] with classification rate of 96.71%.

Ours	HA	SA	SU
HA	**90.79**	1.32	7.89
SA	2.53	**87.34**	10.13
SU	5.06	1.27	**93.67**

[50] / [84]	HA	SA	SU
HA	**97.32 / 95.00**	1.43 / 3.33	1.25 / 1.67
SA	1.11 / 1.67	**98.89 / 91.67**	0.00 / 6.67
SU	4.61 / 0.00	4.36 / 10.00	**91.03 / 90.00**

Table 4.2: Expression recognition for the expressions happiness, sadness, and surprise. Top: our method with classification rate of 90.60%. Bottom: methods of Fang et al. [50] and Le et al. [84] with classification rates of 95.75% and 92.22%.

anger, happiness, and surprise is 90.71% (see Table 4.1). Sandbach et al. [115] achieve for the same expressions 81.93% (they do not provide the full confusion matrix), and Fang et al. [50] 96.71%. For the expressions happiness, sadness, and surprise, we recognize 90.60% (see Table 4.2) correctly, while Le et al. [84] recognize 92.22%, and Fang et al. 95.75%. Compared to the other methods, our recognition method is more general. While our method performs the training on a different database than the classification, the other methods use the 4D motion sequences for training and prediction. Note that our method still has a similar performance, which indicates that our spatial and temporal registration are of high quality.

4.5.8 Influence of landmarks and multi-resolution registration

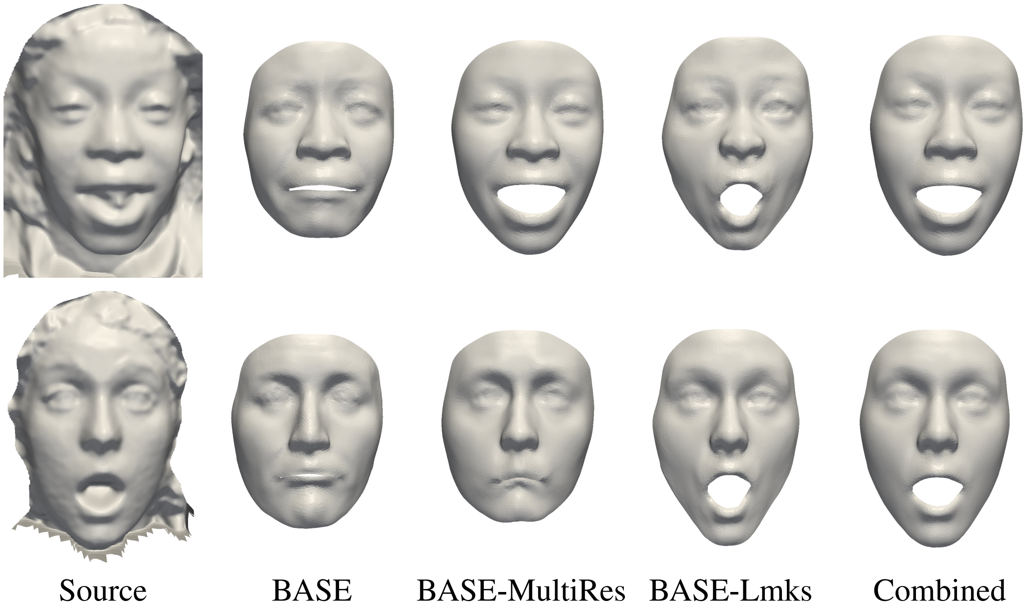


Figure 4.18: Registered sequences for different methods. From left to right: BASE, BASE-MultiRes (use of a multi-resolution approach to minimize the BASE energy), BASE-Lmks (combination of BASE with landmarks without using a multi-resolution approach), and our combined approach. Top: Successfully registered due to multi-resolution fitting. Bottom: Successfully registered by influence of landmarks.

This section shows the influence of landmarks and a multi-resolution optimization on the registration. Let BASE denote the optimization of the energy

$$E_{BASE} = E_D + w_T E_T, \tag{4.6}$$

with E_D and E_T as defined in Equations 4.3 and 4.5. In contrast to BASE, our method has two major algorithmic differences. First, our approach uses a multi-resolution framework during optimization, and second, we predict landmarks for motion sequences and use these landmarks during registration.

Method	BASE	BASE-MultiRes	BASE-Lmks	Combined
# Sequences	412 (82.2%)	455 (90.8%)	437 (87.2%)	470 (93.8%)

Table 4.3: Number of successfully registered sequences for different methods. From left to right: BASE, BASE-MultiRes (use of a multi-resolution approach to minimize the BASE energy), BASE-Lmks (combination of BASE with landmarks without using a multi-resolution approach), and our combined approach.

Influence of multi-resolution optimization: To show the influence of a multi-resolution optimization, we use a multi-resolution approach to optimize E_{BASE} and call this BASE-MultiRes. Table 4.3 shows that BASE successfully registers 412 motion sequences, while BASE-MultiRes successfully registers 455 motion sequences. Running BASE-MultiRes for a sequence with 95 frames, using a non-optimized single-threaded implementation on a standard PC takes approximately 37 minutes. Running BASE with the same number of iteration steps, but always using the full resolution, takes approximately 104 minutes. Using a multi-resolution optimization improves the quality of the registration and leads to a significant speed-up of the algorithm.

Influence of landmarks: To show the influence of landmarks, we combine the optimization of BASE with landmarks, by minimizing E (see Eq. 4.2) without using a multi-resolution approach, and call this BASE-Lmks. Table 4.3 shows that BASE-Lmks successfully registers 437 motion sequences, compared to 412 motion sequences with BASE. The use of landmarks during fitting makes the algorithm more robust to fast motions, where the expression difference between consecutive frames is large.

Combination: Our approach, which combines BASE with a multi-resolution approach and the use of landmarks, successfully registers 470 motion sequences, compared to 412 (BASE), 455 (BASE-MultiRes), and 437 (BASE-Landmarks). Hence, the combination of multi-resolution and landmarks for fitting performs best. Figure 4.18 shows two sequences that are successfully registered by our combined approach, while BASE fails.

4.6 Summary

In this chapter we presented a general and robust approach to fully automatically register 3D faces in motion. The resulting representation is used to perform statistical analysis. Our proposed method predicts landmarks for 3D facial motion sequences and uses these landmarks to initialize our sequence registration. We use a global multilinear model for registration that represents each motion sequence by a vector of coefficients for identity and a high dimensional curve for expression. We use this representation to synthesize new motion sequences and to recognize expressions. We show that the resulting registration result is of high quality, where 56% of all vertices are at most 1 mm away from the input data. We demonstrate the use of our method to synthesize new motion sequences, by generating arbitrary artificial new motion sequences for static face scans of different identities. Furthermore, we achieve classification rates of 90.71% to recognize the expressions anger, happiness, and surprise and 90.60% to recognize the expressions happiness, sadness, and surprise.

While the global multilinear model used for registration preserves the global facial shape well, it fails to capture fine-scale details. To capture more fine-scale details while being robust to noise and occlusions, we combine wavelet transform and multilinear models to obtain a multilinear wavelet model in Chapter 5.

Further limitations of multilinear face models are that the model quality degrades if the vertex correspondence is inaccurate, if not every person is captured in every expression, if face scans are noisy or partially occluded, or if expressions are erroneously labeled. Groupwise optimization methods make it possible to overcome these limitations. In Chapter 6 we present a groupwise correspondence optimization method; in Chapter 7 we propose a framework to robustly learn a multilinear model from 3D face databases with missing data, corrupt data, wrong semantic correspondence, and inaccurate vertex correspondence.

A further application for registered facial motion sequences is the design of gear that fits well despite varying facial expressions. In Chapter 8, we introduce a general framework to generate a sizing system for any kind of 3D motion data applied to face mask design.

Chapter 5

Multilinear wavelets

> “*Animation offers a medium of story telling and visual entertainment which can bring pleasure and information to people of all ages everywhere in the world.*”
>
> – Walt Disney

This chapter introduces a statistical 3D face model that consists of multiple localized, decorrelated multilinear models. In Chapter 4 we used a global multilinear face model to register 3D facial motion sequences. While the global multilinear face model reconstructs the overall face shape well, it is unable to capture fine-scale details. On the other hand, linear wavelet models as described in Section 3.3 are able to reconstruct fine-scale details but they lack a proper handling of facial expressions. In this chapter, we combine the advantages of both methods.

Our model decomposes each shape of a training database into its wavelet coefficients using a discrete wavelet transform, and learns a multilinear model for each coefficient across all training data. This localized statistical model robustly reconstructs 3D face shape from noisy and corrupt data in various expressions. The localized hierarchical structure of the wavelet decomposition makes it possible to capture fine-scale geometric details in a way that is computationally more efficient than can be done with global statistical face models, while retaining robustness to various sources of noise and facial occlusions.

The decoupling of identity and expression variations within the multilinear wavelet model makes it possible to describe motion sequences by varying only the expression while keeping the identity fixed.

5.1 Multilinear wavelet model

This section describes the training of the multilinear wavelet model from a registered and spatially aligned 3D face database containing face scans of d_2 identities with d_3 expressions each. Figure 5.1 depicts the training process of the multilinear wavelet model that is similar to the training of the linear wavelet face model described in Section 3.3.

Let $\mathbf{x}_{ie}$ denote face i in expression e. Instead of computing a global multilinear face model on the vertices of all $\mathbf{x}_{ie}$, we compute many localized, decorrelated multilinear models on the wavelet coefficients of all $\mathbf{x}_{ie}$. First, we decompose each $\mathbf{x}_{ie}$ into its wavelet coefficients $\mathbf{c}_{ie}^{k}$ (middle of Figure 5.1) using a discrete wavelet transform [12] as discussed in Section 3.3.

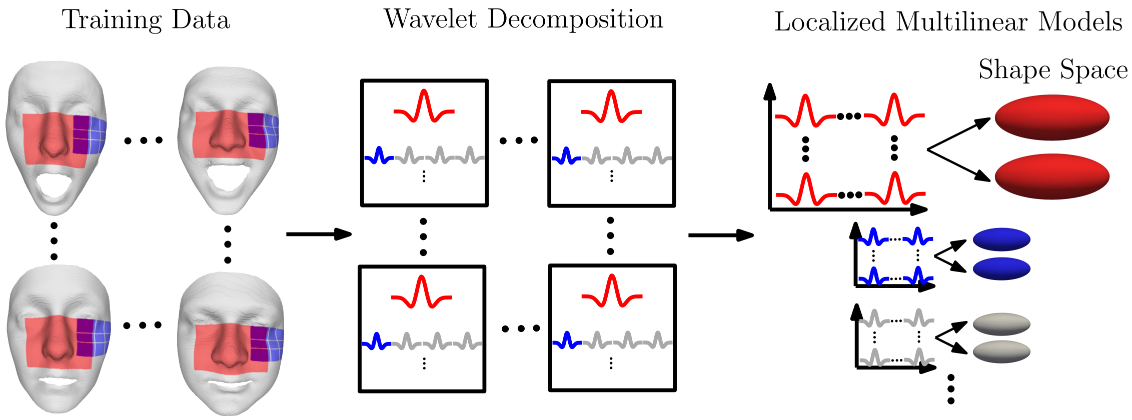


Figure 5.1: Overview of the training. Left: Training data with highlighted impact of the basis function. Middle: Wavelet decomposition of each face of the training data. Right: Corresponding wavelet coefficients and learned multilinear model shape spaces.

Here $k \in \{1, \cdots, n\}$ denotes the index of the wavelet coefficient. Then, we learn for each $\mathbf{c}_{ie}^k$ a multilinear model over all training faces (right side of Figure 5.1). Due to the consistent subdivision subsampling of each $\mathbf{x}_{ie}$, the $\mathbf{c}_{ie}^k$ are in correspondence across the training data. The left side of Figure 5.1 shows the regions influenced by two wavelet coefficients across different identities and different expressions.

We center each $\mathbf{c}_{ie}^k$ by subtracting the mean $\bar{\mathbf{c}}^k = \frac{1}{d_2 d_3} \sum_{i=1}^{d_2} \sum_{e=1}^{d_3} \mathbf{c}_{ie}^k$. We arrange the centered $\mathbf{c}_{ie}^k$ as mode-1 fibers in a 3-mode tensor $\mathcal{C}_k \in \mathbb{R}^{3 \times d_2 \times d_3}$ such that the different identities align with mode 2, and the different expressions with mode 3. HOSVD of $\mathcal{C}_k$ results in

$$\mathcal{C}_k \approx \mathcal{M}_k \times_2 \mathbf{U}_{2,k} \times_3 \mathbf{U}_{3,k}, \tag{5.1}$$

where $\mathcal{M}_k \in \mathbb{R}^{3 \times m_2 \times m_3}$ denotes the multilinear model of coefficient k, and $\mathbf{U}_{2,k} \in \mathbb{R}^{d_2 \times m_2}$ and $\mathbf{U}_{3,k} \in \mathbb{R}^{d_3 \times m_3}$ are the mode-2 and mode-3 factor matrices. Due to the low dimensionality of $\mathcal{C}_k$ in the first mode ($d_1 = 3$), we choose $m_2 = m_3 = 3$.

The surface $\mathbf{f} \in \mathbb{R}^{3n}$ is reconstructed from the multilinear model coefficients $\mathbf{w}_{2,k} \in \mathbb{R}^{m_3}$ for identity, and $\mathbf{w}_{3,k} \in \mathbb{R}^{m_3}$ for expression, as follows. First, each $\mathbf{c}^k$ is reconstructed by Eq. 3.16 as

$$\mathbf{c}^k = \bar{\mathbf{c}}^k + \mathcal{M}_k \times_2 \mathbf{w}_{2,k}^T \times_3 \mathbf{w}_{3,k}^T. \tag{5.2}$$

Then, $\mathbf{f}$ is reconstructed from the $\mathbf{c}^k$ using the inverse wavelet transform given by Equation 3.4.

5.2 Registration of static and dynamic data

This section describes how to fit the multilinear wavelet model to static face scans $\mathbf{s}_i$ or dynamic facial sequences $\mathbf{s}_1, \cdots \mathbf{s}_F$.

5.2.1 Multilinear objective function

We minimize the energy $E : \mathbb{R}^{n(m_2+m_3)} \to \mathbb{R}$

$$E = w_D E_D + w_L E_L + w_R E_R + w_T E_T, \tag{5.3}$$

with respect to the multilinear model coefficients $\mathbf{w}_{2,k}$ and $\mathbf{w}_{3,k}$. The energy E is composed of the data term E_D to fit the model to the scan, the landmark term E_L to fit the model to given landmarks, the regularization term E_R to get a smooth surface, and a temporal smoothness term E_T to avoid jittering of vertices during motion. The influence of each term is controlled by the corresponding weights, namely w_D, w_L, w_R, and w_T. We now describe all terms in more detail.

Data: The data term measures the distance between the model and the data. The data term is

$$E_D = \frac{1}{\sum_{j=1}^{n} b_j} \sum_{j=1}^{n} b_j \left\| \mathbf{v}_j(\mathbf{f}) - \mathbf{nn}_j \right\|^2, \tag{5.4}$$

where $\mathbf{f} \in \mathbb{R}^{3n}$ denotes the reconstruction from the model as detailed in Section 5.1, $\mathbf{v}_j(\mathbf{f})$ denotes the j-th vertex of $\mathbf{f}$, and $\mathbf{nn}_j$ is the nearest neighbor of $\mathbf{v}_j(\mathbf{f})$ in $\mathbf{s}_i$ computed using a point-to-plane distance measure. We use binary weights $b_j \in \{0, 1\}$ to control whether a point is considered for fitting. To lower the influence of outliers, we consider only nearest neighbors that are closer than 10mm. We further discard vertices that are used in E_L corresponding to provided landmarks.

Landmarks: The landmark term measures the distance between given landmarks and the corresponding points of the model. The landmark term is

$$E_L = \frac{1}{L} \sum_{j=1}^{L} \left\| \mathbf{v}_{r_j}(\mathbf{f}) - \mathbf{l}_j \right\|^2, \tag{5.5}$$

where $\mathbf{l}_j \in \mathbb{R}^3$ is the j-th landmark and r_j the index of corresponding vertex on the statistical face model.

Regularization: The regularization term measures the curvature difference of neighboring vertices. The regularization reduces the visibility of patch boundaries and produces smooth surfaces. The regularization term is

$$E_R = \frac{1}{n} \sum_{j=1}^{n} \left\| U^2(\mathbf{v}_j(\mathbf{f})) \right\|^2, \tag{5.6}$$

where $\mathbf{v}_j(\mathbf{f})$ denotes the j-th vertex of shape $\mathbf{f}$. The double-umbrella operator $U^2(\mathbf{p})$ is the discrete bi-Laplacian approximation [79] computed by

$$U^2(\mathbf{p}) = \frac{1}{|N(\mathbf{p})|} \sum_{\mathbf{p}_r \in N(\mathbf{p})} U(\mathbf{p}_r) - U(\mathbf{p}), \tag{5.7}$$

with $U(\mathbf{p}) = \frac{1}{|N(\mathbf{p})|}\sum_{\mathbf{p}_r \in N(\mathbf{p})} \mathbf{p}_r - \mathbf{p}$, and $N(\mathbf{p})$ denotes the set of neighbors of vertex $\mathbf{p}$ within the mesh.

Since the regularization energy affects vertices across patch boundaries, the optimization of E becomes less localized. While a high value of w_R produces a visually smooth surface, it does not accurately fit the surface. Hence, the choice of w_R is a trade-off between getting a smooth surface for high values of w_R and closely resembling $\mathbf{s}_i$ along with a fast optimization for low values of w_R. For a more detailed evaluation see Section 5.3. We choose $w_R = 100$ and $w_R = 0$ throughout our experiments.

Temporal smoothness: The temporal smoothness term measures for motion sequences the distance between corresponding vertices of consecutive frames. The temporal smoothness term is

$$E_T = \sum_{j=1}^{n} \left\| \mathbf{v}_j(\mathbf{f}_i) - \mathbf{v}_j(\mathbf{f}_{i-1}) \right\|^2, \tag{5.8}$$

where $\mathbf{f}_{i-1}$ denotes the reconstruction of the previous frame $\mathbf{s}_{i-1}$, and $\mathbf{f}_i$ denotes the reconstruction of the current frame $\mathbf{s}_i$.

The temporal smoothness term is a trade-off between accurately tracking the facial motion and avoiding jittering. In Chapter 4 we enforce temporal smoothness directly in expression space (Eq. 4.5) since for global multilinear face models, the dimension of the expression space is much lower than the dimension of $\mathbf{f}$ ($m_3 \ll 3n$). For the multilinear wavelet model, instead, the combined dimensions of all expression spaces equal the dimension of the surface. We therefore enforce the temporal smoothness directly in vertex space, since operating in expression space rather than in vertex space does not increase the efficiency of the optimization.

5.2.2 Optimization

The objective function E (Eq. 5.3) is non-linear. Figure 5.2 visualizes the fitting process of the multilinear wavelet model to $\mathbf{s}_i$ with additional landmarks. We first minimize E using the given landmarks to initialize the rigid pose of the model and all $\mathbf{w}_{2,k}$ and $\mathbf{w}_{3,k}$. We then use all vertices of $\mathbf{s}_i$ to refine the fitting. The lower middle of Figure 5.2 illustrates the result after initialization, while the lower right shows the result of the full surface fitting.

We optimize E in a coarse-to-fine manner. First, we minimize E for the multilinear models of the coarse-scale wavelet coefficient to get a coarse approximation of the overall shape. Then, we iteratively refine the result by minimizing E for the models of the finer-scaled wavelet coefficients. As E is analytically differentially with respect to the coefficients $\mathbf{w}_{2,k}$ and $\mathbf{w}_{3,k}$, we minimize E using L-BFGS [92].

As for the global multilinear model, we use a statistical prior to restrict the shape to stay in the learned shape space as described in Section 4.2.1. This statistical prior ensures the robustness of the model to noisy and corrupt data. During optimization, we enforce all $\mathbf{w}_{2,k}$ and $\mathbf{w}_{3,k}$ to stay in hypercubes of side lengths $2c_2$ and $2c_3$ centered at the mean identity coefficients $\overline{\mathbf{w}}_{2,k}$ and the mean expression coefficients $\overline{\mathbf{w}}_{3,k}$.

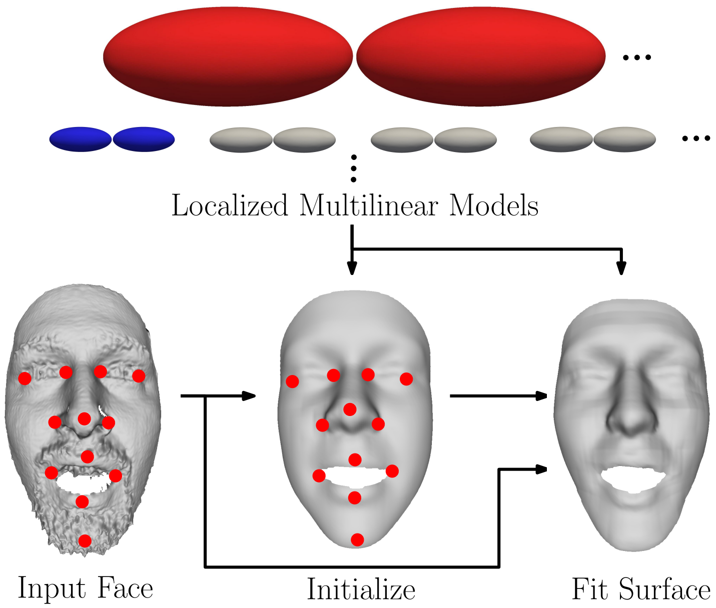


Figure 5.2: Overview of the fitting. Top: Localized multilinear models. Bottom, left to right: input face scan, result after initialization, result of full surface fitting.

Initialization: Since E is non-linear, we require a good initialization for the optimization. To fit the multilinear wavelet model to a 3D face scan, a spatial rigid alignment and initial coefficients $\mathbf{w}_{2,k}$ and $\mathbf{w}_{3,k}$ are needed. We use the provided landmarks for initialization.

We iteratively optimize the rigid alignment and the initialization of $\mathbf{w}_{2,k}$ and $\mathbf{w}_{3,k}$. For the first iteration, we choose $\mathbf{w}_{2,k} = \overline{\mathbf{w}}_{2,k}$ and $\mathbf{w}_{3,k} = \overline{\mathbf{w}}_{3,k}$ for all k. To compute the initial rigid alignment, we reconstruct $\mathbf{f}$ from the model and compute the rigid alignment that minimizes E_L for the given landmarks. The multilinear model coefficients $\mathbf{w}_{2,k}$ and $\mathbf{w}_{3,k}$ are computed by minimizing E for the provided landmarks with regularization ($w_D = w_T = 0$). This deforms the model to closely resemble the given landmarks. We refine the rigid alignment after optimizing each level of the wavelet coefficients.

Since we assume the landmarks to be placed in non-occluded areas, we allow more variations for the initialization and choose $c_2 = c_3 = 1$.

Static registration: After initialization we refine the model to fit the data by minimizing E while setting $w_T = 0$. The nearest neighbors are recomputed after optimizing each level of the wavelet.

To be robust to noise and partial occlusions, we restrict the variations to $c_2 = c_3 = 0.5$ during shape refinement.

Dynamic registration: To register motion sequences, we fit identity and expression coefficients to the first frame of the sequence. For the remaining sequence we keep the identity coefficients fixed, and only optimize for the expression coefficients. This ensures that the shape deformation over time is described by expression variations only. To enforce temporal consistency, we optimize E with temporal smoothness enabled ($w_T = 1$).

Computational complexity: Since the dimension of the search space is the dominant factor during optimization, fitting multiple localized multilinear models is more efficient than fitting one global multilinear model. During optimization, a quasi-Newton optimizer estimates in each iteration the Hessian. For $d = m_2+m_3$ variables, the Hessian is of size $\Omega(d^2)$. This favors solving many small problems rather than one big problem, even if the number of variables increases. Section 5.3 experimentally confirms the increased efficiency.

5.3 Evaluation

This section evaluates the robustness of the multilinear wavelet model registration to noisy or partially occluded data, and the registration quality for facial motion sequences. Due to the lack of ground-truth registration of the face scans, we measure the error as Euclidean distance to the data. All error measures are in millimeters.

Training data: We train the multilinear wavelet model on scans of the BU-3DFE database [139]. We use the template fitting method of Salazar et al. [114], based on provided ground truth landmarks of the database, to register all models.

Test data: The robustness to noisy data is evaluated on 120 face scans (20 identities with up to seven expressions) of the Bosphorus database [116]. The robustness to corrupt data is evaluated on 80 face scans (20 subjects in up to four types of occlusions) of the Bosphorus database. We use the landmarks provided with the database.

