COPPE
Instituto Alberto Luiz Coimbra de
Pós-Graduação e Pesquisa de Engenharia
UFRJ

A METHOD FOR TRIDIMENSIONAL FACIAL MODEL COMPOSITION

Diego Mazala Carvalho Rodrigues

Tese de Doutorado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Doutor em Engenharia de Sistemas e Computação.

Orientador: Claudio Esperança

Rio de Janeiro
Maio de 2022

UM MÉTODO PARA COMPOSIÇÃO DE MODELOS FACIAIS TRIDIMENSIONAIS

Diego Mazala Carvalho Rodrigues

TESE SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Orientador: Claudio Esperança

Aprovada por: Prof. Claudio Esperança
Prof. Ricardo Cordeiro Farias
Prof. Ricardo Guerra Marroquim
Prof. Waldemar Celes Filho
Prof. Anselmo Antune Montenegro

RIO DE JANEIRO, RJ – BRASIL
MAIO DE 2022

Rodrigues, Diego Mazala Carvalho

A method for tridimensional facial model composition/Diego Mazala Carvalho Rodrigues. – Rio de Janeiro: UFRJ/COPPE, 2022.

XI, 50 p.: il.; $29,7$cm.

Orientador: Claudio Esperança

Tese (doutorado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2022.

Referências Bibliográficas: p. 43 – 50.

1. Blending detail. 2. Facial Mesh. 3. Blendshapes. 4. Detail Transfer. I. Esperança, Claudio. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

Para a minha esposa e filhas que
tiveram que pagar o preço da
minha ausência

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

UM MÉTODO PARA COMPOSIÇÃO DE MODELOS FACIAIS TRIDIMENSIONAIS

Diego Mazala Carvalho Rodrigues

Maio/2022

Orientador: Claudio Esperança

Programa: Engenharia de Sistemas e Computação

A criação de modelos faciais tridimensionais é uma tarefa desafiadora. Não somente pelo esforço e habilidades artísticas necessárias mas também por causa da extrema sensibilidade dos humanos de perceber anormalidades em rostos. Neste trabalho, nós apresentamos um método para combinação de faces humanas para composição de um novo modelo. Nossa proposta aplica suavização laplaciana para separar camadas de detalhes. Propomos ainda um editor para compor estes detalhes sobre faces de uma forma simples e prática. Nossa principal contribuição é uma abordagem intuitiva e facilmente inserida na produção artística de modelos faciais.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

A METHOD FOR TRIDIMENSIONAL FACIAL MODEL COMPOSITION

Diego Mazala Carvalho Rodrigues

May/2022

Advisor: Claudio Esperança

Department: Systems Engineering and Computer Science

Designing realistic tridimensional facial models is a challenging task. Not only by the effort and artistic abilities required but also because humans are extremely sensitive to facial abnormalities. In this work, we present a method to blend human faces in order to compose a new one. Our method uses laplacian smooth to segregate the layers of details. Besides, we propose an editor to compose those details over faces in a simple and practical way. Our main contribution is an intuitive approach easily absorbed by artistic pipelines when designing facial human models.

Contents

List of Figures **ix**

1 Introduction **1**

1.1 Motivation 2

1.2 Objectives 3

2 Acquisition Pipeline **5**

2.1 Photogrammetry Rig 5

2.2 Photogrammetry Pipeline 6

2.3 Warping a Template Mesh 6

3 Related Works **10**

3.1 Blend Shapes 10

3.2 Facial Scanning 11

3.3 Mesh Smoothing 13

3.4 Facial Detail Transfer 15

4 Synthesizing a Face Using Multiscale Face Models **18**

4.1 Discussion 20

5 Our Method **22**

5.1 Laplacian Smoothing 22

5.2 Blending details between faces 24

5.2.1 Approximating smoothness levels 25

5.3 Multiple Regions 26

6 The Software **31**

7 Results and Comparisons **33**

7.1 Results 33

7.2 Comparisons 36

8 Conclusion **41**

Bibliography **43**

List of Figures

1.1 Result of blending two faces. The model on the left provides the details (source), the one on the right the base anatomy (target), and the result is shown in the middle. The textures are for illustration purposes only, our method only deals with geometric blending. 4