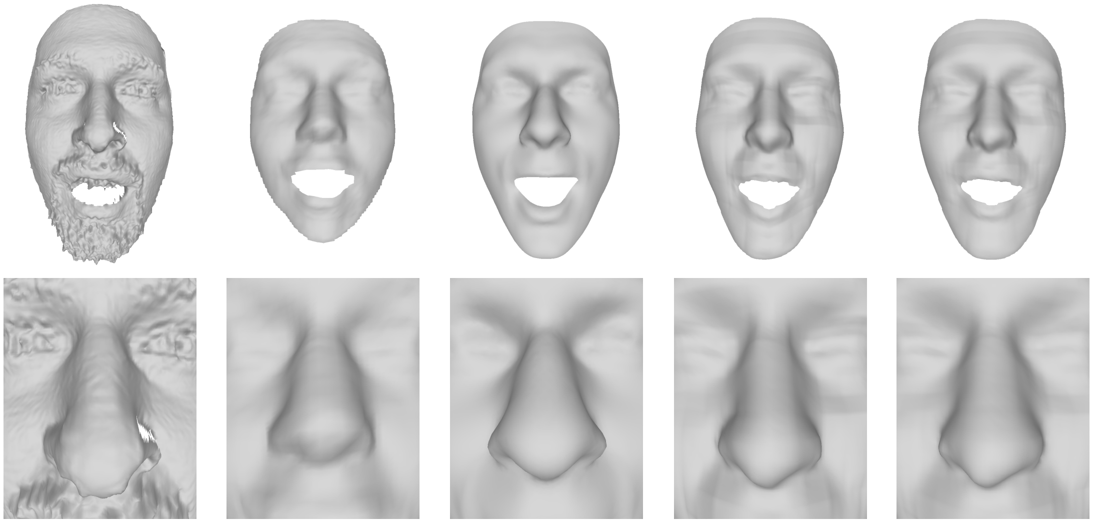
The registration of motion sequences is evaluated for sequences of the BU-4DFE database [138]. We use landmarks automatically predicted by our method as described in Section 4.1.2.

Comparison: We qualitatively and quantitatively compare the multilinear wavelet model to a global multilinear face model and to a linear wavelet model in terms of fitting quality. We use the same training data for all three models. Since the linear wavelet model is unable to handle expression variations, we compute for each expression an individual linear wavelet model. We refer to these multiple linear wavelet models as *local multiple PCA models*. To reconstruct an expression face scan, an expression-specific linear wavelet model is used. The multilinear wavelet model and the global multilinear model use the provided landmarks for fitting.

Performance: Our approach is implemented in C++, using OpenCV [107], ANN [8] and LBFGSB [92]. We evaluate the performance on a 3.3 GHz Intel Xeon E31245 workstation. Fitting the multilinear wavelet model (single-threaded) to a static face scan with about 35000 vertices takes on average 5.37s without regularization ($w_R = 0.0$), and 14.76s with regularization ($w_R = 100$). Compared to this, fitting a global multilinear model takes on average about 2 min, while fitting a linear wavelet model takes about 5 min due to the subspace sampling during optimization.

Fitting the multilinear wavelet model to motion sequences with about 35000 vertices per frame takes on average 4.35s per frame without regularization ($w_R = 0.0$), and 11.14s with regularization ($w_R = 100$).

Reproducibility: We publish the multilinear wavelet model learned from the registered BU-3DFE database and code to fit the model to static input face scans [18].



Scan LMPCA GMM MWM ($w_R = 0$) MWM ($w_R = 100$)

Figure 5.3: Robustness to noisy data. Top: Full face visible. Bottom: Close-up of the nose region. From left to right: Scan, local multiple PCA (LMPCA), global multilinear model (GMM), multilinear wavelet model (MWM) without regularization ($w_R = 0$), and MWM with regularization ($w_R = 100$).

5.3.1 Robustness to noisy data

This section evaluates the ability of our multilinear wavelet model to capture fine-scale details for data with scanner noise, missing data, and facial hair. Since the scans are from 20 identities with up to seven expressions each, the data contain identity and expression variations.

The Figures 5.3 and 5.4 qualitatively compare the local multiple PCA method, the global multilinear model, and the local multilinear model for face scans of three different subjects with three different expressions. Compared to both other models, the multilinear wavelet model captures more fine-scale details. This leads to better reconstructions of nose, mouth, and chin regions. The top two rows of Fig. 5.4 further show that the multilinear wavelet model is able to capture the asymmetric raise of the eyebrow while the global multilinear model only captures the global face shape. Since the expressions of the training data are symmetrically performed, the global multilinear model is unable to capture asymmetric expressions.

Figure 5.3 further shows the effect of the regularization by optimizing the multilinear wavelet model without ($w_R = 0$) and with regularization ($w_R = 100$). The regularization reduces the effect of grid artifacts in the reconstructed face shape that appear between wavelet patches due to the independent optimization.

To quantitatively evaluate the robustness to noisy data, we measure the reconstruction error over all test data. Figure 5.5 shows the median reconstruction error per vertex. For the local multiple PCA models and the global multilinear model 63.2% and 62% of the vertices have a median error < 1mm, compared to 72.4% ($w_R = 0$) and 71.6% ($w_R = 100$) for the multilinear

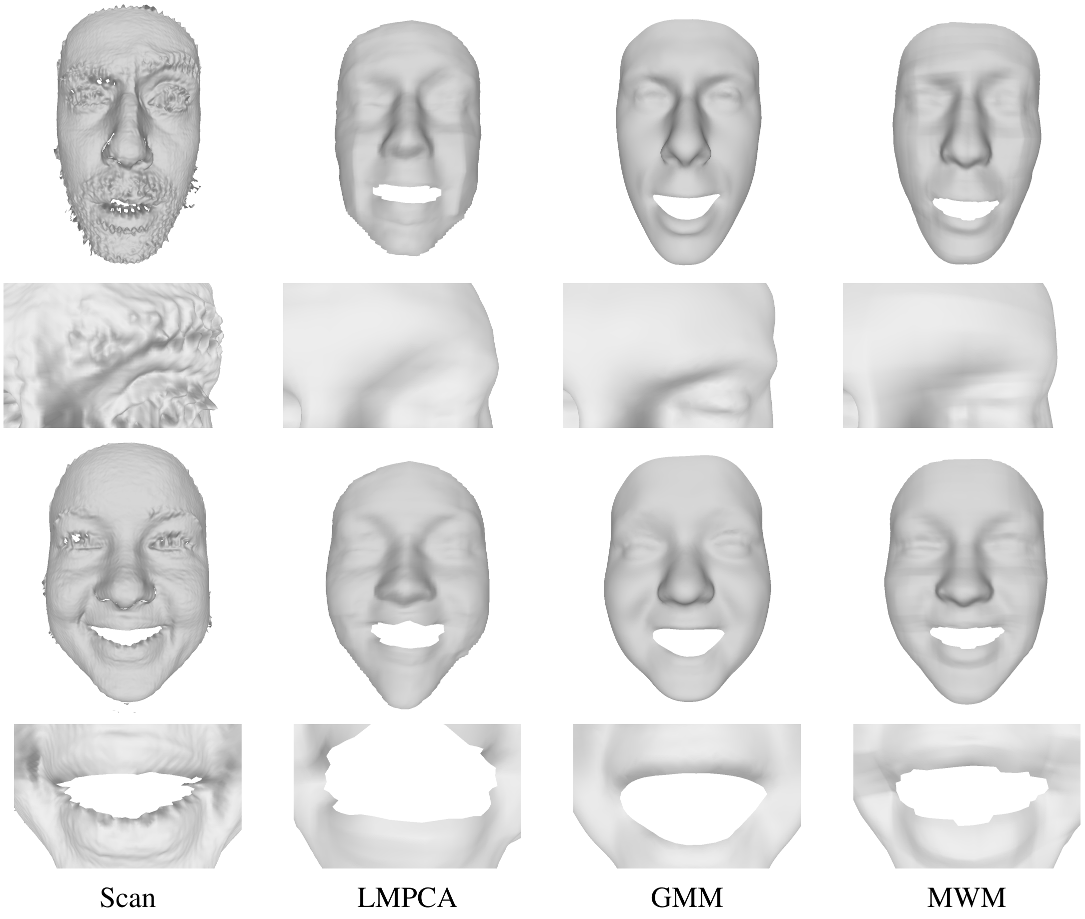


Figure 5.4: Robustness to noisy data in different expressions with full face visible and in close-ups. From left to right: Scan, LMPCA, GMM, MWM.

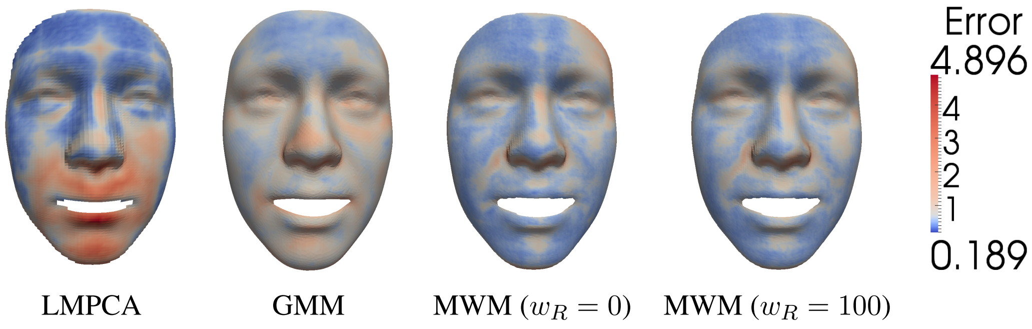


Figure 5.5: Median per vertex reconstruction error for noisy data. From left to right: LMPCA, GMM, MWM without regularization ($w_R = 0$), MWM with regularization ($w_R = 100$).

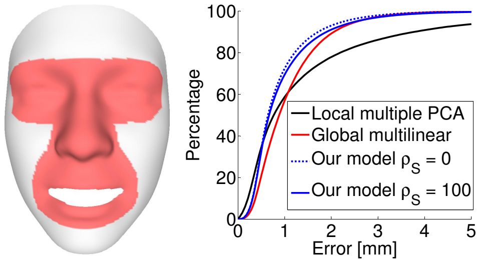


Figure 5.6: Cumulative error for noisy data measured for valid facial regions. Left: Face mask of region used for error measure (red). Right: Cumulative error plot.

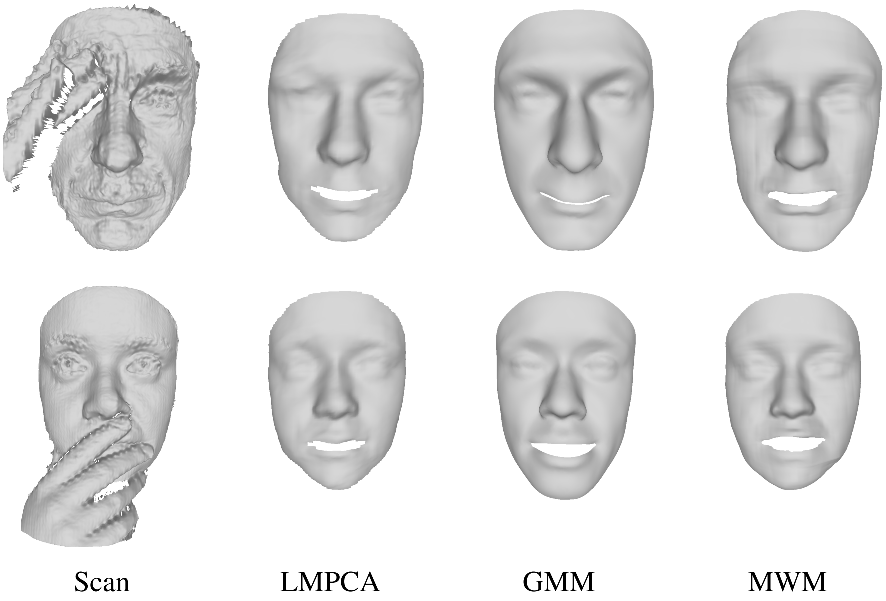


Figure 5.7: Robustness to different partially occluded data. From left to right: Face scan, LMPCA, GMM, MWM.

wavelet model. Figure 5.6 further shows the cumulative error (right) measured for the characteristic facial regions (left). For the local multiple PCA models and the global multilinear model 60.4% and 58.0% of the vertices in the characteristic facial regions are < 1mm, compared to 72.7% ($w_R = 0$) and 70.2% ($w_R = 100$) for the multilinear wavelet model. Hence, while all three statistical models are robust to noise, our multilinear wavelet model reconstructs more fine-scale details, especially in characteristic facial regions like the eye, nose, and mouth regions.

5.3.2 Robustness to corrupt data

This section evaluates the robustness of our multilinear wavelet model to corrupt data. The data corruptions are given in form of 3D face scans with eye, glasses, hair, and mouth occlusions.

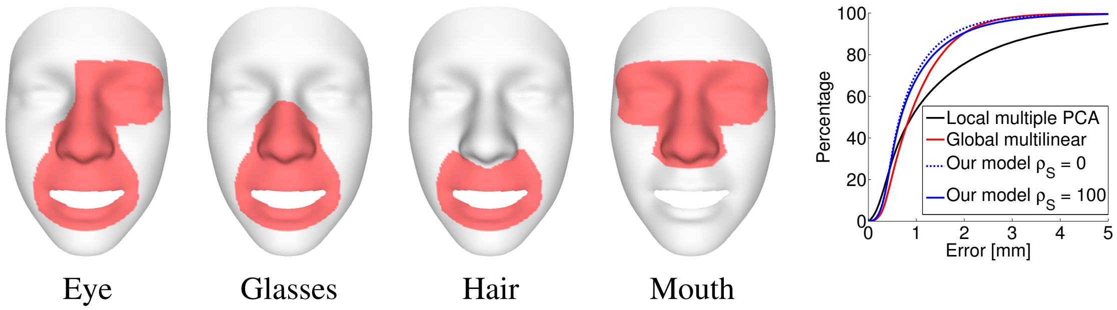


Figure 5.8: Cumulative error for partially occluded data measured for different valid facial regions for eye, glasses, hair, and mouth occlusions. The vertices used for error measure are highlighted in red.

Figure 5.7 qualitatively compares the local multiple PCA method, the global multilinear model, and the multilinear wavelet model for face scans of two different subjects, one with an eye occlusion, one with a mouth occlusion. All three statistical face models produce a plausible face shape and are hence robust to the facial occlusions. Compared to both other methods, the multilinear wavelet model better reconstructs facial details in non-occluded regions (e.g. at the nose and the chin in the top row, and at the nose in the bottom row).

To quantitatively evaluate the robustness to corrupt data, we measure the reconstruction error for all test data. Since the distance to the data is only a valid measure in non-occluded areas, we define for each type of occlusion the non-occluded area. The left side of Figure 5.8 highlights these regions for eye, glasses, hair, and mouth occlusions in red. The right of Figure 5.8 shows the cumulative error measured in these non-occluded areas only. The multilinear wavelet model has the lowest error in these regions compared to both other statistical face models. Hence, our model better represents fine-scale details even in the presence of heavy facial occlusions.

5.3.3 Registration of motion data

This section evaluates the registration of motion sequences. Figure 5.9 shows two motion sequences performing different expressions sampled at four frames. Our multilinear wavelet model accurately reconstructs the face shape for both sequences and tracks the facial expression. Since the landmarks are automatically predicted, the registration process of the motion sequences is fully automatic.

5.4 Summary

This chapter presented a new statistical 3D face model that consists of multiple decorrelated multilinear models. This model allows robust reconstruction of the 3D face shape from noisy and corrupt face scans. In contrast to existing statistical face models, our model handles facial expressions and better reconstructs fine-scale geometric details while retaining robustness to noise and partial occlusions. The multi-scale nature of the wavelet decomposition we used leads to a more efficient optimization. The decoupling of identity and expression variations allows efficient tracking of facial motion sequences.

The decorrelated localized structure allows the fitting for each level of the wavelet coefficients to be parallelized, and an optimized GPU implementation could potentially run in real time. A detailed real-time tracker has various applications as mentioned in Section 2.5.

To obtain a high-quality wavelet face model, the quality of the registration of the training data is essential. Establishing a dense correspondence for databases of 3D human faces of different identities performing multiple expressions is challenging. Existing methods that aim to register 3D faces (see Section 2.3) introduce drift in the registration. To obtain a high-quality registration that suits the needs of statistical face models, we introduce a multilinear correspondence optimization framework in the following chapter.

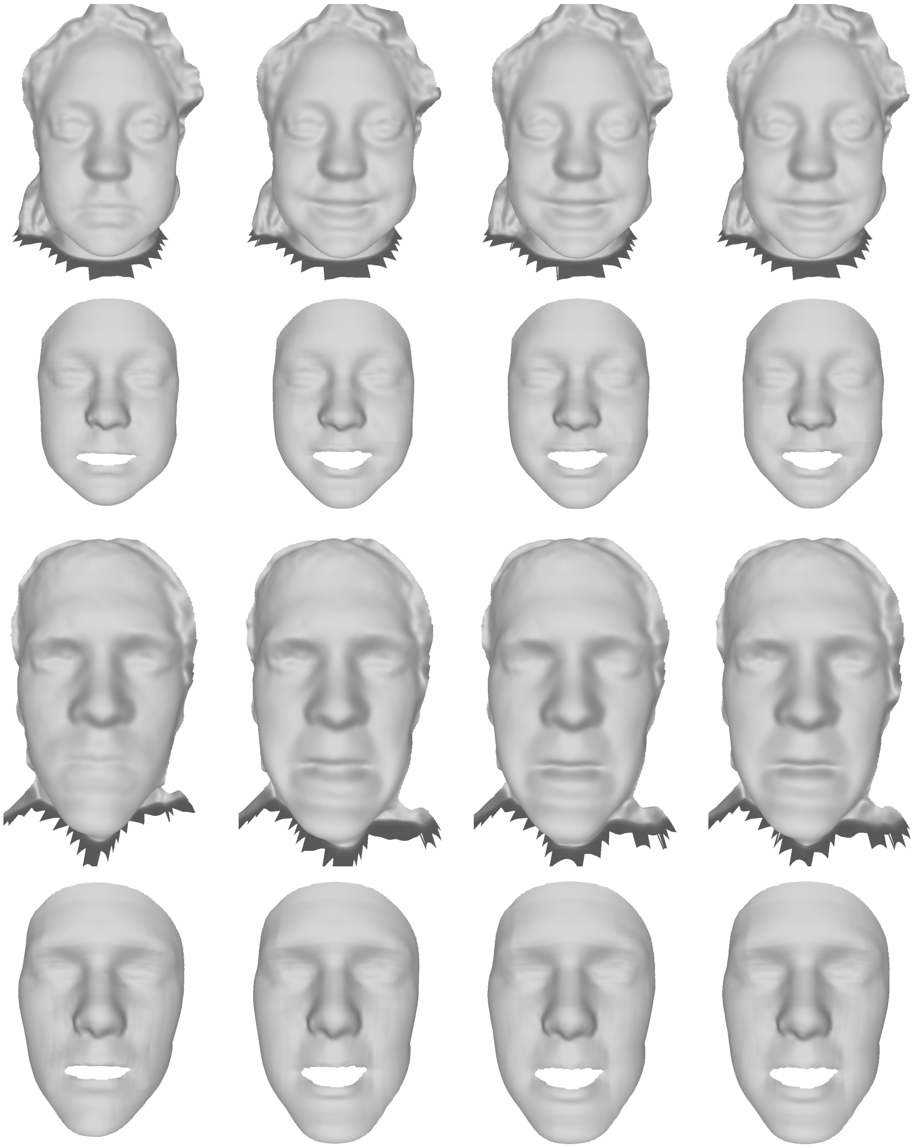


Figure 5.9: Registration results for facial motion sequences performing different expressions. Top: Happy expression. Bottom: Fear expression.

Chapter 6

Registration optimization

"*You can never solve a problem on the level on which it was created.*"

– Albert Einstein

This chapter introduces a method to jointly optimize a multilinear model and the registration of a 3D face database in a groupwise fashion. As discussed in Chapter 2, to compute statistics of a class of shapes requires all shapes to be in correspondence. Given a good registration, a statistical face model can be learned. Statistical face models can be used to reconstruct the 3D geometry from noisy or partially occluded face scans (see e.g. our review [29]) and are therefore directly applicable for registration. Summing up, this is a chicken-and-egg problem: given a good registration, a statistical model can be learned, and given a representative statistical model, a good registration can be computed. This motivates the formulation of the statistical face model learning as a groupwise optimization framework that aims to learn a statistical face model while at the same time optimizing the training data.

Since the variations in databases of human faces from different identities performing different expressions cannot be modeled well using a linear space, the existing methods are not suitable for optimizing the correspondence of human faces. As shown in the two previous chapters, human faces in various expressions can be modeled well using a multilinear model.

This motivates us to propose a fully automatic groupwise correspondence optimization approach for multilinearly distributed 3D face data. The correspondence is optimized based on the MDL principle, which leads to a sparse multilinear model. A key advantage of extending MDL to multilinear models is a reduced parameter space, which can be optimized more efficiently and leads to correspondences of higher quality than existing PCA-based optimization methods.

6.1 Groupwise correspondence optimization

This section introduces the concept of groupwise correspondence optimizations and describes our approach for multilinearly distributed data. Given a set of shapes in correspondence, groupwise correspondence optimization minimizes an objective function that measures the quality of the correspondence depending on all shapes. Using a statistical model that describes the variation of the shapes, the objective function measures favorable properties of the model.

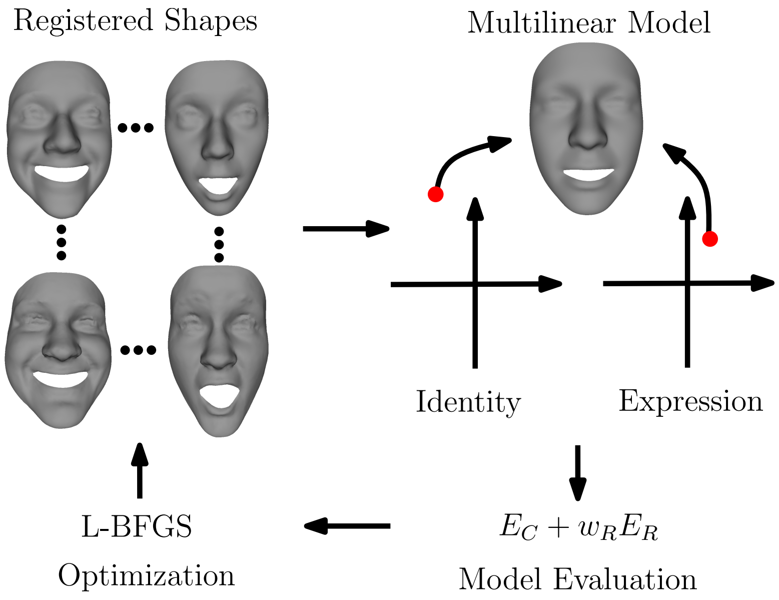


Figure 6.1: Overview of the iterative multilinear registration.

For PCA models, Kotcheff and Taylor [81] choose the objective function to be the determinant of the covariance matrix, which explicitly favors the induced linear statistical model being compact. The compactness of a linear statistical model can be maximized by minimizing the variability of the model, measured by the trace of the covariance matrix.

Compactness (see Section 3.2) measures the variability captured by a model. A compact model can describe instances of a given dataset with the minimum number of parameters and has minimal variance. For models of different compactness that describe the same data, the model with higher compactness and hence lower variance is favorable. It has been shown that minimizing the variance of a PCA model performs similarly to information-theoretic approaches that aim at minimizing the description length of the model [56].

Inspired by these previous works, we develop the first MDL-based optimization approach for multilinear models. This extension is challenging because the notion of compactness needs to be extended to multilinear models, where optimal tensor approximation is NP-hard [63]. For 3D face data, a further challenge arises from manifold boundaries. Figure 6.1 gives an overview of our multilinear optimization approach. Given a set of 3D faces of d_2 different identities performing d_3 different expressions with an initial correspondence, we iteratively optimize the correspondence. We compute a multilinear model on the registered data, and iteratively improve the model. In each iteration, the quality of the model is measured using a groupwise objective function (Section 6.1.1). The registered shapes are represented using a continuous parametrization (Section 6.1.2), and the objective function is optimized in parameter space with a quasi-Newton method (Section 6.1.3).

6.1.1 Multilinear objective function

Our groupwise objective function consists of two parts: a compactness energy E_C, and a regularization energy E_R. We therefore aim to minimize

$$E = E_C + w_R E_R, \tag{6.1}$$

where w_R is a weight that controls the influence of the regularization. We now describe both terms in more detail.

Compactness: The compactness of a multilinear model can be measured as the percentage of data variability captured in the first k components of each mode, where $k = 1, \ldots, \max(d_2, d_3)$. Compactness is maximized by a sparse model that captures all of the variability in few components. To encourage a sparse model, we introduce an energy on the variability of the identity and expression subspaces. Like Kotcheff and Taylor [81], we choose a log-sum penalty function, as log-sum functions are known to encourage sparsity by heavily punishing small values [32]. That is, we aim to minimize

$$E_C = \frac{1}{d_2}\sum_{i=1}^{d_2} \ln(\lambda_i^{(2)} + \delta_2) + \frac{1}{d_3}\sum_{i=1}^{d_3} \ln(\lambda_i^{(3)} + \delta_3), \tag{6.2}$$

where $\lambda_i^{(n)}$ denotes the i-th eigenvalue of the mode-n covariance matrix. Small regularization constants δ_n are used to avoid singularities of E_C for vanishing eigenvalues. Equivalent to HOSVD, the mode-2 and mode-3 covariance matrices are computed as $\frac{1}{d_3}\mathbf{X}_{(2)}\mathbf{X}_{(2)}^T$ and $\frac{1}{d_2}\mathbf{X}_{(3)}\mathbf{X}_{(3)}^T$.

The energy E_C is minimized by moving points within the continuous surface of each shape. Since the computation of the covariance only considers a discrete number of points instead of the continuous surface, E_C can be minimized by moving points away from complex geometric regions with high variability.

Regularization: To avoid undersampling in these regions, Davies et al. [43] approximate the integral of the continuous covariance matrix by weighting the points by their surrounding surface area. Since this does not always prevent the undersampling [56], as done in Burghard et al. [31], we use a regularization within the objective function. As in Chapter 5, the regularization term for each shape is a bi-Laplacian of the form

$$E_R = \frac{1}{n}\sum_{k=1}^{n} \left\| U^2(\mathbf{v}_k(\mathbf{x})) \right\|^2, \tag{6.3}$$

where $\mathbf{v}_k(\mathbf{x})$ denotes the k-th vertex of shape $\mathbf{x}$. The double-umbrella operator $U^2(\mathbf{p})$ is the discrete bi-Laplacian approximation [79] as described in Section 5.2.1.

Despite using the same regularizer as in Section 5.2.1, the effect of E_R on the objective function during optimization is different. In Section 5.2.1, the objective function is optimized in vertex space. Optimizing E_R on the object's surface encourages the resulting surface to be visually smooth. Optimizing the objective function in the 2D parameter domain, as done in this section, encourages the points to be regularly distributed over the mesh and prevents fold-overs.

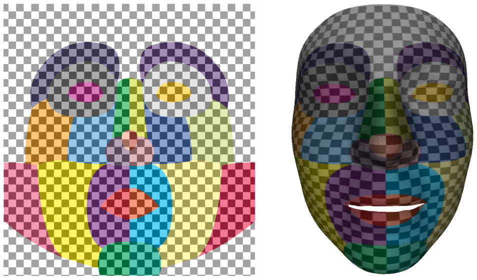


Figure 6.2: Initial surface parametrization of the 3D face template. Left: 2D parameter domain. Right: 3D parametrization.

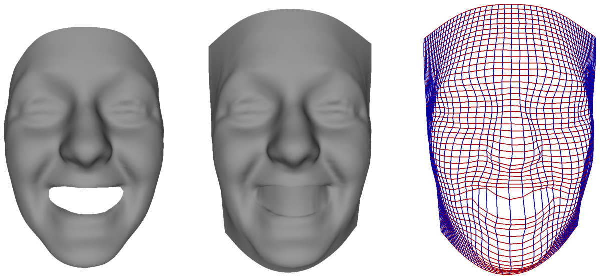


Figure 6.3: Parametrization for one shape. Left: initialization. Middle: thin-plate spline. Right: (u, v)-parameter lines.

6.1.2 Parametrization

The registration is optimized by moving points in the surface of each face. Since the surface of the face is 2-dimensional, moving points within the surface can be done by re-parametrization. This requires an initial parametrization together with a continuous mapping from parameter space to the surface of each face. We compute an initial registration for a database of 3D faces using template fitting, and additionally unwrap the 3D template mesh in 2D parameter space to compute an initial discrete parametrization with parameters $\mathbf{t}_i \in \mathbb{R}^2$. The embedding in 2D is chosen to minimize distortions of angles and areas. Each parameter $\mathbf{t}_i$ is mapped to the mesh vertex $\mathbf{v}_i = (x_i, y_i, z_i) \in \mathbb{R}^3$. Figure 6.2 visualizes the initial parametrization in 2D parameter space (left) and mapped on the 3D surface (right). Due to the full correspondence of all face shapes, this discrete parametrization is the same for all shapes of the database.

With this discrete embedding in parameter space, a continuous mapping Φ is computed that maps parameters $\boldsymbol{\alpha} = (u, v) \in \mathbb{R}^2$ into the surface of the shape. A thin-plate spline [45] defines this mapping, computed as

$$\Phi(\boldsymbol{\alpha}) = \mathbf{c} + \mathbf{A}\boldsymbol{\alpha} + \mathbf{W}^T(\sigma(\boldsymbol{\alpha} - \mathbf{t}_1), \ldots, \sigma(\boldsymbol{\alpha} - \mathbf{t}_n))^T, \tag{6.4}$$

where $\mathbf{c} \in \mathbb{R}^3$, $\mathbf{A} \in \mathbb{R}^{3\times 2}$, and $\mathbf{W} \in \mathbb{R}^{n\times 3}$ are the parameters of the mapping, and where

$\sigma : \mathbb{R}^2 \to \mathbb{R}$ is the function

$$\sigma(\mathbf{h}) = \begin{cases} \|\mathbf{h}\|^2 \log(\|\mathbf{h}\|) & \|\mathbf{h}\| > 0, \\ 0 & \|\mathbf{h}\| = 0. \end{cases} \tag{6.5}$$

The surface of Φ interpolates all vertices of the shape ($\Phi(\mathbf{t}_i) = \mathbf{v}_i$) and gives the surface with the minimum bending energy. Figure 6.3 shows one initially registered shape (left) together with the computed continuous thin-plate spline visualized as densely approximated mesh (middle) and (u, v)-parameter lines (right). The evaluation of Φ at parameters $\boldsymbol{\alpha}$, where u (respectively v) is fixed and v (respectively u) is varied by a fixed discrete step size, gives one (u, v)-parameter line. While the spline interpolates the geometry of the initial shape, it gives a reasonable extrapolation of the shape beyond the outer border of the face.

6.1.3 Optimization

The objective function E in Equation 6.1 is non-linear. Due to the choice of the parametrization, E is analytically differentiable with respect to $\boldsymbol{\alpha}$. Appendix A.1 gives the full analytical gradient. We minimize E using L-BFGS [92]. These linear constraints allow for each vertex in parameter space to specify a valid rectangular area.

Boundary constraints: For meshes with boundaries, E_C is minimized if the entire surface collapses into a single point. Hence, boundary conditions need to be enforced. Face shapes have two boundaries, an inner boundary at the mouth and an outer boundary at the end of the acquired scan. Since landmarks are used during the initial registration, the inner boundary at the mouth is registered well. To avoid points that move from the lower to the upper lip or vice versa, we fix the points in the 1-ring neighborhood of the mouth boundary during optimization. Since the outer boundary is not registered well, as scans in the database are cropped inconsistently, we allow limited movement for points in the 1-ring neighborhood of the outer boundary. Specifically, the movement is restricted to at most 20 mm.

Optimization schedule: Optimizing for the parameters of all shapes at the same time is not feasible for a large population of shapes due to the large number of parameters $(d_2 d_3 2n)$. Instead, we only optimize the parameters of each shape individually, as proposed by Davies et al. [43, Chapter 7.1.1]. This optimization is performed for all shapes of the database during each iteration. Note that E still depends on all shapes for this shape-wise optimization, and the method therefore still optimizes the groupwise correspondence. To avoid bias towards any shape, the order of the shapes is randomly permuted for each iteration step. Since the rigid alignment of the shapes depends on the correspondence, during optimization of one shape, the alignment is updated after a few optimization steps.

Computational complexity: The computational complexity is $O(nd_2^2 d_3 + nd_2 d_3^2)$ of one optimization step (see Appendix A.2 for details). As shown in the following section, our approach is significantly more efficient than existing PCA-based MDL approaches.

While the multilinear correspondence optimization is computationally more efficient than previous linear methods, due to the groupwise objective function, the computational complexity is still high. Our experiments show that only a low number of iterations are necessary to get significant improvements. Note that the registration can be seen as pre-processing that only needs to be done once. Application to larger datasets would require the use of a compute cluster to exploit the full potential of the parallelizability of the method (especially the gradient computation).

6.2 Evaluation

This section evaluates three different tensor decompositions and our model optimization approach.

Data: For evaluation, we use models of the BU-3DFE [139] and Bosphorus [116] databases. A more detailed explanation of both databases is given in Section 3.1. Since both databases are acquired with different scanner systems, the resulting scans have different resolution and noise characteristics. We register both databases with a template fitting method [114] using the provided landmarks.

For BU-3DFE we use 50 randomly chosen identities with seven expressions: neutral and the highest intensity level of each expression. For Bosphorus we use all 65 identities that are present in all seven expressions. In the following, we call these subsets the *BU-3DFE set* and the *Bosphorus set*, respectively.

Model quality: We quantitatively evaluate the quality of the optimization with the widely-used measures compactness, generalization and specificity extended to the multilinear case as described in Section 4.2.2.

Reproducibility: To facilitate evaluating the model for different applications, we make our optimization code and the optimized statistical model available [23].

6.2.1 Tensor decompositions

We evaluate the different tensor decomposition methods described in Section 3.4, namely HOSVD, HOOI, and a Newton-Grassmann optimization approach, by fitting the resulting multilinear models to unseen 3D face scans. We use the code by Nigmetov [105]. For this, we use a 10-fold cross-validation on the registered BU-3DFE scans. We split the database randomly into ten groups, each with the same ratio of male and female subjects, where all scans of one identity belong to the same group. The error is measured as the distance between a vertex in the fitting result and its closest point in the face scan. The error distribution of all three methods is nearly identical. The median vertex error is 1.145 mm for HOSVD, 1.144 mm for HOOI, and 1.144 mm for the Newton Grassmann method. Since all methods perform almost the same, we compute the decomposition with HOSVD in the following as it is the simplest method.

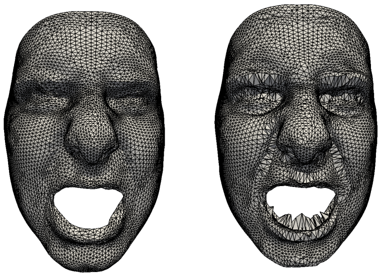


Figure 6.4: Artifacts obtained by optimizing E_C without regularization ($w_R = 0$). Left: initial registration. Right: result.

6.2.2 Influence of regularization

This section evaluates the influence of the regularization E_R on the BU-3DFE set. The optimization is performed twice, once optimizing only E_C without E_R and once optimizing only E_R without E_C. As discussed in Section 6.1.1, the regularizer is needed to avoid undersampling in regions with high variability and fold-overs. Figure 6.4 shows the result for one face after only five iterations of optimizing E_C. When minimizing only E_C, the optimization moves points away from the eyebrows and around the nose, resulting in sparsely sampled regions. Furthermore, fold-overs at the mouth cause visual artifacts. Optimizing E_R leads to regularly sampled meshes. However, E_C increases in this case. Minimizing E is therefore a trade-off between getting a compact model and a regular mesh structure. In the following, we empirically choose $w_R = 0.5$.

6.2.3 Influence of initialization

This section evaluates the robustness to noise in the initialization. State-of-the-art registration methods for faces, as used for the initialization of our method, are able to fit the facial surface well with sub-millimeter accuracy, but the result is likely to contain drift within the surface. To simulate noise with regard to these methods, we use the initial parametrization and add two different levels of noise in the parameter domain. The parameter values of each shape of the BU-3DFE set are disturbed by random Gaussian noise. Since the 1-ring neighborhood of the mouth boundary is fixed during optimization, these vertices are left without noise. For both noise levels we choose noise with mean zero and standard deviation f times the average 3D edge length. For the lower noise level we choose f to be 0.25, and for the higher 0.75, respectively.

The optimization is performed on the BU-3DFE set, initialized with the noisy registration. The top of Figure 6.5 shows an example of the database without noise (left), the lower level of noise (middle) and the higher level of noise (right). The average 3D vertex distance of the initial shapes to the noisy shapes over the entire database is 1.11 mm for the lower and 2.50 mm for the higher noise level.

Adding random noise within the surface to each vertex increases the variance in 3D po-

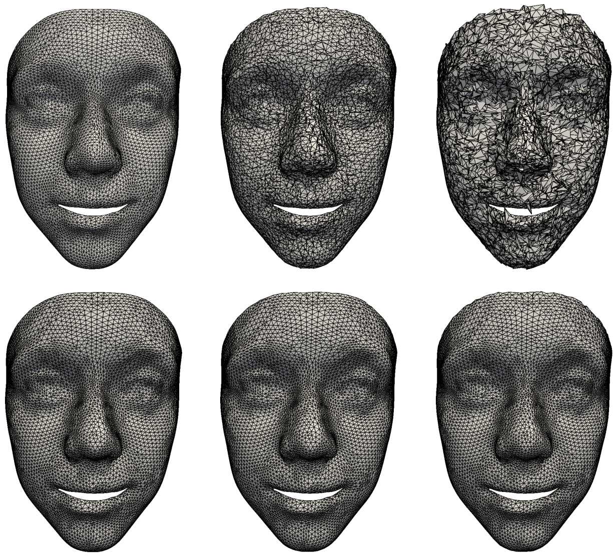


Figure 6.5: Noise example of the database before (top) and after (bottom) optimization. Left to right: no, low, and high noise.

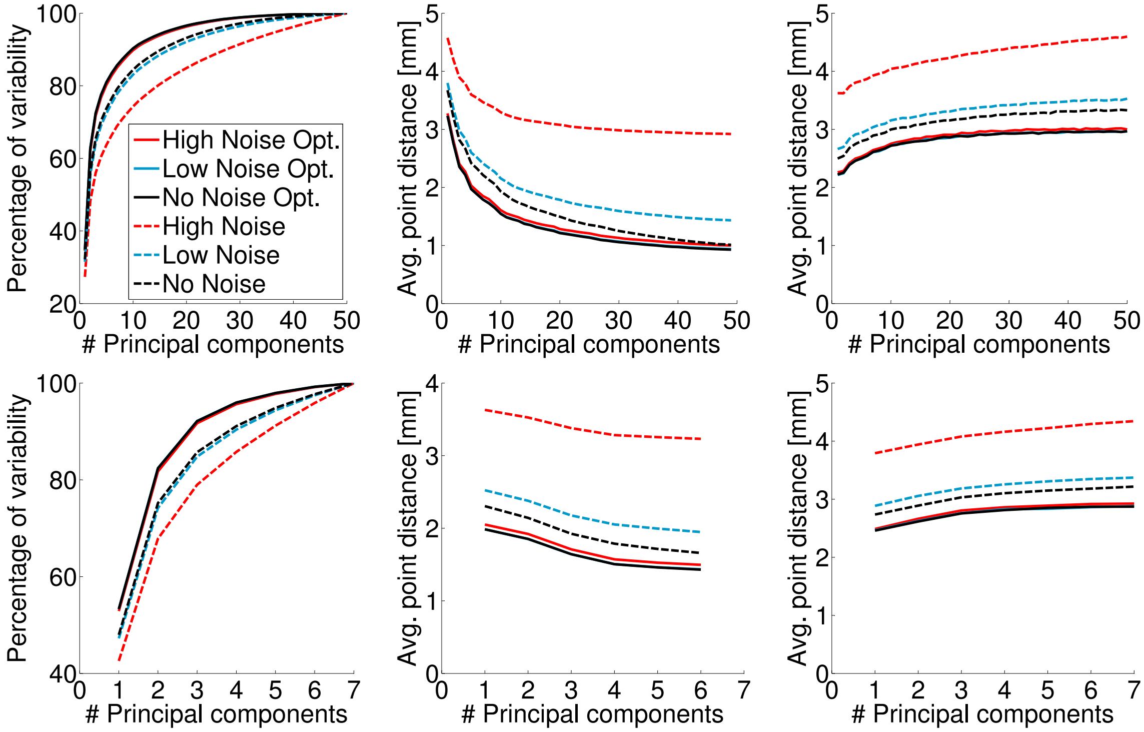


Figure 6.6: Influence of the initialization for different levels of noise. Left: compactness. Middle: generalization. Right: specificity. Top: identity mode. Bottom: expression mode.

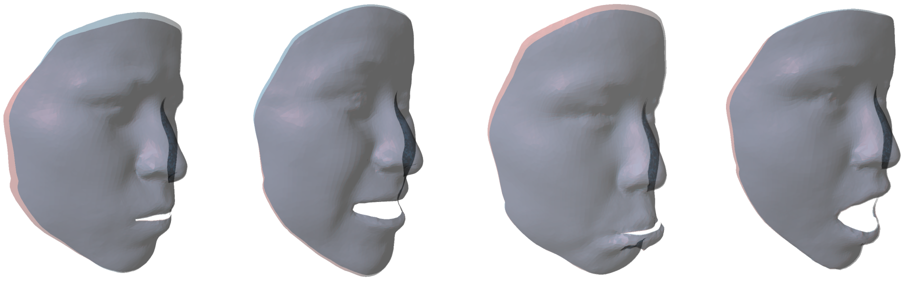


Figure 6.7: Visual comparison of template fitting [114] (red) and our result (blue) for one subject in four expressions (overlap in gray).

sitions and therefore increases the variability of the data. As expected, Figure 6.6 shows that the compactness of identity mode and expression mode decreases with increasing noise, since the multilinear model captures less variability with the same number of components. Further, the multilinear model becomes less general and less specific. After 15 iterations, the average compactness increases by 3.8% for the low noise level, and by 8.7% for the high noise level, respectively. The average generalization error decreases by 0.58 mm and 1.65 mm for the low and high noise level; the average specificity decreases by 0.43 mm and 1.26 mm for the low and high noise level. After optimization, the model quality for both levels of noise is comparable to the optimization of the data without noise. Hence, our optimization method effectively reduces variability caused by drift.

6.2.4 Comparison

This section compares our approach to two state-of-the-art registration methods for 3D faces based on template fitting [114] and PCA-based groupwise correspondence [43].

Template fitting: We compare our optimization to template fitting on the BU-3DFE and Bosphorus sets. For the two subsets, Figures 6.8 and 6.9 show the compactness, generalization, and specificity for template fitting and after 15 iterations of the multilinear optimization. For the BU-3DFE set, the average compactness increases by 3.0%, and the average generalization and specificity decrease by 0.25 mm and 0.32 mm, respectively. For the Bosphorus set, the average compactness increases by 1.7%, and the average generalization and specificity decrease by 0.15 mm and 0.16 mm, respectively.

Figure 6.7 visually compares the template fitting (red) to our result (blue) for one subject of the BU-3DFE set. Before optimization, the shape of the outer boundary differs. The optimization decreases the face area for the first and fourth expressions at the cheek, for the second expression at the jaw, and for the third expression at the forehead. Expressions one, two, and three are extended at the forehead. After 15 iterations, the outer boundaries are similar.

To demonstrate the ability of our method to optimize over large sets of shapes, we consider a second subset of the Bosphorus database consisting of 39 identities performing 26 action units each, leading to a total of over 1000 shapes. To keep 95% of the data variability after

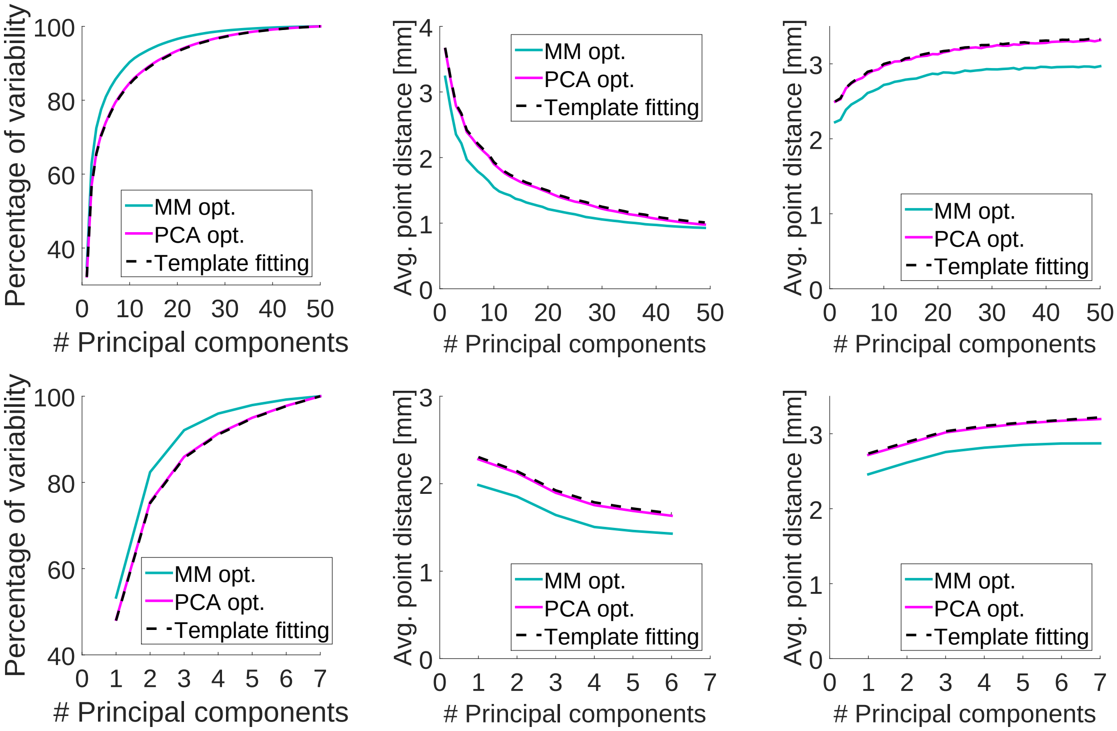


Figure 6.8: Comparison of template fitting [114], PCA optimization [43] (PCA opt.) and multilinear model optimization (MM opt.) on BU-3DFE set. Left: compactness. Middle: generalization. Right: specificity. Top: identity mode. Bottom: expression mode.

template fitting, a total of 27 components are necessary, while after 15 iterations of our optimization, 20 components suffice. As for the other subsets, generalization and specificity also improve after optimization. To the best of our knowledge, this is the first time a registration optimization based on MDL has been applied to such a large set of shapes.

For all three datasets the model improves significantly during optimization, leading to a more compact model with improved generalization and specificity.

PCA: For brevity, we abbreviate PCA optimization as *PCA opt.* and our method as *MM opt.* during the discussion of the comparison. We start by comparing the computational complexity of the two methods. In Appendix A.2, we show that one optimization step for PCA opt. has complexity $O(nd_2^2d_3^2)$, while one optimization step of MM opt. has complexity $O(nd_2^2d_3 + nd_2d_3^2)$. For the BU-3DFE set our non-optimized implementation takes about 16.2h for MM opt. and about 21.5h for PCA opt. for one iteration when executed on a standard PC.

Figure 6.8 quantitatively compares PCA opt. and MM opt., both after 15 iterations. While MM opt. gives significant improvements, PCA opt. only slightly improves the correspondence. For small subsets PCA opt. gives significant improvements within few iterations. Our experiments suggest that for an increasing number of shape space parameters, an increasing number of iterations is required. Since MM opt. models identity and expression indepen-

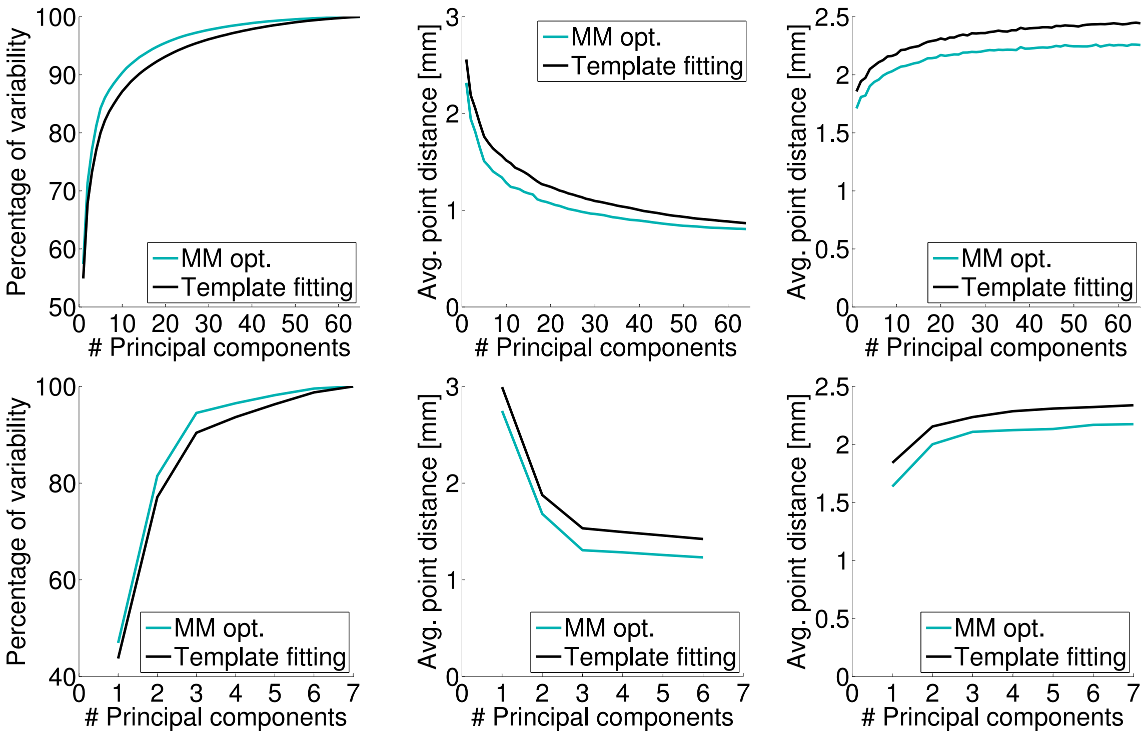


Figure 6.9: Comparison of template fitting [114] and MM opt. on Bosphorus set. Left: compactness. Middle: generalization. Right: specificity. Top: identity mode. Bottom: expression mode.

dently, the number of shape space parameters is $d_2 + d_3$, while for PCA opt. the number of shape space parameters is $d_2 d_3$.