2.1 Photogrammetry Rig with 20 Canon EOS T5i cameras. The subject was a professional volleyball player from the Brazilian team. 6
2.2 Photogrammetry Pipeline 7
2.3 R3DS Wrap interface. Two models being aligned as well as a few correspondence points for the rigid alignment process. On the right corner the nodes used for loading meshes and textures aside from nodes to apply rigid alignment and set correspondence points. 8
2.4 A template mesh being warped to another head model. 9

3.1 Example of five blendshape targets for a human head showing facial expressions. 11
3.2 The schematic diagram of the feature prior constraint. In each viewpoint image, the projection of landmarks (yellow pixels in (a–c)) should have the same coordinate as the extracted feature pixels (green pixels). Illustration reproduced from [1] 12
3.3 Feature detection on 3d face models. Illustration reproduced from [2] 13
3.4 (a) Multi-view images shot under rapidly varying flash directions. (b) Refined geometry (c) Diffuse/specular maps (d) Rendering. Illustration reproduced from [3] 14
3.5 Composition of two new models from various source models. Illustration reproduced from [4] 16
3.6 Example of an input skull model and the output facial reconstruction obtained by [5]. Images reproduced from [5]. 16

3.7 Framework for learning physically based face models present by Li et al. [6]. (a) source face models; (b) combination of anatomical and physically based attributes; (c) output faces; (d) high resolution output. Illustration reproduced from [6]. 17

4.1 Surface approximation by multilevel B-spline: 4.1a is the source face model, (b-g) are examples of approximation surfaces $S_l(u, v)$ at different levels $l = 4, 5, 6, 7, 8, 9$. The surface (d) is shown with normal vector fields. Figure reproduced from [7]. 20

4.2 Illustration of the process to create an MFM. The leftmost face is S_b, that combined with vector field D_0 produces S_d (second face). Then, the scalar CDMs $D_{l=\{1,2,3\}}$ area use to displace S_d along its normal direction. The final face model $\mathcal{M}$ is shown on the far right. Figure reproduced from [7]. 20

4.3 Three results of blending multiscale human faces. Each new face (larger images) is created from two original models (smaller images). Besides each original face its CDM is illustrated, and the red squares mark which components were used, i.e., have weights larger than zero. For the leftmost example, with original faces $\mathcal{M}^B$ and $\mathcal{M}^H$, the base surface S_b^B of face B is chosen and the blending weights are: $\vec{\alpha}_l^B = (0.5, 0, 0, 0)$ and $\vec{\alpha}_l^H = (0.5, 1, 1, 1)$. The weights for the other examples were not specified in the original paper. Figure reproduced from [7]. 21

5.1 Example of the Laplacian smoothing operator applied to a face mesh. Four levels of smoothing for a mesh with approximately 1.2 million vertices. From the left to right, the original model is followed by smoothed meshes using 500, 2500, 5000 and 10000 iterations, respectively. 23

5.2 Blending the source and target faces from Figure 1.1 with the following number of smoothing iterations L from the left to right: 10000, 14000, 18000 and 20000. 24

5.3 Detail extrapolation by manipulating the weights from the example shown in Figure 1.1. From the left to right scaling all weights by 0.75, 1.0, 1.2 and 1.4. Notice that not only details are intensified but the global shape of the face is also deformed for high values. 25

5.4 The image shows two sample vertices extracted from nose and lips region. These are the vertices we used to explain bellow our error measurements. 27

5.5 Comparison between different interpolation methods and their distance measurement to the original vertices path in a test case with 20000 smoothing iterations. 28