Hence, our method gives better improvements after the same number of iterations and is computationally faster than existing linear optimization methods.

6.3 Summary

This chapter presented the first method for multilinearly distributed data that jointly improves a given registration and a multilinear model. A continuous representation of each shape allows the registration to be optimized with a quasi-Newton method. We have evaluated our method on scans of two databases and have demonstrated that our method is robust to noise in the initial registration. A key advantage of our approach over existing linear MDL methods is its increased computational efficiency, which makes it possible for the first time to apply an approach based on MDL to databases containing over 1000 shapes. We have shown that using the efficient HOSVD method to compute the multilinear model performs similarly when reconstructing unseen face data to more elaborate tensor decompositions. To facilitate experiments for different application scenarios, we make our optimization code and the optimized statistical model available.

Our method is generally applicable to other classes of multilinearly distributed data. The geometry of the shapes can contain no or multiple holes as long as the boundaries of the holes are constrained. The regularization energy prevents fold-overs around these holes. Furthermore, the extension of our method to more modes is straightforward, e.g. for faces to associate the fourth mode with viseme or age.

Our proposed method optimizes the correspondence by re-parametrizing the shapes guided by the optimization of a multilinear compactness objective function. This re-parametrization requires a continuous representation of the surface for each shape. While any kind of continuous mapping can be used, we establish this by a thin-plate spline. For other continuous mappings, the gradient changes, and therefore depending on the mapping (e.g. for mappings without analytical gradients) a different optimization must be used.

Computing this continuous surface mapping assumes the original face scans to be regularly densely sampled with points that are within the surface of the scan. To get this sampling, any existing template fitting method can be used. For face scans with partial occlusions or strong distortions, template fitting methods fail, since they are unable to estimate the real face surface in these regions. To optimize the registration for scans with strong distortions, we would either need another initialization that gives a reasonable surface estimation within the occluded and noisy regions (e.g. by using the multilinear wavelet model proposed in Chapter 5), or the optimization must be allowed to leave the surface of the disturbed scan guided by the underlying multilinear model as done in the following chapter.

While our approach is purely geometry based, additionally using texture information is known to be helpful to establish an anatomically meaningful correspondence, as a high-quality texture makes it possible to establish correspondence between multiple scans of the same identity by using freckles and pores as features [26]. Texture information could be used throughout the optimization by adding an additional term to E that measures the difference between these texture features.

Using a multilinear compactness term requires the full Cartesian product of all facial attributes (i.e. all identities need to be present in all expressions), and all scans must be in semantic correspondence (i.e. the expressions must be correctly labeled). To overcome these limitations, the following chapter introduces a framework to robustly learn a multilinear model from 3D face databases with missing data, corrupt data, wrong semantic correspondence, and inaccurate vertex correspondence.

Chapter 7

Robust multilinear model learning

"An experiment is a question which science poses to Nature and a measurement is the recording of Nature's answer."

– Max Planck

This chapter introduces a framework to robustly learn a multilinear model from 3D face databases with missing data, corrupt data, wrong semantic correspondence, and inaccurate vertex correspondence. The methods to learn a multilinear face model as used throughout Chapters 4, 5, and 6 degrade if not every person is captured in every expression, if face scans are noisy or partially occluded, or if expressions are erroneously labeled.

Missing data occur if not all available identities are present in all expressions, i.e. some identities are only captured in a subset of the expressions. Missing data are caused by subjects being unable to perform certain expressions spontaneously, or by the extension of an existing database by additional expressions with some subjects being unavailable for further scanning. Corrupt data arise if the facial geometry is noisy or partially occluded. Wrong semantic correspondences arise if a subject has difficulties in performing specific expressions correctly and mixes up certain expressions, or due to erroneous classifications of the performed expressions. These limitations impose requirements on the training data that disqualify large amounts of available 3D face data from being usable to learn a multilinear model.

If a multilinear face model is given, it is able to complete missing data (e.g. [35]), reconstruct corrupt data (e.g. Chapter 5), to label expressions (e.g. [102]), or to optimize correspondence as described in Chapter 6, all of which is necessary to build up a database that fulfills the needs of a multilinear model. In the spirit of the groupwise multilinear correspondence optimization method from Chapter 6, this motivates us to formulate the multilinear model learning as a groupwise optimization framework that aims to learn a multilinear face model while at the same time correcting the data.

In this chapter, we show that our framework achieves a data completion accuracy that is comparable to state-of-the-art tensor completion methods; our method reconstructs corrupt data more accurately than state-of-the-art methods, and improves the quality of the learned model significantly for erroneously labeled expressions.

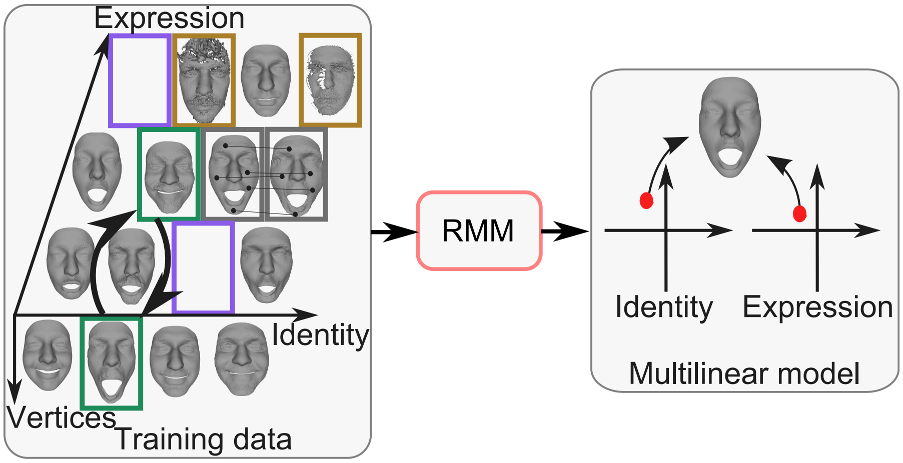


Figure 7.1: Overview of our RMM learning framework that is robust to missing data (purple), corrupt data (brown), wrong semantic correspondence (green), and inaccurate vertex correspondence (gray).

7.1 Groupwise multilinear model learning

This section describes our robust multilinear model (RMM) learning framework as outlined in Figure 7.1 that is robust to missing data, corrupt data, wrong semantic correspondence and erroneous vertex correspondence. To achieve this robustness to erroneous training data, RMM jointly learns a multilinear model and corrects the data. First, we describe the groupwise multilinear objective function that minimizes multilinear compactness in Section 7.1.1. Second, we describe how to optimize the objective function to complete and clean up an incomplete database and correct for wrong semantic correspondence, making it possible to build a multilinear model as described in Section 3.4.

7.1.1 Multilinear objective function

Our objective function consists of three parts: a compactness energy E_C, a data energy E_D, and a regularization energy E_R^μ as

$$E(\mathcal{X}, w_D, w_R, \mu) = E_C + w_D E_D + w_R E_R^\mu, \tag{7.1}$$

where the weights w_D and w_R control the influence of the data and regularization terms, respectively. The parameter μ specifies the influence of the regularization target. We now describe all terms in more detail.

Compactness: The multilinear compactness term (Eq. 6.2) used for registration optimization in Chapter 6 aims to minimize the mode ranks of $\mathcal{X}$ by minimizing the ranks of $\mathbf{X}_{(2)}$ and $\mathbf{X}_{(3)}$. Minimizing E_C implicitly favors compact multilinear models as

$$E_C = \frac{1}{d_2} \ln(det(\mathbf{D}_2 + \delta_2 \mathbf{I}_{d_2})) + \frac{1}{d_3} \ln(det(\mathbf{D}_3 + \delta_3 \mathbf{I}_{d_3})), \tag{7.2}$$

where $\mathbf{D}_2 = \frac{1}{d_3}\mathbf{X}_{(2)}\mathbf{X}_{(2)}^T$ and $\mathbf{D}_3 = \frac{1}{d_2}\mathbf{X}_{(3)}\mathbf{X}_{(3)}^T$ are the mode-2 and mode-3 covariance matrices, and $\mathbf{I}_{d_i} \in \mathbb{R}^{d_i \times d_i}$ is the identity matrix. The small regularization constant δ_n avoids singularities of E_C for mode covariance matrices without full rank.

Note that while the notation differs, Equations 7.2 and 6.2 are equivalent. The eigenvalues in Equation 6.2 are used to show the relation to the compactness measure, and to derive the analytic gradient of the compactness term in Appendix A.1.2. In this chapter, however, we omit explicitly defining the eigenvalues for simplicity.

Data: The data term measures the distance of a corrupt shape $\mathbf{x}$ in $\mathcal{X}$ (aligned with the first mode of $\mathcal{X}$) to a corresponding unregistered face scan $\mathbf{s}$. The data energy is

$$E_D = \frac{1}{n}\sum_{k=1}^{n} \min(\|\mathbf{v}_k(\mathbf{x}) - \mathbf{nn}_k\|^2, \rho), \tag{7.3}$$

where $\mathbf{nn}_k$ denotes the nearest neighbor of $\mathbf{v}_k(\mathbf{x})$ in $\mathbf{s}$ computed by a point-to-plane distance measure, and ρ is a truncation threshold to be robust to outliers.

Regularization: The regularization term for each shape $\mathbf{x}$ in $\mathcal{X}$ is a bi-Laplacian of the form

$$E_R^{\mu} = \frac{1}{n}\sum_{k=1}^{n} \left\|U^2(\mathbf{v}_k(\mathbf{x})) - \mu U^2(\mathbf{v}_k(\widetilde{\mathbf{x}}))\right\|^2, \tag{7.4}$$

where $\mathbf{v}_k(\mathbf{x})$ and $\mathbf{v}_k(\widetilde{\mathbf{x}})$ denote the k-th vertex of shape $\mathbf{x}$ and the fixed reference shape $\widetilde{\mathbf{x}}$, respectively. The energy E_R^{μ} measures the deformation energy of $\mathbf{x}$ relative to $\widetilde{\mathbf{x}}$. The parameter $\mu \in [0, 1]$ controls the regularization influence of $\widetilde{\mathbf{x}}$. Minimizing E_R^{μ} forces $\mathbf{x}$ to be locally smooth, and the local geometry of $\mathbf{x}$ to be similar to $\widetilde{\mathbf{x}}$. Note that E_R^0 (i.e. $\mu = 0$) resembles the regularization from Section 6.1.1. The operator $U^2(\mathbf{p})$ approximates the discrete bi-Laplacian [79] as described in Section 5.2.1.

7.1.2 Optimization

RMM minimizes E (Eq. 7.1) to jointly learn a compact multilinear model, complete and clean up an incomplete database, and improve semantic correspondence, as outlined in Algorithm 1. The input of RMM is a set of $k \leq d_2 d_3$ shapes $\Omega_X = \{\mathbf{x}_{ie}\}$ with $i \in \{1, \cdots, d_2\}$ and $e \in \{1, \cdots, d_3\}$. All shapes in Ω_X are required to be in full per-vertex correspondence, which is possibly inaccurate due to drift. The remaining $d_2 d_3 - k$ shapes $\mathbf{x}_{ie} \notin \Omega_X$ are either corrupt or missing. In contrast to the registered shapes (in Ω_X), for corrupt shapes only partial, possibly noisy data are available that cannot be registered easily. For each corrupt $\mathbf{x}_{ie}$, we require as input an unregistered face scan $\mathbf{s}_{ie} \in \Omega_S$ that is rigidly aligned with the $\mathbf{x}_{ie} \in \Omega_X$. The indices (ie) of $\mathbf{x}_{ie} \in \Omega_X$ and $\mathbf{s}_{ie} \in \Omega_S$ define the initial semantic correspondence. For the remaining shapes (not given in $\Omega_X \cup \Omega_S$) no further information is provided. These shapes are called missing shapes.

After initialization, RMM first optimizes the semantic correspondence as described in Alg. 2. Then, RMM optimizes E for each shape in $\mathcal{X}$ individually as previously described in Section 6.1.3. That is, each iteration of the optimization processes all shapes of the database in random order to avoid bias towards specific shapes. This shape-wise optimization of E makes it possible to independently handle missing data, corrupt data, and inaccurate vertex correspondence as shown in Alg. 1. Finally, the multilinear model $\mathcal{M}$ is built from $\mathcal{X}$ after all shapes in $\mathcal{X}$ are fixed.

Algorithm 1: RMM

```
Data: $\Omega_X; \Omega_S$
Result: $\mathcal{M}$
Initialization;
for $M$ iterations do
    /* Opt. semantic corr. (Alg. 2)                    */
    $\min_{\pi} E(\mathcal{X}, 0, 0, 0)$
    /* Shape-wise optimization                          */
    for each shape do
        if x is missing then
            /* Estimate missing shape                   */
            $\min_{\mathbf{x}} E(\mathcal{X}, 0, w_R, 1)$
        else if x is corrupt then
            /* Reconstruct corrupt shape                */
            $\min_{\mathbf{x}} E(\mathcal{X}, w_D, w_R, 1)$
        else
            /* Vertex corr. opt. (Chapter 6)            */
            $\Phi(\min_{\boldsymbol{\alpha}} E(\mathcal{X}, 0, w_R, 0))$
        end
    end
end
Compute $\mathcal{M}$ (Eq. 3.15)
```

Initialization: All registered shapes are initially parametrized as described in Section 6.1.2. For each registered shape $\mathbf{x}_{ie} \in \Omega_X$ a thin-plate spline [45] defines a continuous mapping from $2D$ parameter space to the surface of $\mathbf{x}_{ie}$. The thin-plate spline is computed from a discrete mapping between parameters $\boldsymbol{\alpha}_k \in \mathbb{R}^2$ and vertices $\mathbf{v}_k(\mathbf{x}_{ie})$ of $\mathbf{x}_{ie}$. Let $\Phi_{ie}(\boldsymbol{\alpha}) = \mathbf{x}_{ie}$ denote the mapping of $\boldsymbol{\alpha} = (\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_n)^T$ to $\mathbf{x}_{ie}$.

Each missing and corrupt shape $\mathbf{x}_{ie} \notin \Omega_X$ is initialized by the mean over the registered shapes of the same identity i and expression e. Let $\Omega_i := \{\mathbf{x}_{ie} | \forall e \in \{1, \ldots, d_3\} : \mathbf{x}_{ie} \in \Omega_X\}$ and $\Omega_e := \{\mathbf{x}_{ie} | \forall i \in \{1, \ldots, d_2\} : \mathbf{x}_{ie} \in \Omega_X\}$ denote the set of registered shapes of identity i,

Algorithm 2: Semantic correspondence opt.

```
Data: $\mathcal{X}$; threshold $\tau$
Result: $\mathcal{X}$ relabeled
for each identity $i$ do
    $\tau_i = \tau$
    $\pi_i := \{\pi_i(1), \dots, \pi_i(d_3)\} = \{1, \dots, d_3\}$
    $\pi_{best} = \pi_i$; $E_{best} = E_i = E(\mathcal{X}, 0, 0, 0)$
    for $N_t$ iterations do
        for $N_s$ iterations do
            Locally change $\pi_i$ randomly to $\pi_*$
            $\mathcal{X}^* = \mathcal{X}$
            $\mathbf{x}^*_{ie} = \mathbf{x}_{i\pi_*(e)} \;\; \forall e \in \{1, \dots, d_3\}$
            $E^* = E(\mathcal{X}^*, 0, 0, 0)$
            if $E^* < E_i + \tau_i$ then
                $\pi_i = \pi^*$; $E_i = E^*$
            end
            if $E^* < E_{best}$ then
                $\pi_{best} = \pi^*$; $E_{best} = E^*$
            end
        end
        $\tau_i = 0.5 \cdot \tau_i$
    end
    $\mathbf{x}_{ie} = \mathbf{x}_{i\pi_{best}(e)} \;\; \forall e \in \{1, \dots, d_3\}$
end
```

and expression e, respectively. The shape $\mathbf{x}_{ie}$ is initialized as

$$\mathbf{x}_{ie} = 0.5 \left(\frac{1}{|\Omega_i|} \sum_{\mathbf{x}\in\Omega_i} \mathbf{x} + \frac{1}{|\Omega_e|} \sum_{\mathbf{x}\in\Omega_e} \mathbf{x} \right), \tag{7.5}$$

where $|\Omega_i|$ and $|\Omega_e|$ denote the cardinality of Ω_i and Ω_e, respectively. We call this initialization technique the averaging scheme (AVS) in the following. We use the result of AVS as reference shape $\widetilde{\mathbf{x}}$ in E_R.

Semantic correspondence optimization: To optimize semantic correspondence, RMM minimizes $E(\mathcal{X}, 0, 0, 0) = E_C$. Jointly optimizing the semantic correspondence over all data is infeasible due to the large number of parameters $(m_2 m_3)$. Further, as multiple expressions should be permuted, shape-wise optimizing E is impossible. Instead, we use a strategy inspired by the shape-wise optimization that optimizes E for each identity individually. Note that as for the shape-wise optimization, E still depends on all shapes, and hence the method remains a groupwise optimization. To avoid any bias towards specific identities, the order of the processed identities in each iteration is chosen randomly.

For each identity i we search for the permutation $\pi_i = \{\pi_i(1), \ldots, \pi_i(d_3)\}$ with $\pi_i(e) \in \{1, \ldots, d_3\}$ of the expressions of i that minimizes E. Note that π_i only changes the labeling of the expressions for each identity; the geometry of the shapes remains unchanged. Due to the domain of π_i, this is an integer problem.

Integer problems are often solved by discretization, i.e. instead of the integer problem $\pi \subseteq \mathbb{Z}$ a discretized problem $\pi \subseteq \mathbb{R}$ is optimized. The optimization of the discretization of E with a local method such as L-BFGS like in the other RMM optimization steps fails due to many local minima.

Instead, we directly solve the integer problem approximatively. We optimize E with a threshold accepting (TA) method [106] as outlined in Algorithm 2. Given an initial threshold τ, the iteratively decreasing τ equates to the cooling schedule of simulated annealing. TA uses two iterations, one to lower the threshold, and one for optimization for a certain threshold. TA stores the minimum E_{best} of E together with the corresponding best permutation π_{best}. In one optimization iteration, π_i is randomly altered to π_* by permuting 10% of the elements of π_i, the expressions of i in $\mathcal{X}$ are permuted accordingly to $\mathcal{X}^*$, and E is evaluated for $\mathcal{X}^*$. Depending on τ_i, π^* is used as the starting point for the next iteration. If a new minimum is found, E_{best} and π_{best} are updated. Finally, the expressions of i in $\mathcal{X}$ are permuted by π_{best}. The threshold τ can be chosen automatically.

Vertex correspondence optimization: The vertex correspondence is optimized as described in Section 6.1.3. To optimize the vertex correspondence of $\mathbf{x}_{ie} \in \Omega_X$, RMM minimizes $E(\mathcal{X}, 0, w_R, 0) = E_C + w_R E_R^0$ by reparametrizing $\mathbf{x}_{ie}$. As the energy E is analytically differentiable with respect to the parameters $\boldsymbol{\alpha}$ of $\mathbf{x}_{ie}$ (see Appendix A.1 for the derivatives), E is minimized as in Chapter 6. The optimized shape $\mathbf{x}_{ie}$ is updated as $\mathbf{x}_{ie} = \Phi_{ie}(\boldsymbol{\alpha})$.

Missing data estimation: To estimate a missing shape $\mathbf{x}_{ie} \notin \Omega_X$, $\mathbf{s}_{ie} \notin \Omega_S$, RMM minimizes $E(\mathcal{X}, 0, w_R, 1) = E_C + w_R E_R^1$. In contrast to the vertex correspondence optimization, E is minimized in Euclidean vertex space using L-BFGS [92] rather than in parameter space. That

is, during optimization each vertex of the missing shape moves in $\mathbb{R}^3$ to minimize E. This is required as the geometry of the missing shape is unknown.
Corrupt data estimation: To estimate the shape from a corrupt face scan $\mathbf{s} \in \Omega_S$, RMM minimizes $E(\mathcal{X}, w_D, w_R, 1) = E_C + w_D E_D + w_R E_R^1$. To be robust to erroneous initial alignments, the alignment of **s** is refined using an iterative closest point algorithm. As for the missing data estimation, E is minimized in Euclidean vertex space using L-BFGS [92].

7.2 Evaluation

This section evaluates the robustness of RMM to missing data, to corrupt data, and to wrong semantic correspondence.
Data: We evaluate RMM on the BU-3DFE database [139] and the Bosphorus database [116]. Both databases are initially registered with an automatic template fitting method [114] that uses the landmarks provided with the databases. For BU-3DFE we use the same subset, called the *BU-3DFE set* as in Chapter 6. For Bosphorus we randomly choose 30 identities and use 17 action units and call this subset the *Bosphorus set*. Note that the Bosphorus set contains more expressions for fewer subjects than the Bosphorus set used in Chapter 6.

The robustness of RMM to missing data is evaluated on the BU-3DFE set and the Bosphorus set, each with randomly removed shapes. For evaluation, we use, for both datasets, configurations with 1%, 5%, 10%, 25%, and 50% of the shapes missing.

The robustness of RMM to corrupt data is evaluated on the BU-3DFE set and the Bosphorus set, each with subsets of corrupt data due to simulated and real partial occlusions. While the BU-3DFE set is only corrupted by simulated occlusions, the Bosphorus set contains noisy and partially occluded face scans, which we use to substitute the complete scans in our experiments. The occlusions are selected to affect the facial regions shown in the top row of Figure 7.4. We use, for both datasets, configurations with 1%, 5%, 10%, 25%, and 50% corrupt shapes during evaluation.

The robustness of RMM to wrong semantic correspondence is evaluated on the BU-3DFE set and the Bosphorus set, each with a subset of randomly generated erroneously labeled expressions. To simulate erroneously labeled expressions, the wrong semantic correspondence subsets consist of randomly chosen identities, where the expressions are randomly permuted. We use for both datasets configurations with randomly permuted expression labelings of 5%, 10%, 25%, 50%, and 100% of the identities.
Parameter settings: For our experiments, all parameters are fixed for all experiments on two different databases and with varying degrees of missing data, corrupt data, and wrong semantic correspondence. The parameters w_D and w_R (Eq. 7.1) control the influence of the data and regularization terms, respectively. We choose $w_D = 1e-3$ and $w_R = 20$ to reconstruct missing and corrupt data, and $w_R = 0.5$ to optimize vertex correspondence. For databases that contain less corrupt data than in our experiments, w_D could be set higher and w_R could be set lower to allow the recovery of more facial detail. The parameters δ_2 and δ_3 are used to avoid singularities of E_C (Eq. 7.2), and we choose them as $\delta_2 = \delta_3 = 0.01$ as in Chapter 6.

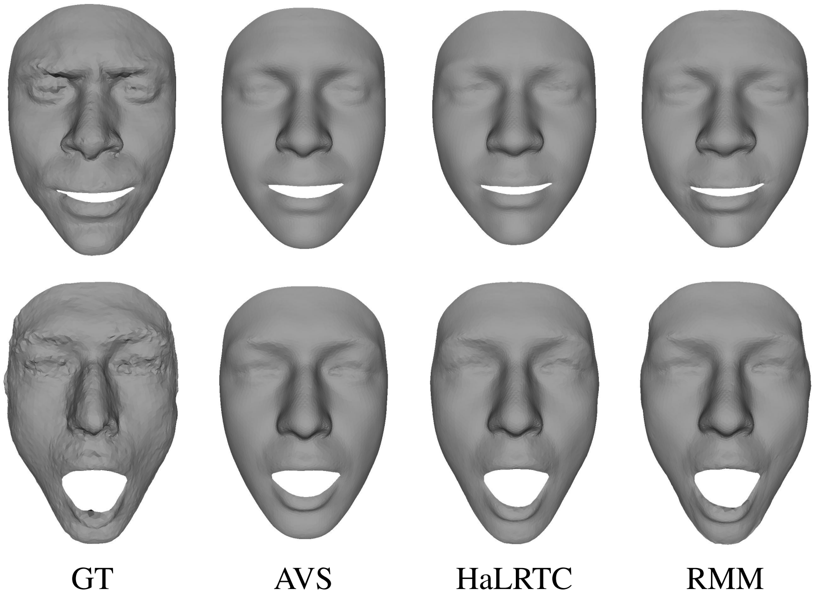


Figure 7.2: Comparison of robustness to missing data. From left to right: Ground truth (GT). Averaging scheme (AVS). HaLRTC [93].