5.6 The distance error measured between different interpolation methods for the two sampled vertices shown in Figure 5.4 29

5.7 An example of using region masks manually defined. On the left, one of the original textures and the painted regions in texture space. Note the blending region between the top (red) and middle (blue), and middle (blue) and bottom (green) masks to achieve a smooth transition. On the right, the resulting regions applied on one of the heads. 30

6.1 The main screen of our interface. (1) resultant mesh A; (2) one of the thumbnails representing a source face; (3) dashed blue circle indicates selected target face B; (4) a controller for a specific region defining position; (5) sliders to control the smoothness level for each region; (6) available textures that can be applied to the models; (7) buttons to save resulting model or image. 32

7.1 Example of using multiple faces with different weights per region. Here, each source face has full weight for only one specific region to better illustrate the effect. 34

7.2 Some results of our approach applied on meshes with 1.2 million vertices. For each row, column (a) shows the source face from the details are extract, column (c) shows the target face and column (b) the blending result. 35

7.3 Comparison between our method and Yoon [7] proposal. The colors show the distance between the vertices of the meshes. The left column are results from our proposal; the right column from Yoon's method; and the center column is the difference between both. 37

7.4 Comparison between our method and Yoon [7] proposal. The colors show the distance between the vertices of the meshes. The left column are results from our proposal; the right column from Yoon's method; and the center column is the difference between both. 38

7.5 Comparison between our method and Yoon [7] proposal. The colors show the distance between the vertices of the meshes. The left column are results from our proposal; the right column from Yoon's method; and the center column is the difference between both. 39

7.6 The circles and arrows in red show the boundaries problems we found in our implementation of Yoon et al [7]. 40

Chapter 1

Introduction

Throughout generations, humans have been developing as a society through the interaction between their peers. This interaction exposed us to countless numbers of faces and their nuances. This experience shaped our brain throughout the visual system, to become a highly trained instrument in facial recognition and detecting its imperfections. This advanced skill of our brains also renders it challenging to create believable digital faces.

In fact, the creation of digital models of human faces is a task that has been extensively researched and improved over the years. Numerous artists are currently producing their models using a variety of software. They create not only the mesh, but also produce animations and develop appropriate tone and color for the skin, eyes, hair, and all other elements that compose a human face. If one of these elements is out of tune, it is quickly perceived by a human observer, giving the impression of an artificial face. Some of the achieved results are, nevertheless, surprising and, in some case, it is impossible to determine if it is a computer generated face. Yet, this high fidelity and realism comes with a price, as it still takes a long time to develop a digital face that is able to reach a level of credibility high enough to fool our brain.

The film and gaming industries [8] have played a key role in advancing research in this area. Mainly because the continuous demand of producing computer graphics generated humans and regarding the small scale facial details, such as wrinkles, pores, and hairs. Albeit the advances in scanning technologies and how much they have contributed to capturing and reproducing facial details, many characters in animated movies are imaginary, so they cannot be scanned. Besides, several details and imperfections are designed and sculpted by artists. Thus, considering the need to create original faces, it is evident the lack of a tool capable of providing a modeling pipeline that is simple, understandable, practical, fast, and inexpensive. Our goal is to reduce this gap.

One of the points that we consider essential in our proposal is the form of parameterization, as we want it to be intuitive, practical, and simple. *Blendshapes*

[9], [10] and [11] are a good example of such case, as through a few parameters one face, or facial expression, may be transformed into a new one. We believe that similarities with such well-known modeling and animation techniques will favor the understanding and usability of our proposal by artists. Our goal, nevertheless, is different from Blendshapes, as will be further detailed below.

1.1 Motivation

In the company where I was employed, TV Globo, there was continuous demand for computed generated actors. TV Globo is the largest commercial TV network in Latin America, the second-largest commercial TV network of the world, and the biggest soap operas producer. Several of the entertainment productions require computed generated characters, humans and non-humans. My team and I have developed a workflow to help automatizing the generation of tridimensional facial models, and, as a result of this development, we built a database with digital faces from human actors captured using photogrammetry [8, 12, 13]. These models share the same topological structure, as well as the same texture coordinate space. Our system can incorporate new faces as they become available.

In one occasion we had a further challenging requirement: to build a software tool to facilitate the process of generating virtual actors for a variety of scenes. Considering the different aspects related to a scene, one of the most significant for us was the level of the details required for the actors. In this case, we were not targeting the closest viewer plan, where it was played by real actors, but a second or third plan, where more detailed models were necessary and they should have characteristics that could identify them as natives of the geographical region where the story took place. Since there were no available solutions to generate these specific models, and producing multiple 3D face models from scratch is a highly laborious and time-consuming task, the idea of a method to quickly generate new 3D faces from a few available models came to light. Even more, the aim was to capture a few models from real subjects, in order to then generate new original virtual characters with a high level of detail.

In spite of the short period of time, we could implement a pipeline that provides substantial help on designing new faces, reducing the time and effort required for this task. It was not completely automatized but saved considerable artistic time and cost. Nevertheless, we understood that the goal was only partially accomplished since our objective was further, having a well defined pipeline for photogrammetry and asset production was an important requirement but not enough in our case. Even by minimizing time cost for each photogrammetry section, it is still impractical if the goal is to generate crowds. In fact, there is still a series of remaining issues with the current pipeline, as detailed below:

- **Required time**: even if the photo acquisition phase only takes a few seconds, the total time necessary to generate a new face is around 5 days, at least.
- **Set size**: since we want to have dozens or hundreds of faces and all of them with specific characteristics that define races and ethnicity, digitizing everyone is not an option. Not only due to the difficulty of finding actors with the characteristics we are looking for, but also because of the long time to process each one.
- **Image rights**: there is a monetary cost for image rights to have an actor scanned. This cost might be prohibitive for a huge number of faces.
- **Availability**: apart from the cost, arranging the agenda of the actors with studio availability would not be a simple task.

Given the issues cited above, and the lack of available commercial solutions, this research idea was proposed.

1.2 Objectives

In this proposal we present an approach to generate new three-dimensional face models from preexisting ones. Our target is to provide a simple, practical, and intuitive system to blend details, in order to allow artists to create new models in a well controlled way. Since we specifically target background or secondary characters, and not heroes, a system comprised of only a few sliders to weight the blended features is ideal, as the artist usually does not have the time to worry about all the small details for these characters. For main characters, on the other hand, control of all aspects of the face would be preferable rather than a semi-automated system that can provide a believable face in a short amount of time.

Briefly, the main idea of our approach is to transfer the details from one face to another. We achieve this by smoothing the faces in order to separate the coarse and fine details, and then by transferring the details of one face to the coarse mesh of another in a controlled way. Figure 1.1 illustrates this concept.

This thesis is organized in the following way: Chapter 2 we show how we developed our acquisition pipeline; Chapter 3 describes related works; in Chapter 4 we discuss in more details a specific work that is most related to our method; in Chapter 5 we explain our proposed method; in Chapter 7 we debate about some results and make some comparisons; and, finally, in Chapter 8 we conclude and discuss future steps.