The parameter ρ (Eq. 7.3) relates directly to the size of the face, and can be fixed at 5 mm. The parameters M (Alg. 1), N_t (Alg. 2), and N_s (Alg. 2) control the number of iterations performed, and allow a trade-off between running time and accuracy. We choose them as $M = 15$, $N_t = 10$, and $N_s = 200$.

7.2.1 Robustness to missing data

Objective function: To study the influence of E_R on E for missing data completion, we optimize E with ($w_D = 1e - 3$) and without ($w_D = 0$) regularization. During optimization, each shape has only limited influence on E. We observed that the shape-wise optimization of E_C overcompensates for the limited influence of few shapes and may produce unlikely shapes. The regularization successfully prevents this overcompensation, as it penalizes strong local distortions.

Comparison: We compare our RMM to the ground-truth shape, to AVS, and to the result of the state-of-the-art tensor completion method HaLRTC [93]. Figure 7.2 visually compares the completed shapes. While HaLRTC and RMM result in a better estimation of the missing shape than AVS, they perform rather similarly. Figure 7.3 shows the median error, measured as the distance of all completed shapes to the ground truth for all configurations. HaLRTC and RMM perform better than AVS if up to 10% of the data are missing. While for the Bosphorus set RMM performs slightly better than HaLRTC, the overall performance of the two methods is similar.

Summing up, given a dataset with missing data, RMM reconstructs the missing data well.

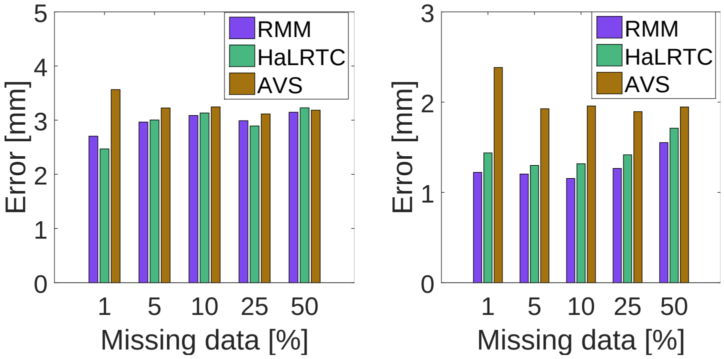


Figure 7.3: Median error of HaLRTC [93] and AVS for different missing data configurations compared to RMM. Left: BU-3DFE set. Right: Bosphorus set.

7.2.2 Robustness to corrupt data

Objective function: To show the individual influence of each term of E to reconstruct corrupt data, we optimize E with different combinations of energy terms. Figure 7.5 visually compares the results for the different combinations. The optimization of E_D closely reconstructs **s** in non-corrupt regions, but corrupt regions produce strong artifacts, and the expressions are not always well reconstructed. The optimization of $E_C + w_D E_D$ reconstructs the shape and the expression of **s** well in non-corrupt regions, and gives a reasonable prediction of the shape for corrupt regions, but corrupt regions contain artifacts. Note that E_C is unable to regularize E_D sufficiently as (even strong) local distortions in the reconstruction only have a negligible influence on E_C. The optimization of $w_D E_D + w_R E_R$ avoids the artifacts in corrupt regions, but the facial expression is not reconstructed well. The full optimization of E (RMM) reconstructs the facial expression well and is robust to corrupt data.

Comparison: As statistical face models are known to be robust to partial occlusions and noise (see e.g. Chapter 5), we compare RMM to a multilinear model reconstruction of the corrupt data. Since the multilinear face model requires a complete data tensor for training, the data tensor is completed using HaLRTC [93]. A multilinear face model is trained that keeps 95% of the identity and expression variations on the completed data, and all corrupt shapes of the dataset are reconstructed. We call this combination of existing methods HaLRTC+MM in the following. In contrast to RMM, HaLRTC+MM gets facial landmarks for fitting to initialize the expression.

Figure 7.6 visually compares HaLRTC+MM and RMM for 10% corrupt data. While both methods are robust to corrupt data, RMM better reconstructs the facial expression. Further, RMM is better at reconstructing the facial shape, e.g. at the nose. Since the distance-to-data measure is only a valid error measure in non-occluded regions, we define for each type of occlusion a valid region as visualized in the bottom of Figure 7.4. The error measure then only uses vertices within the valid regions. Figure 7.7 shows the cumulative error plots for both datasets with 10% corrupt data. For both datasets RMM performs better than HaLRTC+MM.

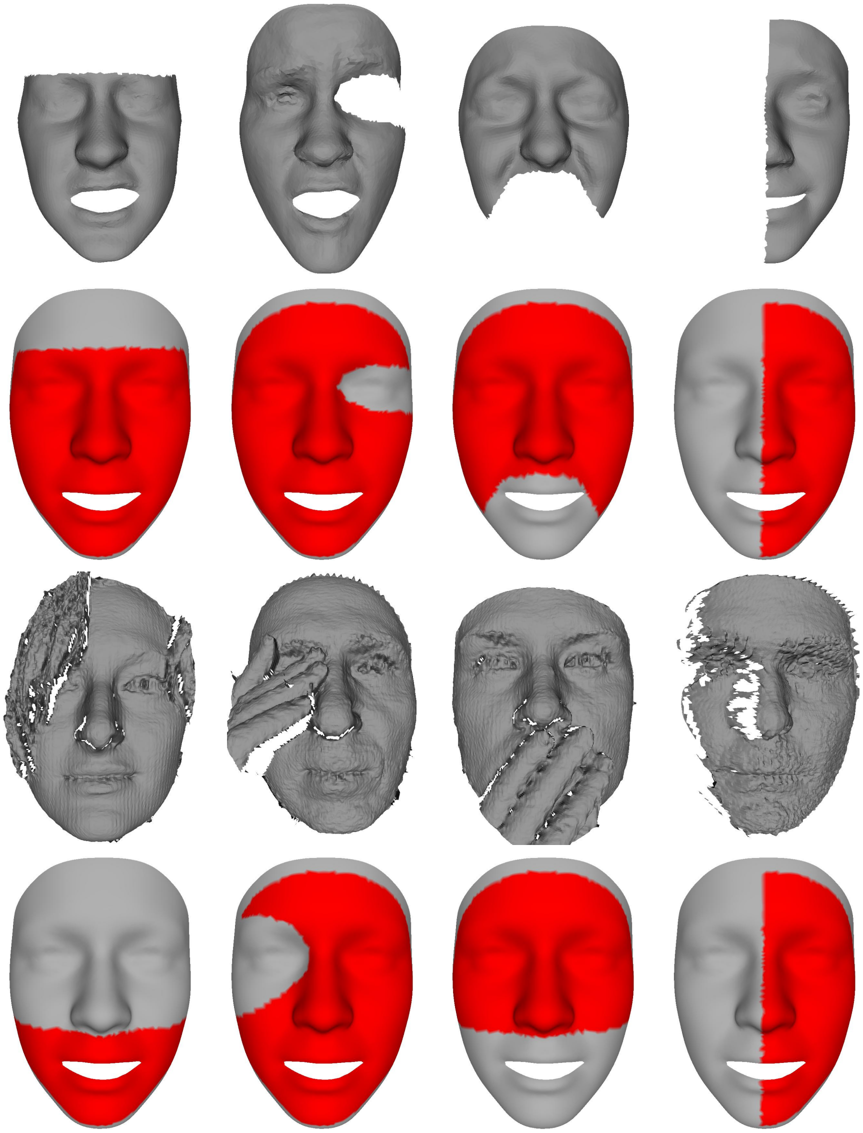


Figure 7.4: Samples of corrupt data and corresponding valid regions (red) for each type of occlusion used for error measure. Top: Simulated occlusions. Bottom: Real occlusions in the Bosphorus database.

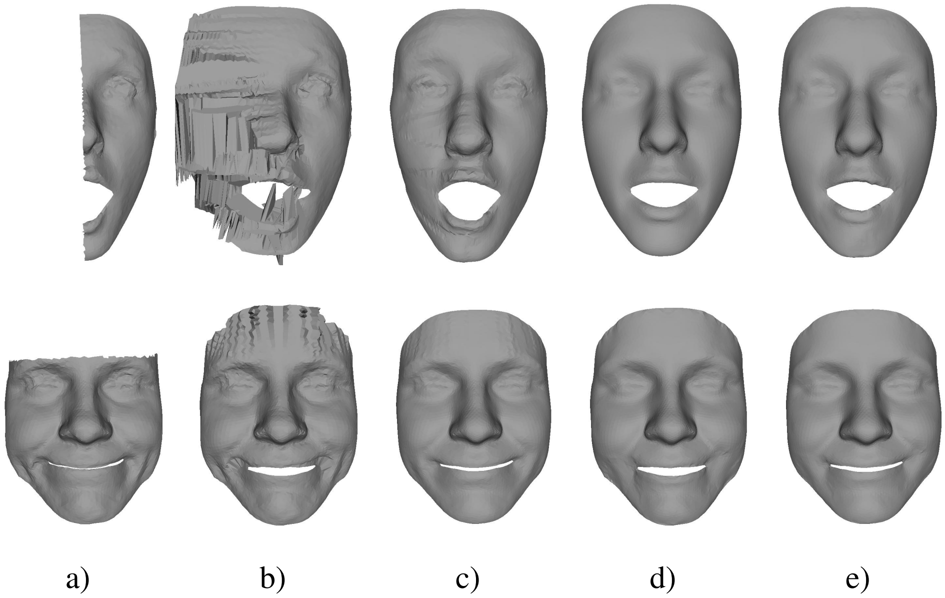


Figure 7.5: Influence of each term in E (Eq. 7.1) to reconstruct corrupt data (10% corrupt). From left to right: a) Corrupt scan $\mathbf{s}$. Optimization of: b) E_D. c) $E_C + w_D E_D$. d) $w_D E_D + w_R E_R$. e) RMM. Top: BU-3DFE set. Bottom: Bosphorus set.

For most other configurations RMM performs better than HaLRTC+MM as shown in Figure 7.8. For the BU-3DFE set with 50% corrupt data, RMM reconstructs a few expressions incorrectly due to the sparse sampling of the data, while HaLRTC+MM more successfully reconstructs the expressions thanks to the additionally provided landmarks. To reconstruct corrupt data, RMM assumes AVS to give a reasonable initialization of the expression of $\mathbf{s}$ as the iterative nearest neighbor terms E_D is known to only converge locally. This requires the expression of $\mathbf{s}$ to be similar to the expressions in Ω_X. Using landmarks for initialization could help RMM to reconstruct extreme expressions more reliably.

Summing up, given a dataset with corrupt data, RMM provides a reconstruction that preserves facial details while being robust to partial occlusions and noise.

7.2.3 Robustness to wrong semantic correspondence

We quantitatively evaluate the optimized semantic correspondence using compactness, generalization, and specificity extended to the multilinear case as described in Section 4.2.2. Figure 7.9 shows the influence of wrong semantic correspondence on compactness, generalization, and specificity (identity mode) for the BU-3DFE set (top) and the Bosphorus set (bottom) for randomly distorted expression labelings of 50% of the identities. Compared to the ground truth (GT), the model with wrong semantic correspondence (Init) is less compact, less general, and more specific. After optimization (RMM) the model becomes significantly more compact, more general, and less specific, comparable to the ground truth. Hence, after

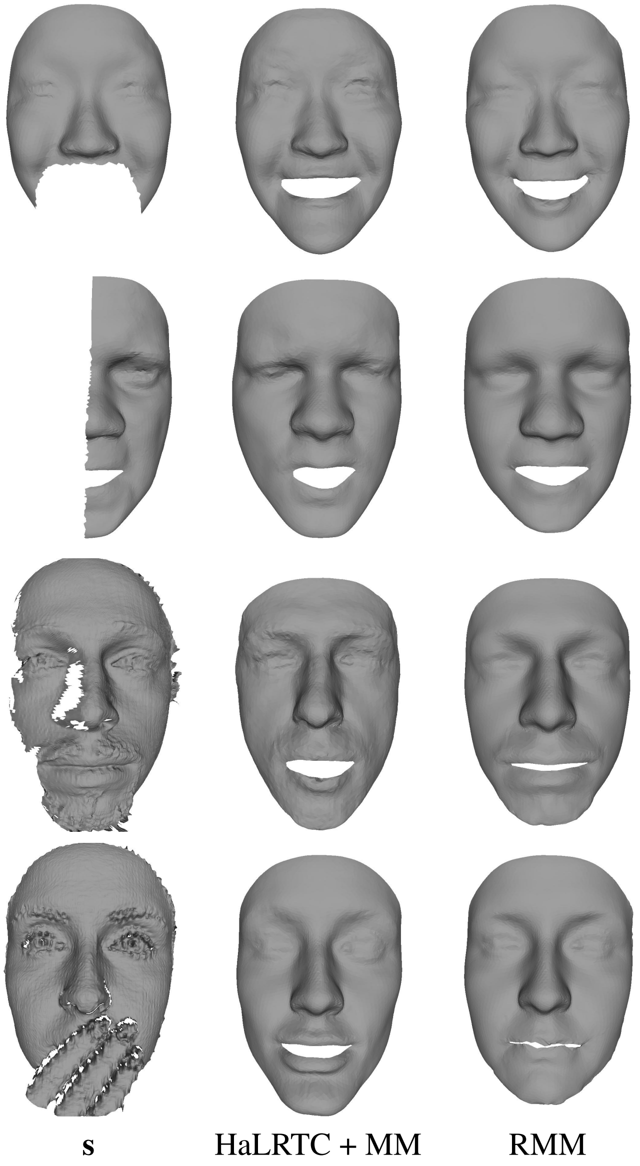


Figure 7.6: Comparison with combination of HaLRTC [93] and multilinear model (MM) to reconstruct corrupt data (10% corrupt). Top: BU-3DFE set. Bottom: Bosphorus set.

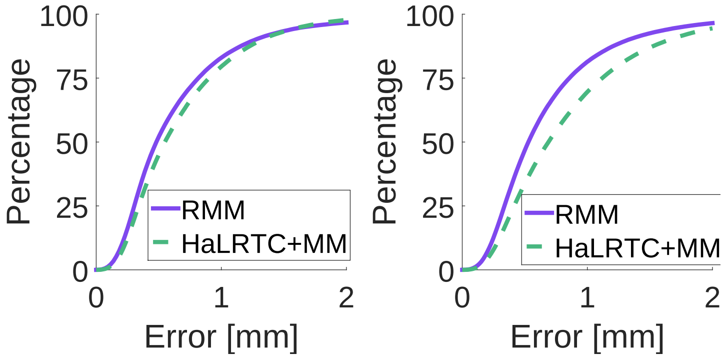


Figure 7.7: Cumulative error of combination of HaLRTC [93] and multilinear model for 10% corrupt data compared to RMM. Left: BU-3DFE set. Right: Bosphorus set.

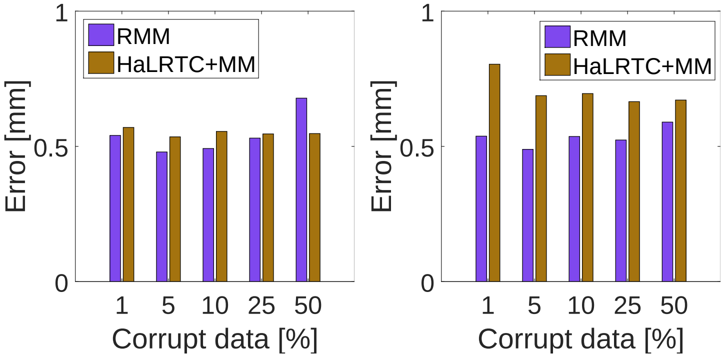


Figure 7.8: Median error of combination of HaLRTC [93] and multilinear model for different corrupt data configurations compared to RMM. Left: BU-3DFE set. Right: Bosphorus set.

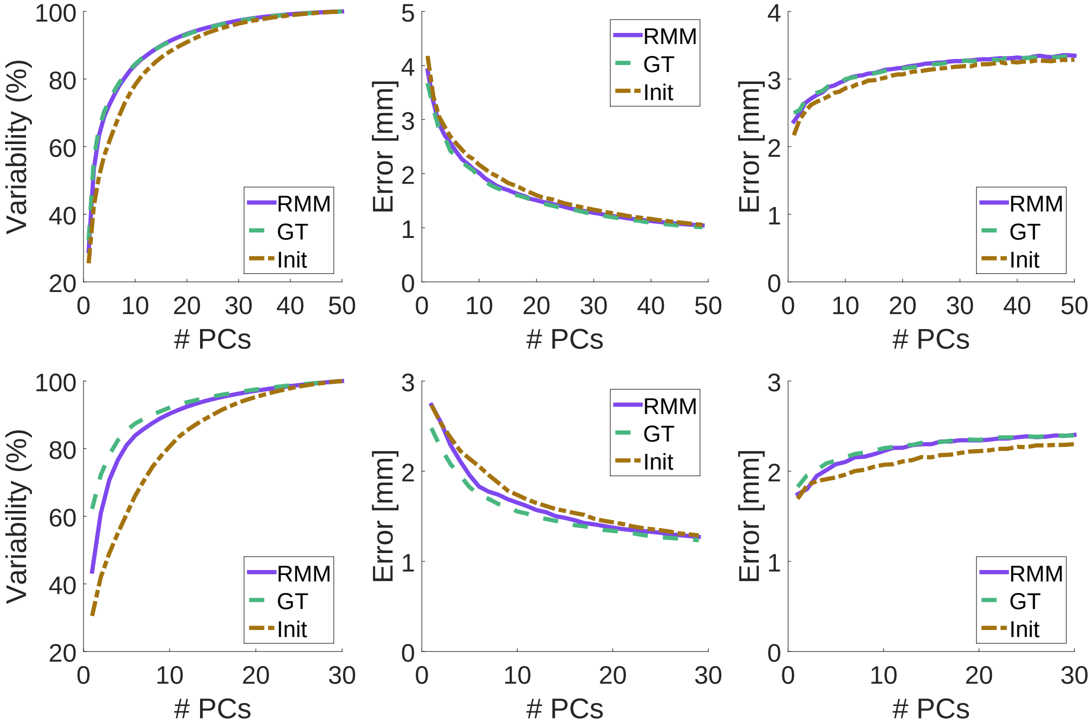


Figure 7.9: Comparison to ground truth (GT) for randomly permuted labeling of 50% of the identities before (Init) and after optimization (RMM). Left: Compactness. Middle: Generalization: Right: Specificity. Top: BU-3DFE set. Bottom: Bosphorus set.

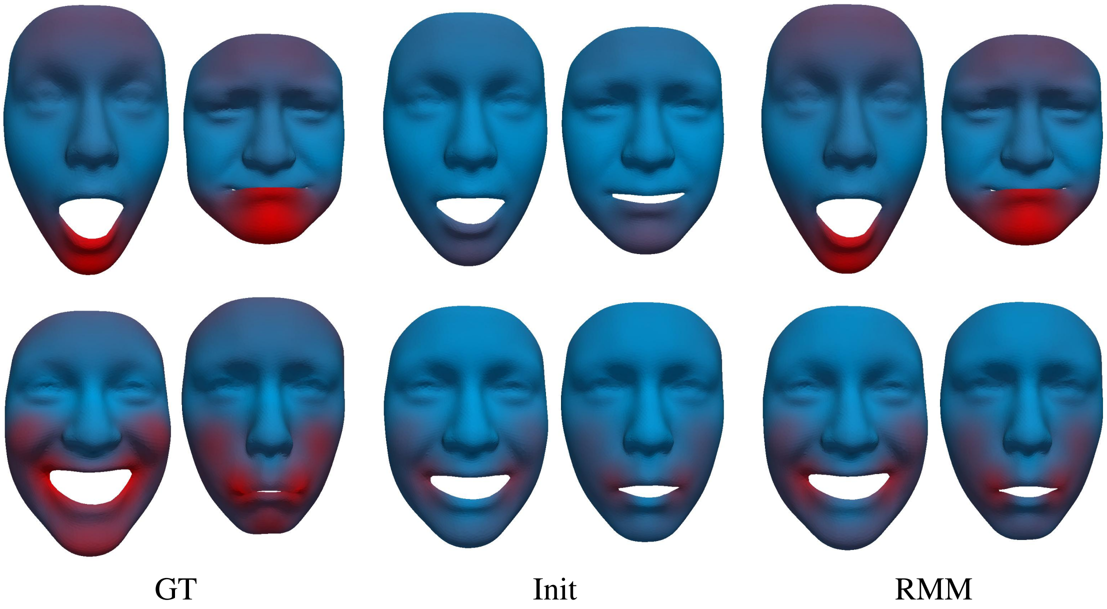


Figure 7.10: Expression variations of two expression components (rows) for randomly permuted labeling of 50% of the identities for the BU-3DFE set. The magnitude of the vertex displacement is color coded from blue (zero) to red (maximum). Left: GT. Middle: Init. Right: RMM.

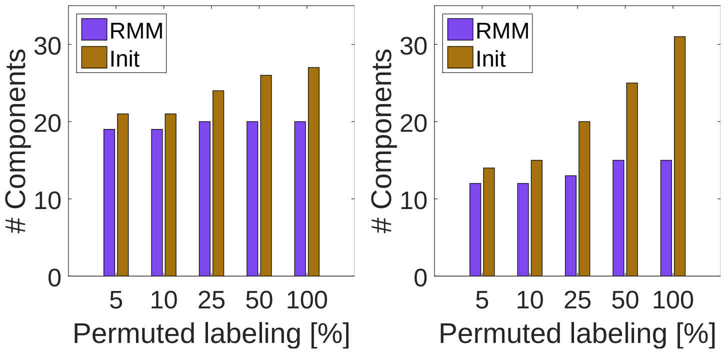


Figure 7.11: Number of components needed to keep 90% of the data variability before (Init) and after optimization (RMM). Left: BU-3DFE set. Right: Bosphorus set.

optimizing the semantic correspondence, the model requires fewer components to capture the same variability of the data.

When 50% of the data are permuted, to keep 90% of the data variability before optimization, a total of 26 and 25 components are necessary for the BU-3DFE and Bosphorus sets, respectively, while after optimization 20 and 15 components suffice for the BU-3DFE and Bosphorus sets, respectively. Figure 7.10 shows the variations of two expression components. The variations of the model increase significantly after optimization. For the other configurations RMM also gives significant improvements as shown in Figure 7.11.

Summing up, given a dataset with wrong semantic correspondence, RMM improves the semantic correspondence, and results in a more compact model.

7.3 Summary

This chapter presented a groupwise multilinear model learning framework that is robust to missing data, corrupt data, wrong semantic correspondence, and inaccurate vertex correspondence. This allows highly accurate multilinear face models to be built from existing 3D face databases. We have evaluated our framework on two databases with multiple levels of missing data, corrupt data caused by noise and partial occlusions, and erroneously labeled expressions. We have shown that our framework completes data comparably to state-of-the-art tensor completion methods, that it reconstructs corrupt data better than state-of-the-art methods, and that the quality of the learned model increases significantly for erroneously labeled expressions.

Chapter 8

Motion sizing system

> "*It is the common wonder of all men, how among so many millions of faces, there should be none alike.*"
>
> – Sir Thomas Browne

This chapter presents one possible application of the shape space-based registration methods developed in this thesis. More specifically, Chapters 4 and 5 described methods to fully automatically register entire 3D facial motion sequences. This chapter introduces a general framework to compute a sizing system by leveraging such registered motion data.

For the design of mass-produced wearable objects for a population it is important to find a small number of sizes, called a sizing system, that will fit well on a wide range of individuals in the population. To obtain a sizing system that incorporates the shape of an identity along with its motion, we introduce a general framework to generate a sizing system for dynamic 3D motion data. Based on a registered 3D motion database a sizing system is computed for task-specific anthropometric measurements and tolerances, specified by designers.

8.1 Motivation

Face masks and respirators exist in many different types and sizes and are widely used by the military (e.g. for pilots' oxygen masks [86]), by public safety departments (e.g. respirators for firefighters [9]), and for medical (e.g. aerosol face masks [7]) and automotive applications (e.g. paint respirators). Depending on the type of face mask, it is designed to supply oxygen or filter air. For most kinds of face masks it is important to fit many different kinds of face shapes. Leakage could cause, for aerosol face masks, a contamination of the caregiver's area, and for respirators, an inhalation of harmful gases and particles, which could cause lung diseases or other health problems. Furthermore, loosely fitting oxygen masks with leakage towards the eyes are uncomfortable to wear. A tight fit without leakage is therefore crucial for the design of an effective face mask.

In ergonomics, many works exist that aim at creating sizing systems based on anthropometric measurements for the design of face masks [7, 86, 9, 60], helmets [134], gloves [82], or more generally, for apparel [99]. The aim of generating a sizing system with a low number of different sizes is that a designed product should fit a wide range of individuals in the

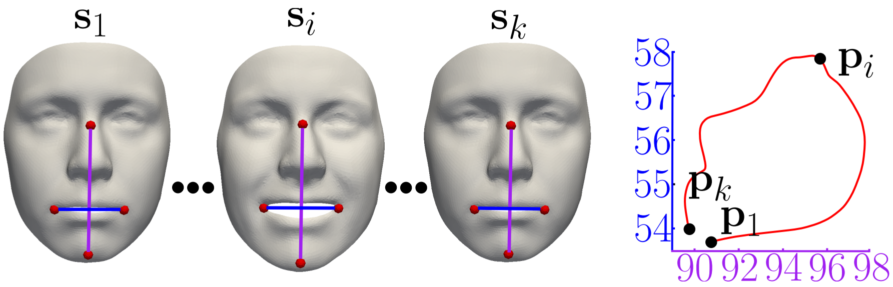


Figure 8.1: Representation of the anthropometric measurements face length (purple) and lip width (blue) for motion sequence. Left: 3D motion sequence. Right: Resulting curve in parameter space.

population it is designed for. To generate a sizing system, design-specific anthropometric measurements are gathered for a population and groups are formed, where identities with similar measurements are within the same group. Each group is then represented by a size within the sizing system.

Currently, the design of face masks only considers the shape of neutral faces. Since face masks are worn for long periods, it is likely that a wearer will move his or her face while wearing the mask, e.g. by talking or changing facial expressions. Therefore, a tight fit of the face mask is also necessary in the presence of facial motion, to avoid leakage caused by motion.

Given a registered motion database, the input for our framework is the specification of the anthropometric measurements used. Furthermore, an ordered set of tolerances must be specified for each dimension, and the number of sizes that should be computed must be given (otherwise a sizing system is found that fits for all input data). These input parameters are specific to the designed product and must be specified by designers. Given these parameters, our framework outputs a sizing system with the specified number of sizes, together with representative 3D shape models for each size.

Given a set of problem-specific anthropometric measurements, each shape in the database of 3D motion data is represented by a point in high-dimensional parameter space. A sizing system is then computed by solving a stabbing problem in parameter space.

8.2 Parameter space for dynamic motion data

In this section, we introduce a parameter space of anthropometric measurements for dynamic data, and describe a method to fully automatically compute a sizing system for this parameter space. Given a database of 3D faces in motion in full correspondence, we extract an ordered set of d anthropometric measurements from each scan. For each scan $\mathbf{s}_i$ of a motion sequence, the set of measurements is denoted by $\mathbf{p}_i \in \mathbb{R}^d$. The set of all measurements of all scans defines the high-dimensional parameter space $\mathbb{P} \subseteq \mathbb{R}^d$. Since each frame of a motion sequence

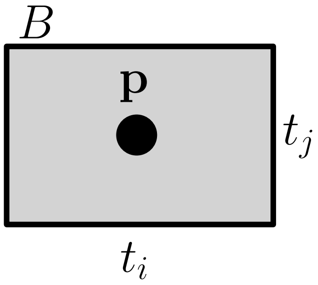


Figure 8.2: Given some tolerances t_i and t_j and some gear designed for measurements represented by a point **p** in parameter space, all points within a parameter box B centered at **p** are fit by the gear.

gives a point in $\mathbb{P}$, an entire sequence is represented by a curve in $\mathbb{P}$. Figure 8.1 shows two measurements extracted from a motion sequence, resulting in a curve in $\mathbb{P}$. Since for each identity, multiple motion sequences may exist, one identity is represented by a set of curves, one for each motion sequence.

The designer can specify a tolerance t_i along each dimension i that specifies the amount of stretch supported by the specific gear. For the specified tolerances, a d-dimensional axis-aligned parameter box B is defined, where the length of the side in dimension i is t_i. Some gear designed to fit for some measurements $\mathbf{p} \in \mathbb{P}$ therefore also fits to all points in $\mathbb{P}$ within a translated copy of B centered at **p** (see Figure 8.2). A sizing system can then be computed by covering the parameter space using translated copies B_i of B. Since our goal is to design a sizing system for motion data, where the gear fits for an identity through various motions, all curves of one identity must be contained within the same box B_i.

8.3 Covering of parameter space using box stabbing

All curves of one identity need to be covered by the same box. The greedy box covering method by Wuhrer et al. [134] repeatedly selects the box centered at a point in parameter space that covers the most uncovered points. This greedy covering method cannot be applied to dynamic data, since a box centered at one point does not necessarily cover all curves of the identity.

Instead, we transform the problem into a d-dimensional stabbing problem as shown in Figure 8.3. First, we compute, for each identity, the area I_{id}, where a box B_i can be centered to cover all curves of that identity. Figure 8.4 shows the construction of I_{id} for three selected points of one identity. For each point $\mathbf{p}_i$ from one identity (for one identity, each frame of each motion sequence is represented by $\mathbf{p}_i \in \mathbb{P}$) we define I_i to be the area within a copy of B, centered at $\mathbf{p}_i$. By construction, any B_i with center within I_i contains $\mathbf{p}_i$. We obtain I_{id} by intersecting all I_i of one identity. For each identity the area I_{id} defines a region where each point chosen as the center of B_i covers all points $\mathbf{p}_i$ belonging to one identity. If a point within

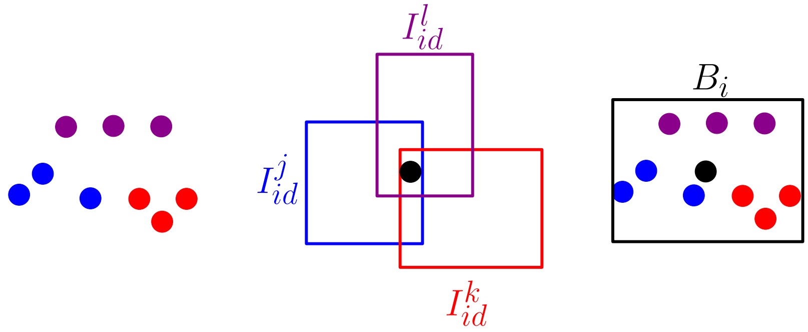


Figure 8.3: Computation of box covering using box stabbing. Left: Multiple points in parameter space from different identities (one color per identity) that should be covered. Center: Identity boxes together with a stabbing point (black). Right: Parameter box centered at the computed stabbing point that covers all points of different identities.

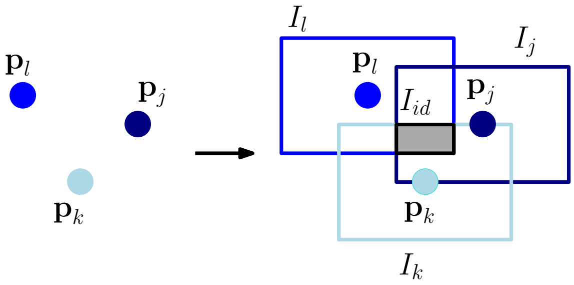


Figure 8.4: Computation of the identity box I_{id} for points of one identity. The box I_{id} bounds the area, where each point chosen as center of B_i covers all points of the identity in parameter space.

the intersection of multiple I_{id} is chosen as the center of B_i, B_i contains multiple identities.

To get a covering of the parameter space we now search for the minimum set of points such that each I_{id} is stabbed by at least one point. Each stabbing point represents the center of a cover box in parameter space. We use the method by Nielson [104] to compute this stabbing.

8.3.1 Full stabbing of dynamic identity boxes

To compute the optimal stabbing of 1-dimensional intervals and axis-parallel d-dimensional boxes, Nielson [104] proposes two divide-and-conquer algorithms. While the 1-dimensional stabbing can be solved optimally, computing a d-dimensional stabbing for $d \geq 2$ is NP-complete [53]. The proposed algorithm to compute the d-dimensional stabbing gives a bounded approximation of the optimal solution.

To get an optimal 1-dimensional stabbing, the rightmost lower interval point is selected and all intervals that are stabbed by this point are removed. This is repeated until all intervals are stabbed. This stabbing is computed using the following output-sensitive algorithm. The input set of n intervals I is recursively split into right and left subsets of intervals, with respect to the median of all lower interval endpoints. If a subset contains only one interval, the lower endpoint of the interval is chosen as a stabbing point. All intervals stabbed by the chosen stabbing point are removed from further processing. The algorithm stops once all intervals are stabbed. The time complexity of this stabbing is $\Theta(n \log c^*(I))$, where $c^*(I)$ denotes the minimum number of stabbing points necessary to stab all intervals.

To compute a stabbing of a set I of n d-dimensional axis-parallel boxes, the input set of boxes is separated into three subsets. For dimension d of the boxes, a stabbing is computed for the 1-dimensional intervals and the median stabbing point is used to separate the input set of boxes into three subsets: all boxes that intersect the median stabbing point, the subsets to the left, and the subset to the right of the median stabbing point. The right and left subsets are then recursively separated into three subsets. For the intersecting subset, the stabbing median value is fixed for dimension d and the stabbing of the $(d-1)$-dimensional boxes is computed recursively. The method outputs $c(I)$ points in time $O(dn \log c(I))$, where $c(I) \leq b^*(I)(1 + \log_2 b^*(I))^{d-1}$ with $b^*(I)$ is the maximum number of pairwise disjoint boxes.

8.3.2 Stabbing with a fixed number of points

For the design of wearables for large populations, it is not desirable to create a sizing system with a large number of different sizes that fits the entire population. Instead, a sizing system with a fixed number of sizes that fit the maximum number of individuals is sought. We therefore search for a fixed number of stabbing points that stab the maximum number of identity boxes. We use a greedy approach to solve this. We first compute the full stabbing of the parameter space using the method described in Section 8.3.1. We then iteratively select the stabbing points that stab the most unstabbed identity boxes.

8.4 Representation of covering

After computing a sizing system for the parameter space, we aim at computing a representative 3D face model for each of the sizes. This representative face model can be used for fabrication. One possibility is to compute the full Procrustes mean [45] of all identities covered by the box. To compute the full Procrustes mean of a set of shapes in correspondence, we iteratively compute the mean over all shapes, and each of the shapes is rigidly aligned to the mean shape. This is also used by Wuhrer et al. [134] to compute a representative model.

Another possibility is to select the model that is closest in parameter space to the cover box center as used by Han et al. [60] and Lee [86]. For data that are dense in parameter space, the model closest to the cover box center is expected to give a good representation of the box.

A further method to compute a representative model for the cover box is feature analysis by Allen et al. [3] as used by Wuhrer et al. [134]. Wuhrer et al. compute a linear mapping between the parameter space and a linear PCA space of 3D faces to reconstruct 3D faces for given sets of measurements in parameter space. In contrast to our approach, their method only uses faces

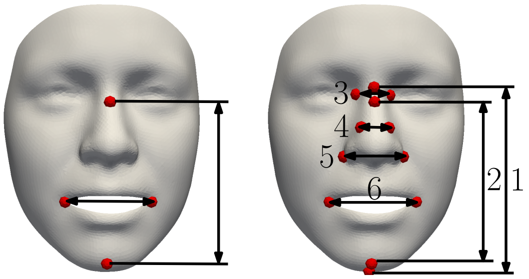


Figure 8.5: Important measurements for face mask design. Left: Two important measurements for the design of aerosol face masks [7]. Right: Six measurements classified as being of high importance for the design of oxygen masks [86].

in one neutral expression, and the variations of the data can therefore be modeled using a linear PCA model. Since our data contain variations due to motion and shape differences of different identities, the variations cannot be modeled using a linear space. Therefore, a linear mapping between the parameter space and the non-linear model space does not lead to representative 3D face models.

8.5 Evaluation

This section evaluates the proposed space covering using measurements associated with the design of face masks. The motion data are from the BU-4DFE database [138], registered using the method described in Chapter 4. Based on the temporal registration of the motion sequences, we automatically select five representative frames of each sequence that cover the full range of motion. In the following, each sequence is therefore represented by five points in parameter space.

In our experiments we show how well the computed sizing system fits for a given dataset, and its generalization to unseen data. To this end, we randomly divide the motion sequences into a training and a test set, each containing about 50% of the data, with the same ratio of male and female subjects. For our experiments we do not consider the surprised facial expression, since many of the surprise motion sequences are performed in an artificial fashion by fully opening the mouth, which we think would be an unnatural behavior for a person wearing a face mask. Hence, for each identity up to five motion sequences are used, which gives us up to 25 points in parameter space for each identity. Overall we use 390 dynamic motion sequences from 98 identities.

8.5.1 Anthropometric measurements for face mask design

For the design of face masks, different measurements are important, depending on the type of mask and its application area. Amirav et al. [7] use two measurements (shown in the left

Measurement	Face length	Lip width
Mean	10.87	8.05
Standard deviation	3.35	3.05
Median	10.98	8.08
Maximum	19.43	14.40

Table 8.1: Statistics in mm computed over the maximum measurement range over all identities for the 2D parameter space (for measurements see Figure 8.5, left).

Measurement	1	2	3	4	5	6
Mean	10.47	10.87	1.47	1.07	4.68	8.05
Standard deviation	3.32	3.35	0.56	0.50	1.80	3.05
Median	10.2	10.98	1.52	1.02	4.66	8.08
Maximum	18.59	19.43	2.91	2.34	8.28	14.40

Table 8.2: Statistics in mm computed over the maximum measurement range over all identities for the 6D parameter space (for measurements see Figure 8.5, right).

of Figure 8.5) for the design of aerosol face masks. Lee [86] classifies 22 facial measurements according to their importance for the design of oxygen masks. The six facial measurements shown at the right in Figure 8.5 are classified as being of high importance for oxygen masks. We use two different sets of measurements to evaluate our approach: first, the two measurements used by Amirav et al. leading to a 2D parameter space, and second, the six measurements by Lee, leading to a 6D parameter space.

8.5.2 Dynamic data analysis

This section evaluates the variations within the training data caused by motion. For each identity, we compute the axis-aligned bounding box covering all points in parameter space. This axis-aligned bounding box is computed as the difference of maximum and minimum values along each measurement dimension over all points of the identity in parameter space. For each identity the axis-aligned bounding box is the smallest possible parameter box that is able to cover the identity. Since for static data each identity consists of only a single point in parameter space, the side length of an axis-aligned bounding box for static data would be zero. The side length of the box measures the influence of the motion for dynamic motion data. We analyze the variation of the measurements due to motion by computing mean, standard deviation, median and maximum of the side lengths of the axis-aligned bounding boxes over all identities (see Table 8.1 for the 2D parameter space, and Table 8.2 for 6D, respectively). For both tables, the maximum values describe the minimum parameter box size necessary for a full covering of the parameter space to be computed.

Space dimension	3 boxes	5 boxes
2D	94.0	100.0
6D	74.0	82.0

Table 8.3: Percentage of covered training data with a fixed number of parameter boxes for 2D and 6D parameter space.

8.5.3 Space covering of training data

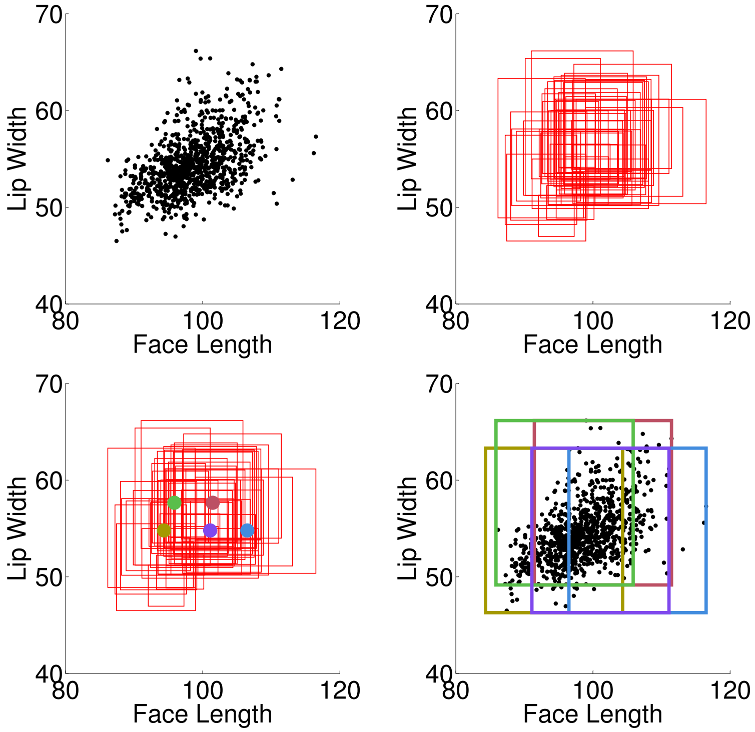


Figure 8.6: Overview of our parameter space covering approach. Upper left: Points in 2D parameter space. Upper right: Computed identity boxes I_{id}. Lower left: Full stabbing of identity boxes with 5 stabbing points. Lower right: Resulting covering in parameter space.

Given a fixed number of boxes, we want to get a good covering of the parameter space of the training data. We therefore choose the tolerances for the size of the box B based on the analysis of the training data from Section 8.5.2. For the covering of the 2D parameter space (at left in Figure 8.5) we choose tolerances of 20 mm for the face length and 17 mm for the lip width. Figure 8.6 shows the different steps of our covering method for the training data. The upper left of Figure 8.6 shows the training data in parameter space, where each identity is represented by up to 25 points. The upper right shows the identity boxes computed as described in Section 8.3. The lower left then shows the stabbing points for the identity boxes from Section 8.3.1. The lower right shows the resulting covering. For the covering of

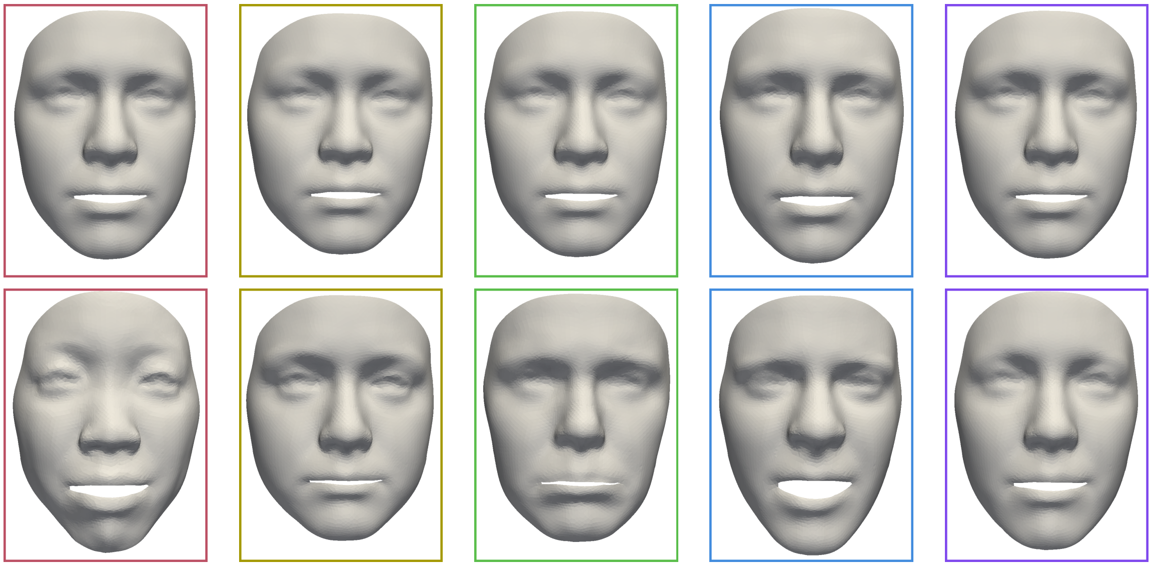


Figure 8.7: Representation of the motion space covering. Top: Procrustes mean shape for the five cover boxes for the 2D parameter space of the training data. Bottom: Faces from the training data closest to the box center in parameter space for the 2D parameter space of the training data.

the 6D parameter space, spanned by the measurements at the right in Figure 8.5, we choose the tolerances $1 = 20$ mm, $2 = 20$ mm, $3 = 5$ mm, $4 = 5$ mm, $5 = 10$ mm, and $6 = 17$ mm.

For both parameter spaces, we compute a covering with three and five boxes and measure the number of identities that are fully covered by these boxes (see Table 8.3). With three boxes, 94% of the identities in 2D parameter space are covered, and 74.0% of the 6D parameter space. With five boxes, all identities of the 2D parameter space are covered, and 82.0% of the 6D parameter space. Since for the 6D case the same number of points is embedded in a higher-dimensional parameter space, it is expected that more boxes are needed to cover the full space and that the same number of boxes cover a lower percentage of the data. Computing the full covering of both parameter spaces takes less than a second, running on a standard PC.

For each of the computed 2D cover boxes, we compute representative 3D face shapes as described in Section 8.4. First, for each box, we compute the full Procrustes mean over all identities fully covered by the box. The top of Figure 8.7 shows the full Procrustes mean for the five 2D cover boxes. Computing the full Procrustes mean leads to a good representation if the mean of the shapes used for computation is close to the box center. For our dynamic motion data a large amount of variation in parameter space is caused by the motion rather than by shape differences between different identities. Since identities need to be fully covered by boxes, the sizes of the boxes need to be large for data with large motion variations. With large boxes the overlap between different boxes is also large, and some identities are covered by multiple boxes. This causes the Procrustes mean shapes of different boxes to be similar.

Second, we find, for each box, the shape within the training database that is closest to the

Space dimension	3 boxes	5 boxes
2D	81.4	91.7
6D	58.3	64.6

Table 8.4: Generalization of the covering. Percentage of covered test data with the covering computed for the training data for 2D and 6D parameter space.

center of the box in parameter space. The bottom of Figure 8.7 shows the face shapes closest to the box centers in parameter space. Compared to the Procrustes mean shape, they are more distinctive and give a representative 3D geometry for the boxes.

8.5.4 Generalization of space covering

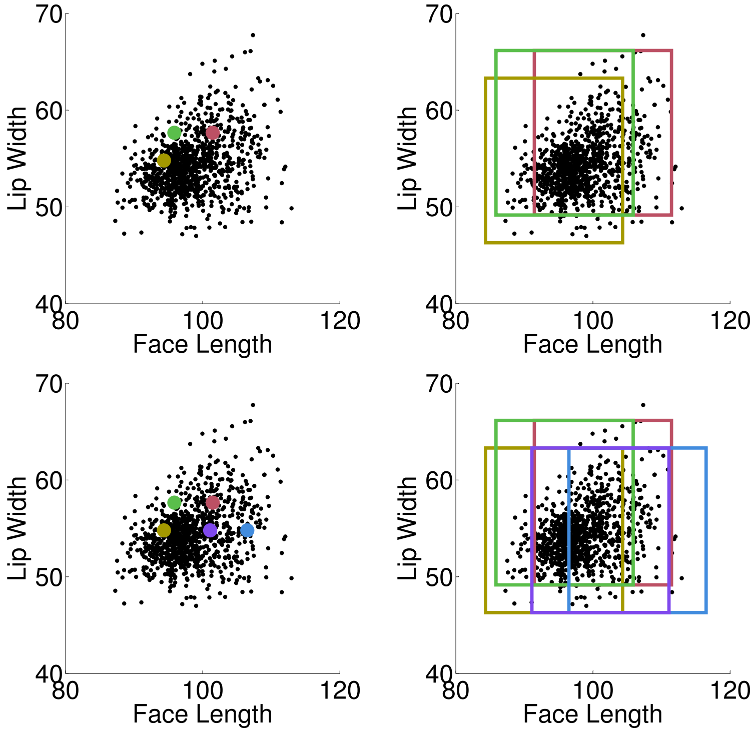


Figure 8.8: Covering applied to unseen data. Upper left: Midpoints of first three greedily selected cover boxes (stabbing points). Upper right: First three greedily selected cover boxes. Lower left: Midpoints of full training covering (stabbing points). Lower right: Full training covering.

In this section we evaluate how well the space covering computed for the training data from Section 8.5.3 generalizes to unseen data. Figure 8.8 shows in 2D parameter space the covering computed on the training data applied to the test data. The top row of Figure 8.8 shows the first three greedily selected stabbing points (left) and cover boxes (right); the bottom row shows the stabbing points (left) and cover boxes (right) of the full training covering.

To compute the generalization capability, we check, for each identity of the test data, whether it is fully covered by one of the training parameter boxes. An identity is fully covered by a parameter box if for that identity, all its points in parameter space are within the same box. Table 8.4 shows the covering rates for the test data. For three cover boxes, 81.4% of the test data identities are covered in 2D parameter space, and 58.3% in 6D parameter space. For five cover boxes, 91.7% of the test data identities are covered in 2D parameter space, and 64.6% in 6D parameter space. As for the covering of the training data, it is expected that the same number of boxes covers a lower percentage of the data in 6D than in 2D.

8.6 Summary

This chapter presented a general framework to compute a sizing system for dynamic motion data. This framework is one application of the registration methods developed in this thesis. We compute a covering of the low-dimensional parameter space with translated copies of a box of fixed size, defining the tolerances of a designed product along each measurement dimension. The covering is computed using a d-dimensional box stabbing method. We apply our framework to sets of anthropometric measurements used for the design of face masks, and evaluate our sizing system in terms of its ability to fit unseen data. For each size of the sizing system created, we compute a representative 3D geometry that can be used by designers to produce a prototype model.

While the sizing system computation in our framework is generally applicable for all kinds of measurements, our overall framework has some limitations. Our framework uses a registered database to compute a representative 3D face for each size. The registration methods developed throughout this thesis filter out effects of facial hair or other partial occlusions as caused e.g. by glasses. If the sizing system needs to consider facial hair or glasses, further data and different registration techniques are required.

Furthermore, we assume the tolerances for each measurement dimension to be independent and therefore to form a box in parameter space. If the tolerances are not independent, e.g. they form any other convex shape I_i in parameter space covering $\mathbf{p}_i$, the region I_{id} for each identity is given by an arbitrarily shaped convex object (intersection of I_i of all points). To obtain a sizing system for these tolerances, we would need to compute the stabbing of arbitrary-shaped convex shapes.

One very important point to test the benefit of our framework for designers would be to produce a real prototype of a face mask based on our computed sizing system for dynamic data. Producing a real prototype together with a user study to evaluate its quality in a real-world application is left for future work, since this would require an interdisciplinary study.

Chapter 9

Conclusion

"I did then what I knew how to do. Now that I know better, I do better."

– Maya Angelou

This chapter briefly summarizes the main contributions of this thesis and discusses some open problems and future work.

Closing remarks

This thesis has presented methods to statistically analyze static and dynamic 3D face data. The fundamental principle of our techniques is to exploit redundancies in the data for shape processing. The framework from Chapter 4 makes it possible to fully automatically register large databases of facial motion sequences. Due to the multilinear model used as statistical prior, the registration approach is robust to data corruptions, and it results in a compact representation of each motion sequence that is used for statistical analysis of the motion data.

While these global multilinear models represent the global face shape well, they are unable to represent geometric fine-scale details. To overcome this limitation, Chapter 5 introduced a novel localized multilinear model that effectively combines wavelet transform and multilinear models. This localized multilinear model preserves more fine-scale geometric details than global multilinear models do, while retaining robustness to various data corruptions when reconstructing static or dynamic face data.

These global and local multilinear models require all training faces to be in full correspondence. On the other hand, once such a multilinear model is learned, it can be used to register new face scans. Inspired by the minimum description length principle, Chapter 6 presented a groupwise multilinear correspondence optimization method that jointly optimizes vertex correspondence and learns a multilinear face model. Compared to existing PCA-based optimization methods this multilinear correspondence optimization method leads to correspondences of higher quality and is computationally more efficient, which makes it possible for the first time to apply an approach based on MDL to databases containing over 1000 shapes.

Previous methods to learn a multilinear model further degrade if not every person is captured in every expression, if face scans are noisy or partially occluded, or if expressions are erroneously labeled. To overcome these limitations, the groupwise framework from Chapter 7 makes it possible to robustly learn a multilinear face model from 3D face databases with missing data, corrupt data, wrong semantic correspondence, and inaccurate vertex correspondence. This robust model learning framework makes it possible to build highly accurate multilinear face models from databases that otherwise would not be usable for learning a multilinear model.

Finally, we presented an application of the registration methods developed throughout this thesis. Chapter 8 leveraged registered dynamic 3D face data to generate a sizing system applicable for the design of face masks. Our framework computes for each size a representative 3D shape that can be used by designers to produce a prototype model.

Open problems

This section summarizes some open problems and future research directions that relate to this thesis.

Open problem 1: Throughout this thesis we describe the deforming faces using multilinear models. It has been shown that such multilinear models can be used to capture facial performance from monocular video input [120]. Reconstructing a photorealistic 3D avatar from monocular video has recently attracted a lot of attention (see Section 2.5). The "Digital Emily Project" [2] creates a photorealistic digital virtual avatar with extensive manual effort. Despite the substantial advances in automating this process that have been achieved lately, the problem of capturing the entire facial performance (i.e. including the eyes and inner mouth region) from monocular video in real time in photorealistic quality remains unsolved to our knowledge.

Open problem 2: The multilinear models used throughout this thesis require full vertex correspondence across face shapes of different identities and different expressions. For one subject a dense correspondence can be defined naturally on pore-level resolution by tracking pores and freckles across multiple expressions [26]. However, a very dense correspondence across different subjects may not be semantically meaningful, as different subjects have different numbers of pores, freckles, etc. A similar observation has been made for human body shapes [17]. The global and local multilinear models proposed in this thesis have a resolution that is far from pore-level accuracy. To increase the accuracy of our methods to pore-level details or beyond therefore poses additional scientifically interesting questions and is not just a matter of increasing resolution. The reason is that it remains unclear how to obtain a high-quality registration with the required resolution.

Open problem 3: While the theoretical background of our methods is general, we only focus on 3D faces in this thesis. Hence, one open question is the performance of our methods on other data. Since our methods rely heavily on multilinear models, the data are assumed to be multilinearly distributed. This requires data with one or multiple sources of variations, where each source of variation can be modeled linearly. It has been shown that multilinear models can also be used e.g. to describe medical data like the prostate [73]. Our methods should work in these cases as well.

Chapter A

Appendix

"It is the story that matters not just the ending."

– Paul Lockhart

This chapter provides further details on the multilinear correspondence optimization approach proposed in Chapter 6. Section A.1 provides more details on how the objective function that measures the model quality depending on correspondences is optimized. Further, the computation complexities of the multilinear model and existing linear approaches are compared in Section A.2.

A.1 Formulation of registration optimization

The objective function E in Equation 6.1 is analytically differentiable with respect to the 2D shape parameters $\boldsymbol{\alpha}$. Let $\mathbf{x}_{ie} \in \mathbb{R}^{3n}$ denote the face of identity i in expression e that consists of n vertices. To simplify notation, whenever a fixed shape is used, we omit the subscripts ie.

For a fixed shape $\mathbf{x}$, the gradient $\frac{\partial E}{\partial \boldsymbol{\alpha}_k} \in \mathbb{R}^2$ of E with respect to the parameters $\boldsymbol{\alpha}_k$ of vertex $\mathbf{v}_k(\mathbf{x})$ is

$$\frac{\partial E}{\partial \boldsymbol{\alpha}_k} = \frac{\partial \mathbf{x}}{\partial \boldsymbol{\alpha}_k} \frac{\partial E_C}{\partial \mathbf{x}} + w_R \frac{\partial \mathbf{x}}{\partial \boldsymbol{\alpha}_k} \frac{\partial E_R}{\partial \mathbf{x}}. \tag{A.1}$$

In the following, we provide derivations for these partial derivatives. Section A.1.1 derives $\frac{\partial E_R}{\partial \mathbf{x}}$ and gives the result in Equation A.4, Section A.1.2 derives $\frac{\partial E_C}{\partial \mathbf{x}}$ and gives the result in Equation A.5, and Section A.1.3 derives $\frac{\partial \mathbf{x}}{\partial \boldsymbol{\alpha}_k}$ and gives the result in Equation A.17.

A.1.1 Regularization derivative

This section gives the derivative of E_R (Eq. 6.3) with respect to the shape $\mathbf{x}$. To compute the derivative $\frac{\partial E_R}{\partial \mathbf{x}}$, we first define a block matrix $\mathbf{S} \in \mathbb{R}^{3n \times 3n}$. One submatrix $\mathbf{S}_{jk} \in \mathbb{R}^{3 \times 3}$ of $\mathbf{S}$ is defined as

$$\mathbf{S}_{jk} = \begin{cases} -\mathbf{I} & j = k \\ \frac{1}{|N(\mathbf{p}_j)|}\mathbf{I} & \mathbf{p}_k \in N(\mathbf{p}_j) \\ \mathbf{0} & otherwise, \end{cases} \tag{A.2}$$

where $\mathbf{I} \in \mathbb{R}^{3\times 3}$ denotes the identity matrix.

The energy E_R can then be expressed as

$$E_R = \frac{1}{n} (\mathbf{SSx})^T (\mathbf{SSx}). \tag{A.3}$$

The derivative $\frac{\partial E_R}{\partial \mathbf{x}}$ is then

$$\frac{\partial E_R}{\partial \mathbf{x}} = \frac{2}{n} (\mathbf{SS})^T (\mathbf{SS})\, \mathbf{x}. \tag{A.4}$$

A.1.2 Compactness derivative

This section gives the derivative of E_C (Eq. 6.2) with respect to the shape $\mathbf{x}$. For PCA models, the compactness energy is measured by the trace of the covariance matrix. Instead of minimizing the trace of the covariance matrix, Kotcheff and Taylor [81] minimize the logarithm of the determinant of the covariance matrix. They show that minimizing the logarithm of the determinant of the covariance matrix leads to a better model than minimizing the trace of the covariance matrix. This observation can be explained because the resulting energy is a log-sum penalty function on the eigenvalues of the covariance matrix, which is known to encourage sparsity [32] as small eigenvalues are heavily punished.

Due to the different centering, the extension of the PCA compactness gradient to the multilinear case is not straightforward. The derivative of E_C with respect to the shape $\mathbf{x}$ is by the chain rule

$$\frac{\partial E_C}{\partial \mathbf{x}} = \frac{1}{d_2}\sum_{a=1}^{d_2} \underbrace{\frac{\partial \lambda_a^{(2)}}{\partial \mathbf{x}}}_{Eq.\ A.10} \underbrace{\frac{\partial E_{C,a}^{(2)}}{\partial \lambda_a^{(2)}}}_{Eq.\ A.6} + \frac{1}{d_3}\sum_{a=1}^{d_3} \underbrace{\frac{\partial \lambda_a^{(3)}}{\partial \mathbf{x}}}_{Eq.\ A.13} \underbrace{\frac{\partial E_{C,a}^{(3)}}{\partial \lambda_a^{(3)}}}_{Eq.\ A.12}, \tag{A.5}$$

with different partial derivatives for the identity mode and the expression mode.
Mode-2: The derivative for mode-2 is

$$\frac{\partial E_{C,a}^{(2)}}{\partial \lambda_a^{(2)}} = \frac{1}{\lambda_a^{(2)} + \delta_2}, \tag{A.6}$$

and by the chain rule

$$\frac{\partial \lambda_a^{(2)}}{\partial \mathbf{x}} = \sum_{j=1}^{d_2}\sum_{k=1}^{d_2} \frac{\partial \mathbf{D}^{(2)}[j,k]}{\partial \mathbf{x}} \frac{\partial \lambda_a^{(2)}}{\partial \mathbf{D}^{(2)}[j,k]}, \tag{A.7}$$

where $\mathbf{D}^{(2)}[j,k] \in \mathbb{R}$ denotes the element of row j and column k of the mode-2 covariance matrix $\mathbf{D}^{(2)}$.

It follows from infinitesimal considerations [43, Appendix B.2] that the partial derivative of the eigenvalue $\lambda_a^{(2)}$ with respect to the element of the covariance matrix $\mathbf{D}^{(2)}[j,k]$ is

$$\frac{\partial \lambda_a^{(2)}}{\partial \mathbf{D}^{(2)}[j,k]} = \mathbf{e}_a^{(2)}[j]\mathbf{e}_a^{(2)}[k], \tag{A.8}$$

where $\mathbf{e}_a^{(2)}[j]$ denotes the j-th element, and $\mathbf{e}_a^{(2)}[k]$ the k-th element of the corresponding eigenvector of $\lambda_a^{(2)}$.

The partial derivative of the element of the covariance matrix $\mathbf{D}^{(2)}[j,k]$ with respect to the shape $\mathbf{x}$ is computed with the chain rule as

$$\frac{\partial \mathbf{D}^{(2)}[j,k]}{\partial \mathbf{x}} = \sum_{l=1}^{d_2}\sum_{m=1}^{d_3} \frac{\partial \mathbf{c}_{lm}}{\partial \mathbf{x}} \frac{\partial \mathbf{D}^{(2)}[j,k]}{\partial \mathbf{c}_{lm}}. \tag{A.9}$$

This leads to the overall partial derivative of the eigenvalue $\lambda_a^{(2)}$ with respect to the shape $\mathbf{x}_{ie}$. That is,

$$\frac{\partial \lambda_a^{(2)}}{\partial \mathbf{x}_{ie}} = \frac{2}{d_2 d_3 d_3} \sum_{j=1}^{d_2} \left(\mathbf{e}_a^{(2)}[j] \sum_{k=1}^{d_2} \left(\mathbf{e}_a^{(2)}[k] \sum_{m=1}^{d_3} \Psi_{ik}^{em} \mathbf{c}_{jm} \right)\right), \tag{A.10}$$

where

$$\Psi_{ik}^{em} = \begin{cases} d_2 d_3 - 1 & i = k \text{ and } e = m \\ -1 & \text{otherwise,} \end{cases} \tag{A.11}$$

and where $\mathbf{e}_a^{(2)}[j]$ denotes the j-th element, and $\mathbf{e}_a^{(2)}[k]$ the k-th element of the corresponding eigenvector of $\lambda_a^{(2)}$.

Mode-3: The derivative for mode-3 is

$$\frac{\partial E_{C,a}^{(3)}}{\partial \lambda_a^{(3)}} = \frac{1}{\lambda_a^{(3)} + \delta_3}, \tag{A.12}$$

and using a similar argument as above,

$$\frac{\partial \lambda_a^{(3)}}{\partial \mathbf{x}_{ie}} = \frac{2}{d_2 d_3 d_2} \sum_{j=1}^{d_3} \left(\mathbf{e}_a^{(3)}[j] \sum_{k=1}^{d_3} \left(\mathbf{e}_a^{(3)}[k] \sum_{m=1}^{d_2} \Psi_{im}^{ek} \mathbf{c}_{mj} \right)\right), \tag{A.13}$$

where

$$\Psi_{im}^{ek} = \begin{cases} d_2 d_3 - 1 & i = m \text{ and } e = k \\ -1 & \text{otherwise,} \end{cases} \tag{A.14}$$

and where $\mathbf{e}_a^{(3)}[j]$ denotes the j-th element, and $\mathbf{e}_a^{(3)}[k]$ the k-th element of the corresponding eigenvector of $\lambda_a^{(3)}$.

A.1.3 Parametrization derivative

This section derives $\frac{\partial \mathbf{x}}{\partial \boldsymbol{\alpha}_k}$. Recall that any kind of continuous mapping can be used to parametrize the shape $\mathbf{x}$ in 2D. We establish a continuous mapping from a 3D face to a 2D unit square by a thin-plate spline [45]. For other mappings, the term $\frac{\partial \mathbf{x}}{\partial \boldsymbol{\alpha}} \in \mathbb{R}^{2n \times 3n}$ of the derivative changes, while the rest of the gradient stays unchanged.

A thin-plate spline is computed for each shape as

$$\Phi(\boldsymbol{\alpha}) = \mathbf{c} + \mathbf{A}\boldsymbol{\alpha} + \mathbf{W}^T(\sigma(\boldsymbol{\alpha} - \mathbf{t}_1), \ldots, \sigma(\boldsymbol{\alpha} - \mathbf{t}_n))^T, \tag{A.15}$$

where $\mathbf{c} \in \mathbb{R}^3$, $\mathbf{A} \in \mathbb{R}^{3\times 2}$, and $\mathbf{W} \in \mathbb{R}^{n\times 3}$ are the parameters of the mapping, and where $\sigma : \mathbb{R}^2 \to \mathbb{R}$ is the function

$$\sigma(\mathbf{h}) = \begin{cases} \|\mathbf{h}\|^2 \log(\|\mathbf{h}\|) & \|\mathbf{h}\| > 0, \\ 0 & \|\mathbf{h}\| = 0, \end{cases} \tag{A.16}$$

where $\|\mathbf{h}\|$ is the Euclidean length of $\mathbf{h}$.

To find the derivative $\frac{\partial \mathbf{x}}{\partial \boldsymbol{\alpha}_k}$, we can compute $\frac{\partial \mathbf{v}_b(\mathbf{x})}{\partial \boldsymbol{\alpha}_k}$ for every vertex $\mathbf{v}_b(\mathbf{x})$ and combine the resulting derivatives. It is

$$\frac{\partial \mathbf{v}_b(\mathbf{x})}{\partial \boldsymbol{\alpha}_k} = \frac{\partial \Phi(\boldsymbol{\alpha}_b)}{\partial \boldsymbol{\alpha}_k} = \begin{cases} \mathbf{A}^T + (\frac{\partial \sigma(\boldsymbol{\alpha}_b) - \mathbf{t}_1)}{\partial \boldsymbol{\alpha}_k} \cdots \frac{\partial \sigma(\boldsymbol{\alpha}_b) - \mathbf{t}_n)}{\partial \boldsymbol{\alpha}_k})\mathbf{W} & x_k = x_b \\ \mathbf{0} & x_k \neq x_b \end{cases} \tag{A.17}$$

with

$$\frac{\partial \sigma(\boldsymbol{\alpha} - \mathbf{t}_l)}{\partial \boldsymbol{\alpha}} = \begin{cases} (\boldsymbol{\alpha} - \mathbf{t}_l)(2 \log \|\boldsymbol{\alpha} - \mathbf{t}_l\| + 1) & \|\boldsymbol{\alpha} - \mathbf{t}_l\| > 0 \\ \mathbf{0} & \|\boldsymbol{\alpha} - \mathbf{t}_l\| = 0. \end{cases} \tag{A.18}$$

A.2 Comparison of computational complexities

This section gives the computational complexities for the correspondence optimization for the multilinear case and the linear case ($d_3 = 1$). For both cases, we use that computing the singular values and singular vectors of a $m \times n$ matrix takes $O(mn^2 + n^3)$ time [129, Chapter 31].

A.2.1 Multilinear registration optimization

Derivative: The partial derivatives of the eigenvalues $\lambda_a^{(2)}$ and $\lambda_a^{(3)}$ with respect to the shape $\mathbf{x}_{ie}$ (Equations A.10 and A.13) can be reformulated as

$$\frac{\partial \lambda_a^{(2)}}{\partial \mathbf{x}_{ie}} = \underbrace{\frac{2}{d_3} \mathbf{e}_a^{(2)}[i] \sum_{j=1}^{d_2} \mathbf{e}_a^{(2)}[j] \mathbf{c}_{je}}_{O(nd_2)} - \frac{2}{d_2 d_3 d_3} \underbrace{\left(\sum_{k=1}^{d_2} \mathbf{e}_a^{(2)}[k] \right)}_{O(d_2)} \underbrace{\left(\sum_{j=1}^{d_2} \sum_{m=1}^{d_3} \mathbf{e}_a^{(2)}[j] \mathbf{c}_{jm} \right)}_{O(nd_2 d_3)}, \tag{A.19}$$

and

$$\frac{\partial \lambda_a^{(3)}}{\partial \mathbf{x}_{ie}} = \underbrace{\frac{2}{d_2} \mathbf{e}_a^{(3)}[e] \sum_{j=1}^{d_3} \mathbf{e}_a^{(3)}[j] \mathbf{c}_{ij}}_{O(nd_3)} - \frac{2}{d_2 d_3 d_2} \underbrace{\left(\sum_{k=1}^{d_3} \mathbf{e}_a^{(3)}[k] \right)}_{O(d_3)} \underbrace{\left(\sum_{j=1}^{d_3} \sum_{m=1}^{d_2} \mathbf{e}_a^{(3)}[j] \mathbf{c}_{mj} \right)}_{O(nd_2 d_3)}. \tag{A.20}$$

Computational complexity: The gradient of E_C with respect to the shape **x** (Equation A.5) can therefore be computed in time $O(nd_2^2d_3 + nd_2d_3^2)$. Computing the eigenvalues and eigenvectors of the mode-2 and mode-3 covariance matrices $\mathbf{D}^{(2)}$ and $\mathbf{D}^{(3)}$ takes $O(nd_2^2d_3 + d_2^3)$ and $O(nd_2d_3^2 + d_3^3)$ time, respectively. This leads to the overall computational complexity of $O(nd_2^2d_3 + nd_2d_3^2 + d_2^3 + d_3^3)$.

Assuming $n \gg d_2, d_3$ the complexity becomes $O(nd_2^2d_3 + nd_2d_3^2)$.

A.2.2 Linear registration optimization

Derivative: Most previous methods use a linear model for correspondence optimization (see e.g. Davies et al. [43, Chapter 4]). Note that the linear model is a special case of our multilinear approach, where the multilinear model degenerates to the linear model for $d_2 = 1$ or $d_3 = 1$. We use a consistent notation to previous sections but omit the subscripts and superscripts for mode 2 and mode 3, since only one mode is present in the linear case.

Let d denote the number of shapes, where $\mathbf{x}_j \in \mathbb{R}^{3n}$ denotes the j-th shape, and $\mathbf{c}_j \in \mathbb{R}^{3n}$ denotes the j-th centered shape. The derivative of E_C with respect to the shape **x** is

$$\frac{\partial E_C}{\partial \mathbf{x}} = \frac{1}{d}\sum_{a=1}^{d} \frac{\partial \lambda_a}{\partial \mathbf{x}} \frac{\partial E_{C,a}}{\partial \lambda_a}. \tag{A.21}$$

From the centering of the data ($\sum_{j=1}^{d} \mathbf{c}_j = 0$) follows

$$\sum_{j=1}^{d} \mathbf{e}_a[j] = 0. \tag{A.22}$$

The partial derivative of the eigenvalue λ_a with respect to the shape $\mathbf{x}_j$ is therefore

$$\frac{\partial \lambda_a}{\partial \mathbf{x}_j} = 2\mathbf{e}_a[j]\sum_{k=1}^{d} \mathbf{e}_a[k]\mathbf{c}_k. \tag{A.23}$$

Computational complexity: The partial derivative of λ_a with respect to $\mathbf{x}_j$ (Equation A.23) can be computed in time $O(nd)$. The gradient of E_C with respect to **x** (Equation A.21) can therefore be computed in time $O(nd^2)$. Computing the eigenvalues and eigenvectors of the covariance matrix **D** takes $O(nd^2 + d^3)$ time. This leads to the overall computational complexity of $O(nd^2 + d^3)$.

Assuming $n \gg d$ the complexity becomes $O(nd^2)$.

A.2.3 Comparison

For both existing linear methods and our method, the minimum description length optimization is non-linear and solved with the help of optimizers that require an explicit gradient computation in each iteration. Hence, the computational complexity of the gradient computation has a strong influence on the overall run time of the optimization.

For the same number of shapes $d = d_2 d_3$, it takes time $O(nd_2^2 d_3^2)$ to compute a gradient for existing linear methods. Our multilinear model, where a gradient computation takes time $O(nd_2^2 d_3 + nd_2 d_3^2)$, is significantly more efficient if both d_2 and d_3 are large.

Bibliography

[1] O. Aldrian and W.A.P. Smith. Inverse rendering of faces with a 3D morphable model. *Transactions on Pattern Analysis and Machine Intelligence*, 35(5):1080–1093, 2013.

[2] O. Alexander, M. Rogers, W. Lambeth, M. Chiang, and P. Debevec. The digital emily project: Photoreal facial modeling and animation. In *SIGGRAPH 2009 Courses*, pages 12:1–12:15, 2009.

[3] B. Allen, B. Curless, and Z. Popović. The space of human body shapes: reconstruction and parameterization from range scans. *Transactions on Graphics (Proceedings of SIGGRAPH)*, 22(3):587–594, 2003.

[4] B. Amberg, R. Knothe, and T. Vetter. Expression invariant 3D face recognition with a morphable model. In *International Conference on Automatic Face Gesture Recognition*, pages 1–6, 2008.

[5] B. Amberg, P. Paysan, and T. Vetter. Weight, sex, and facial expressions: On the manipulation of attributes in generative 3D face models. In *International Symposium on Advances in Visual Computing: Part I*, pages 875–885, 2009.

[6] B. Amberg, S. Romdhani, and T. Vetter. Optimal step nonrigid ICP algorithms for surface registration. In *Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.

[7] I. Amirav, A. S. Luder, A. Halamish, D. Raviv, R. Kimmel, D. Waisman, and M. T. Newhouse. Design of aerosol face masks for children using computerized 3D face analysis. *Journal of Aerosol Medicine and Pulmonary Drug Delivery*, 26(0):1–7, 2013.

[8] ANN. `http://www.cs.umd.edu/~mount/ANN/`.

[9] M. A. Balkhyour. Evaluation of full-facepiece respirator fit on fire fighters in the municipality of Jeddah, Saudi Arabia. *International Journal of Environmental Research and Public Health*, 10(1):347–360, 2013.

[10] T. Beeler, F. Hahn, D. Bradley, B. Bickel, P. Beardsley, C. Gotsman, R. W. Sumner, and M. Gross. High-quality passive facial performance capture using anchor frames. *Transactions on Graphics (Proceedings of SIGGRAPH)*, 30(4):75:1–75:10, 2011.

[11] S. Berretti, B. Ben Amor, M. Daoudi, and A. Bimbo. 3D facial expression recognition using sift descriptors of automatically detected keypoints. *The Visual Computer*, 27(11):1021–1036, 2011.

[12] M. Bertram, M. A. Duchaineau, B. Hamann, and K. I. Joy. Generalized b-spline subdivision-surface wavelets for geometry compression. *Transactions on Visualization and Computer Graphics*, 10(3):326–338, 2004.

[13] P. J. Besl. Active, optical range imaging sensors. *Machine Vision and Applications*, 1(2):127–152, 1988.

[14] B. Bickel, P. Kaufmann, M. Skouras, B. Thomaszewski, D. Bradley, T. Beeler, P. Jackson, S. Marschner, W. Matusik, and M. Gross. Physical face cloning. *Transactions on Graphics (Proceedings of SIGGRAPH)*, 31(4):118:1–118:10, 2012.

[15] V. Blanz, K. Scherbaum, and H.-P. Seidel. Fitting a morphable model to 3D scans of faces. In *International Conference on Computer Vision*, pages 1–8, 2007.

[16] V. Blanz and T. Vetter. A morphable model for the synthesis of 3D faces. In *SIGGRAPH*, pages 187–194, 1999.

[17] F. Bogo, J. Romero, M. Loper, and M.J. Black. Faust: Dataset and evaluation for 3D mesh registration. In *Conference on Computer Vision and Pattern Recognition*, pages 3794–3801, 2014.

[18] T. Bolkart, A. Brunton, A. Salazar, and S. Wuhrer. Statistical 3D shape models of human faces, 2013. `http://statistical-face-models.mmci.uni-saarland.de/`.

[19] T. Bolkart and S. Wuhrer. Statistical analysis of 3D faces in motion. In *International Conference on 3D Vision*, pages 103–110, 2013.

[20] T. Bolkart and S. Wuhrer. A general framework to generate sizing systems from 3D motion data applied to face mask design. In *International Conference on 3D Vision*, pages 425–431, 2014.

[21] T. Bolkart and S. Wuhrer. 3D faces in motion: Fully automatic registration and statistical analysis. *Computer Vision and Image Understanding*, 131:100–115, 2015.

[22] T. Bolkart and S. Wuhrer. A groupwise multilinear correspondence optimization for 3D faces. In *International Conference on Computer Vision*, pages 3604–3612, 2015.

[23] T. Bolkart and S. Wuhrer. Multilinear MDL for 3D faces, 2015. http://multilinear-mdl.gforge.inria.fr/.

[24] T. Bolkart and S. Wuhrer. A robust multilinear model learning framework for 3D faces. In *Conference on Computer Vision and Pattern Recognition*, page to appear, 2016.

[25] S. Bouaziz, Y. Wang, and M. Pauly. Online modeling for realtime facial animation. *Transactions on Graphics (Proceedings of SIGGRAPH)*, 32(4):40:1–40:10, 2013.

[26] D. Bradley, W. Heidrich, T. Popa, and A. Sheffer. High resolution passive facial performance capture. *Transactions on Graphics (Proceedings of SIGGRAPH)*, 29(4):41:1–41:10, 2010.

[27] A. Bronstein, M. Bronstein, and R. Kimmel. *Numerical Geometry of Non-Rigid Shapes*. Springer, 2008.

[28] A. Brunton, T. Bolkart, and S. Wuhrer. Multilinear wavelets: A statistical shape space for human faces. In *European Conference on Computer Vision*, pages 297–312, 2014.

[29] A. Brunton, A. Salazar, T. Bolkart, and S. Wuhrer. Review of statistical shape spaces for 3D data with comparative analysis for human faces. *Computer Vision and Image Understanding*, 128(0):1–17, 2014.

[30] A. Brunton, C. Shu, J. Lang, and E. Dubois. Wavelet model-based stereo for fast, robust face reconstruction. In *Canadian Conference on Computer and Robot Vision*, pages 347–354, 2011.

[31] O. Burghard, A. Berner, M. Wand, N. J. Mitra, H.-P. Seidel, and R. Klein. Compact part-based shape spaces for dense correspondences. *CoRR*, abs/1311.7535, 2013.

[32] E.J. Candès, M.B. Wakin, and S.P. Boyd. Enhancing sparsity by reweighted l_1 minimization. *JFAA*, 14(5–6):877–905, 2008.

[33] C. Cao, D. Bradley, K. Zhou, and T. Beeler. Real-time high-fidelity facial performance capture. *Transactions on Graphics (Proceedings of SIGGRAPH)*, 34(4):46:1–46:9, 2015.

[34] C. Cao, Q. Hou, and K. Zhou. Displaced dynamic expression regression for real-time facial tracking and animation. *Transactions on Graphics (Proceedings of SIGGRAPH)*, 33(4):43:1–43:10, 2014.

[35] C. Cao, Y. Weng, S. Lin, and K. Zhou. 3D shape regression for real-time facial animation. *Transactions on Graphics (Proceedings of SIGGRAPH)*, 32(4):41:1–41:10, 2013.

[36] J. D. Carroll and J. J. Chang. Analysis of individual differences in multidimensional scaling via an N-way generalization of ”Eckart-Young” decomposition. *Psychometrika*, 35(3):283–319, 1970.

[37] E. Catmull and J. Clark. Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer-Aided Design*, 10(6):350–355, 1978.

[38] O. Çeliktutan, S. Ulukaya, and B. Sankur. A comparative study of face landmarking techniques. *Eurasip Journal on Image and Video Processing*, 2013(1):1–27, 2013.

[39] J.-H. Chen, K. C. Zheng, and L. G. Shapiro. 3D point correspondence by minimum description length in feature space. In *European Conference on Computer Vision*, pages 621–634, 2010.

[40] D. Cosker, E. Krumhuber, and A. Hilton. A FACS valid 3D dynamic action unit database with applications to 3D dynamic morphable facial modeling. In *International Conference on Computer Vision*, pages 2296–2303, 2011.

[41] C. Creusot, N. Pears, and J. Austin. A machine-learning approach to keypoint detection and landmarking on 3D meshes. *International Journal of Computer Vision*, 102(1-3):146–179, 2013.

[42] K. Dale, K. Sunkavalli, M. K. Johnson, D. Vlasic, W. Matusik, and H. Pfister. Video face replacement. *Transactions on Graphics (Proceedings of SIGGRAPH Asia)*, 30(6):130:1–10, 2011.

[43] R. Davies, C. Twining, and C. Taylor. *Statistical Models of Shape: Optimisation and Evaluation*. Springer, 2008.

[44] R.H. Davies, C.J. Twining, T.F. Cootes, and C.J. Taylor. Building 3-D statistical shape models by direct optimization. *Transactions on Medical Imaging*, 29(4):961–981, 2010.

[45] Ian L. Dryden and Kanti V. Mardia. *Statistical shape analysis*. Wiley, 1998.

[46] N. Dyn, D. Levine, and J. A. Gregory. A butterfly subdivision scheme for surface interpolation with tension control. *Transactions on Graphics*, 9(2):160–169, 1990.

[47] P. Ekman. *Emotions Revealed: Recognizing Faces and Feelings to Improve Communication and Emotional Life*. Henry Holt and Company, 2003.

[48] P. Ekman and W. Friesen. *Facial Action Coding System: A Technique for the Measurement of Facial Movement*. Consulting Psychologists Press, 1978.

[49] L. Eldén and B. Savas. A newton-grassmann method for computing the best multilinear rank-(r_1, r_2, r_3) approximation of a tensor. *SIAM Journal on Matrix Analysis and Applications*, 31(2):248–271, 2009.

[50] T. Fang, X. Zhao, O. Ocegueda, S. K. Shah, and I. A. Kakadiaris. 3D/4D facial expression analysis: An advanced annotated face model approach. *Image and Vision Computing*, 30(10):738–749, 2012.

[51] C. Ferrari, G. Lisanti, S. Berretti, and A. Del Bimbo. Dictionary learning based 3D morphable model construction for face recognition with varying expression and pose. In *International Conference on 3D Vision*, pages 509–517, 2015.

[52] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[53] R.J. Fowler, M.S. Paterson, and S.L. Tanimoto. Optimal packing and covering in the plane are NP-complete. *Information Processing Letters*, 12(3):133–137, 1981.

[54] P. Garrido, L. Valgaerts, C. Wu, and C. Theobalt. Reconstructing detailed dynamic face geometry from monocular video. *Transactions on Graphics (Proceedings of SIGGRAPH Asia)*, 32(6):158:1–158:10, 2013.

[55] S.Z. Gilani, F. Shafait, and A. Mian. Shape-based automatic detection of a large number of 3D facial landmarks. In *Conference on Computer Vision and Pattern Recognition*, pages 4639–4648, 2015.

[56] S. T. Gollmer, M. Kirschner, T. M. Buzug, and S. Wesarg. Using image segmentation for evaluating 3D statistical shape models built with groupwise correspondence optimization. *Computer Vision and Image Understanding*, 125(0):283–303, 2014.

[57] A. Golovinskiy, W. Matusik, H. Pfister, S. Rusinkiewicz, and T. Funkhouser. A statistical model for synthesis of detailed facial geometry. *Transactions on Graphics (Proceedings of SIGGRAPH)*, 25(3):1025–1034, 2006.

[58] J. Guo, X. Mei, and K. Tang. Automatic landmark annotation and dense correspondence registration for 3D human facial images. *BMC Bioinformatics*, 14(1):1–12, 2013.

[59] P. Hammond, C. Foster-Gibson, A. E. Chudley, J. E. Allanson, T. J. Hutton, S. A. Farrell, J. McKenzie, J. J. A. Holden, and M. E. S. Lewis. Face-brain asymmetry in autism spectrum disorders. *Molecular Psychiatry*, 13(6):614–623, 2008.

[60] D.-H. Han, J. Rhi, and J. Lee. Development of prototypes of half-mask facepieces for Koreans using the 3D digitizing design method: A pilot study. *Annals of Occupational Hygiene*, 48(8):707–714, 2004.

[61] R. A. Harshman. Foundations of the PARAFAC procedure: Models and conditions for an ”explanatory” multi-modal factor analysis. *UCLA Working Papers in Phonetics*, 16(1):1–84, 1970.

[62] T. Hierl, S. Arnold, D. Kruber, F.-P. Schulze, and H. Hümpfner-Hierl. CAD-CAM-assisted esthetic facial surgery. *Journal of Oral and Maxillofacial Surgery*, 71(1):15–23, 2013.

[63] C. J. Hillar and L.-H. Lim. Most tensor problems are NP-hard. *Journal of the ACM*, 60(6):45:1–45:39, 2013.

[64] D.A. Hirshberg, M. Loper, E. Rachlin, and M.J. Black. Coregistration: Simultaneous alignment and modeling of articulated 3D shape. In *European Conference on Computer Vision*, pages 242–255, 2012.

[65] F. L. Hitchcock. The expression of a tensor or a polyadic as a sum of products. *Journal of Mathematics and Physics*, 6(1):164–189, 1927.

[66] P.-L. Hsieh, C. Ma, J. Yu, and H. Li. Unconstrained realtime facial performance capture. In *Conference on Computer Vision and Pattern Recognition*, pages 1675–1683, 2015.

[67] J. Hu and Q. Xie. Mesh denoising based on spherical wavelets in reverse engineering. *Applied Mechanics and Materials*, 232:904–907, 2012.

[68] Y. Hu, M. Zhou, and Z. Wu. An automatic non-rigid point matching method for dense 3D face scans. In *International Conference on Computational Science and Its Applications*, pages 215–221, 2009.

[69] Y. Huang, X. Zhang, Y. Fan, L. Yin, L. Seversky, J. Allen, T. Lei, and W. Dong. Reshaping 3D facial scans for facial appearance modeling and 3D facial expression analysis. *Image and Vision Computing*, 30(10):750–761, 2012.

[70] A. E. Ichim, S. Bouaziz, and M. Pauly. Dynamic 3D avatar creation from hand-held video input. *Transactions on Graphics (Proceedings of SIGGRAPH)*, 34(4):45:1–45:14, 2015.

[71] 3DMD Inc. `http://www.3dmd.com/`.

[72] DI4D Inc. `http://www.di4d.com/`.

[73] Y. Jeong, R. J. Radke, and D. M. Lovelock. Bilinear models for inter- and intra-patient variation of the prostate. *Physics in Medicine and Biology*, 55(13):3725–3739, 2010.

[74] A. E. Johnson and M. Hebert. Recognizing objects by matching oriented points. In *Conference on Computer Vision and Pattern Recognition*, pages 684–692, 1997.

[75] I.A. Kakadiaris, G. Passalis, T. Theoharis, G. Toderici, I. Konstantinidis, and N. Murtuza. Multimodal face recognition: combination of geometry with physiological information. In *Conference on Computer Vision and Pattern Recognition*, pages 1022–1029, 2005.

[76] A. Kapteyn, H. Neudecker, and T. Wansbeek. An approach to n-mode components analysis. *Psychometrika*, 51(2):269–275, 1986.

[77] V. Kazemi, C. Keskin, J. Taylor, P. Kohli, and S. Izadi. Real-time face reconstruction from a single depth image. In *International Conference on 3D Vision*, pages 369–376, 2014.

[78] Microsoft Kinect. `http://www.xbox.com/en-GB/kinect`.

[79] L. Kobbelt, S. Campagna, J. Vorsatz, and H.-P. Seidel. Interactive multi-resolution modeling on arbitrary meshes. In *SIGGRAPH*, pages 105–114, 1998.

[80] T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500, 2009.

[81] A. C.W. Kotcheff and C. J. Taylor. Automatic construction of eigenshape models by direct optimization. *Medical Image Analysis*, 2(4):303–314, 1998.

[82] O. Kwon, K. Jung, H. You, and H.-E. Kim. Determination of key dimensions for a glove sizing system by analyzing the relationships between hand dimensions. *Applied Ergonomics*, 40(4):762–766, 2009.

[83] L. De Lathauwer. *Signal processing based on multilinear algebra*. PhD thesis, K.U. Leuven, Belgium, 1997.

[84] V. Le, H. Tang, and T. S. Huang. Expression recognition from 3D dynamic faces using robust spatio-temporal shape features. In *International Conference on Automatic Face and Gesture Recognition and Workshops*, pages 414–421, 2011.

[85] D.T. Lee and C.K. Wong. Worst-case analysis for region and partial region searches in multidimensional binary search trees and balanced quad trees. *Acta Informatica*, 9(1):23–29, 1977.

[86] W. Lee. *Development of a Design Methodology of Pilot Oxygen Mask Using 3D Facial Scan Data*. PhD thesis, Pohang University of Science and Technology, Korea, 2013.

[87] D. Lefloch, R. Nair, F. Lenzen, H. Schäfer, L. Streeter, M. J. Cree, R. Koch, and A. Kolb. *Time-of-Flight and Depth Imaging. Sensors, Algorithms, and Applications*, chapter Technical Foundation and Calibration Methods for Time-of-Flight Cameras, pages 3–24. Springer Berlin Heidelberg, 2013.

[88] J. P. Lewis, K. Anjyo, T. Rhee, M. Zhang, F. Pighin, and Z. Deng. Practice and theory of blendshape facial models. In *Eurographics - State of the Art Reports*, 2014.

[89] H. Li, . Trutoiu, K. Olszewski, L. Wei, T. Trutna, P.-L. Hsieh, A. Nicholls, and C. Ma. Facial performance sensing head-mounted display. *Transactions on Graphics (Proceedings of SIGGRAPH)*, 34(4):47:1–47:9, 2015.

[90] H. Li, T. Weise, and M. Pauly. Example-based facial rigging. *Transactions on Graphics (Proceedings of SIGGRAPH)*, 29(4):32:1–32:6, 2010.

[91] H. Li, J. Yu, Y. Ye, and C. Bregler. Realtime facial animation with on-the-fly correctives. *Transactions on Graphics (Proceedings of SIGGRAPH)*, 32(4):42:1–42:10, 2013.

[92] D.C. Liu and J. Nocedal. On the limited memory method for large scale optimization. *Mathematical Programming: Series B*, 45(3):503–528, 1989.

[93] J. Liu, P. Musialski, P. Wonka, and J. Ye. Tensor completion for estimating missing values in visual data. *Transactions on Pattern Analysis and Machine Intelligence*, 35(1):208–220, 2013.

[94] J. Liu, Q. Zhang, and C. Tang. Find dense correspondence between high resolution non-rigid 3D human faces. *Communications in Computer and Information Science*, 472:259–275, 2014.

[95] C. T. Loop. Smooth subdivision surfaces based on triangles. Master's thesis, University of Utah, 1987.

[96] J. M. Lounsbery. *Multiresolution Analysis for Surfaces of Arbitrary Topological Type*. PhD thesis, University of Washington, 1995.

[97] Breidt M., Bülthoff H.H., and Curio C. Robust semantic analysis by synthesis of 3D facial motion. In *International Conference on Automatic Face and Gesture Recognition and Workshops*, pages 713–719, 2011.

[98] S. Mallat. *A Wavelet Tour of Signal Processing: The Sparse Way*. Academic Press, 2008.

[99] C. E. McCulloch, B. Paal, and S. P. Ashdown. An optimisation approach to apparel sizing. *Journal of the Operational Research Society*, 49(5):492–499, 1998.

[100] T. Moons, L. Van Gool, and M. Vergauwen. 3D reconstruction from multiple images part 1: Principles. *Foundations and Trends in Computer Graphics and Vision*, 4(4):287–404, 2010.

[101] M. Mori. Bukimi no tani (the uncanny valley). *Energy*, 7(4):33–35, 1970.

[102] I. Mpiperis, S. Malassiotis, and M. G. Strintzis. Bilinear models for 3-D face and facial expression recognition. *Transactions on Information Forensics and Security*, 3:498–511, 2008.

[103] T. Neumann, K. Varanasi, S. Wenger, M. Wacker, M. Magnor, and C. Theobalt. Sparse localized deformation components. *Transactions on Graphics (Proceedings of SIGGRAPH Asia)*, 32(6):179:1–179:10, 2013.

[104] Frank Nielsen. Fast stabbing of boxes in high dimensions. *Theoretical Computer Science*, 246(1-2):53–72, 2000.

[105] A. Nigmetov. Analysis of tensor decomposition for 3D human face modeling. Master's thesis, Saarland University, 2014.

[106] V. Nissen and H. Paul. A modification of threshold accepting and its application to the quadratic assignment problem. *OR Spektrum*, 17(2-3):205–210, 1995.

[107] OpenCV. `http://opencv.org/`.

[108] G. Passalis, P. Perakis, T. Theoharis, and I.A. Kakadiaris. Using facial symmetry to handle pose variations in real-world 3D face recognition. *Transactions on Pattern Analysis and Machine Intelligence*, 33(10):1938–1951, 2011.

[109] A. Patel and W.A.P. Smith. 3D morphable face models revisited. In *Conference on Computer Vision and Pattern Recognition*, pages 1327–1334, 2009.

[110] A. Patel and W.A.P. Smith. Simplification of 3D morphable models. In *International Conference on Computer Vision*, pages 271–278, 2011.

[111] W. Qin, Y. Hu, Y. Sun, and B. Yin. An automatic multi-sample 3D face registration method based on thin plate spline and deformable model. In *International Conference on Multimedia and Expo Workshops*, pages 453–458, 2012.

[112] L. Rabiner and B.-H. Juang. *Fundamentals of Speech Recognition*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.

[113] Intel RealSense. `http://www.intel.com/realsense/`.

[114] A. Salazar, S. Wuhrer, C. Shu, and F. Prieto. Fully automatic expression-invariant face correspondence. *Machine Vision and Applications*, 25(4):859–879, 2014.

[115] G. Sandbach, S. Zafeiriou, M. Pantic, and D. Rueckert. A dynamic approach to the recognition of 3D facial expressions and their temporal models. In *International Conference on Automatic Face and Gesture Recognition and Workshops*, pages 406–413, 2011.

[116] A. Savran, N. Alyuöz, H. Dibeklioglu, O. Celiktutan, B. Gökberk, B. Sankur, and L. Akarun. Bosphorus database for 3D face analysis. In *Biometrics and Identity Management*, pages 47–56, 2008.

[117] K. Scherbaum. *Data Driven Analysis of Faces from Images*. PhD thesis, Saarland University, 2013.

[118] K. Scherbaum, T. Ritschel, M. Hullin, T. Thormählen, V. Blanz, and H.-P. Seidel. Computer-suggested facial makeup. In *Computer Graphics Forum (Proceedings of Eurographics)*, 2011.

[119] P. Schröder and W. Sweldens. Spherical wavelets: Efficiently representing functions on the sphere. In *SIGGRAPH*, pages 161–172, 1995.

[120] F. Shi, H.-T. Wu, X. Tong, and J. Chai. Automatic acquisition of high-fidelity facial performances using monocular videos. *Transactions on Graphics (Proceedings of SIGGRAPH Asia)*, 33(6):222:1–222:13, 2014.

[121] M. Smet and L. Van Gool. Optimal regions for linear model-based 3D face reconstruction. In *Asian Conference on Computer Vision*, pages 276–289, 2010.

[122] M. Styner, I. Oguz, T. Heimann, and G. Gerig. Minimum description length with local geometry. In *International Symposium on Biomedical Imaging*, pages 1283–1286, 2008.

[123] R.W. Sumner and J. Popović. Deformation transfer for triangle meshes. *Transactions on Graphics (Proceedings of SIGGRAPH)*, 23(3):399–405, 2004.

[124] W. Sweldens. The lifting scheme: A custom-design construction of biorthogonal wavelets. *Applied and Computational Harmonic Analysis*, 3(2):186–200, 1996.

[125] G.K.L. Tam, Z.-Q. Cheng, Y.-K. Lai, F.C. Langbein, Y. Liu, D. Marshall, R.R. Martin, X.-F. Sun, and P.L. Rosin. Registration of 3D point clouds and meshes: A survey from rigid to nonrigid. *Transactions on Visualization and Computer Graphics*, 19(7):1199–1217, 2013.

[126] F.B. ter Haar R.C. Veltkamp. 3D face model fitting for recognition. In *European Conference on Computer Vision*, pages 652–664, 2008.

[127] J. Thies, M. Zollhöfer, M. Nießner, L. Valgaerts, M. Stamminger, and C. Theobalt. Real-time expression transfer for facial reenactment. *Transactions on Graphics (Proceedings of SIGGRAPH Asia)*, 34(6):183:1–183:14, 2015.

[128] H.H. Thodberg. Minimum description length shape and appearance models. *Information Processing in Medical Imaging*, 18:51–62, 2003.

[129] L. N. Trefethen and D. Bau. *Numerical Linear Algebra*. SIAM, 1997.

[130] L.R. Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311, 1966.

[131] O. van Kaick, H. Zhang, G. Hamarneh, and D. Cohen-Or. A survey on shape correspondence. *Computer Graphics Forum*, 30(6):1681–1707, 2011.

[132] D. Vlasic, M. Brand, H. Pfister, and J. Popović. Face transfer with multilinear models. *Transactions on Graphics (Proceedings of SIGGRAPH)*, 24(3):426–433, 2005.

[133] T. Weise, S. Bouaziz, H. Li, and M. Pauly. Realtime performance-based facial animation. *Transactions on Graphics (Proceedings of SIGGRAPH)*, 30(4):77:1–77:10, 2011.

[134] S. Wuhrer, C. Shu, and P. Bose. Automatically creating design models from 3D anthropometry data. *Journal of Computing and Information Science in Engineering*, 12(4), 2012.

[135] F. Yang, L. Bourdev, J. Wang, E. Shechtman, and D. Metaxas. Facial expression editing in video using a temporally-smooth factorization. In *Conference on Computer Vision and Pattern Recognition*, pages 861–868, 2012.

[136] F. Yang, J. Wang, E. Shechtman, L. Bourdev, and D. Metaxas. Expression flow for 3D-aware face component transfer. *Transactions on Graphics (Proceedings of SIGGRAPH)*, 30(4):60:1–10, 2011.

[137] J. S. Yedidia, W. T. Freeman, and Y. Weiss. *Exploring Artificial Intelligence in the New Millennium*, chapter Understanding Belief Propagation and its Generalizations, pages 239–269. Morgan Kaufmann Publishers Inc., 2003.

[138] L. Yin, X. Chen, Y. Sun, T. Worm, and M. Reale. A high-resolution 3D dynamic facial expression database. In *International Conference on Automatic Face and Gesture Recognition*, pages 1–6, 2008.

[139] L. Yin, X. Wei, Y. Sun, J. Wang, and M. J. Rosato. A 3D facial expression database for facial behavior research. In *International Conference on Automatic Face and Gesture Recognition*, pages 211–216, 2006.

[140] F. Zhou and F. De La Torre. Generalized time warping for multi-modal alignment of human motion. In *Conference on Computer Vision and Pattern Recognition*, pages 1282–1289, 2012.