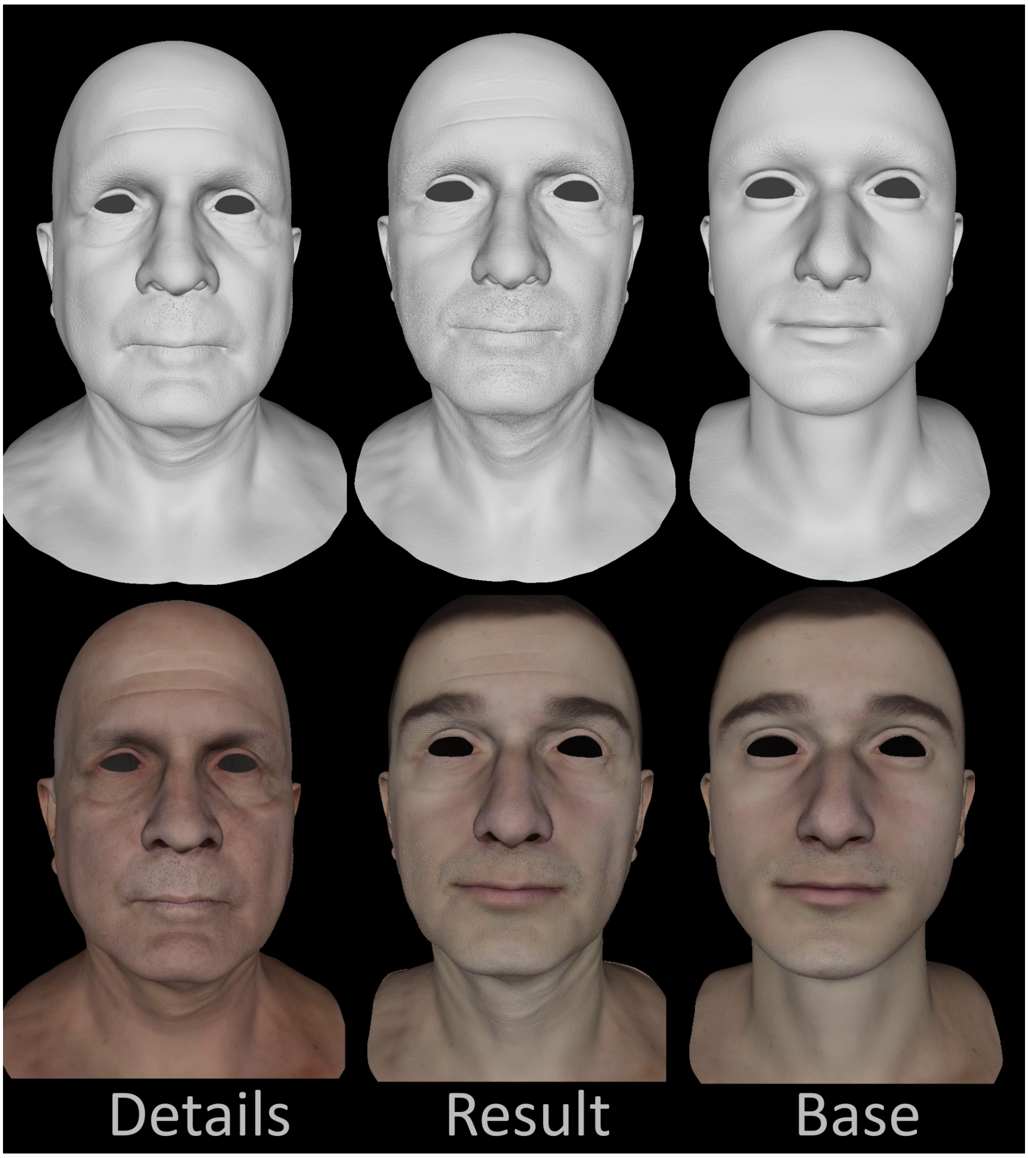


Figure 1.1: Result of blending two faces. The model on the left provides the details (source), the one on the right the base anatomy (target), and the result is shown in the middle. The textures are for illustration purposes only, our method only deals with geometric blending.

Chapter 2

Acquisition Pipeline

In this chapter we will discuss the pipeline that was developed in order to scan actors using photogrammetry and generate the final tridimensional digital models. Even though the pipeline is not part of the contribution of this work, it helps in contextualizing the proposal and how our source database is created, apart from clarifying the current challenges in digitizing human faces.

2.1 Photogrammetry Rig

Most of the state-of-the-art facial performance capture pipelines are based on a multi-view stereo setup. This arrangement captures not only fine geometric details as well high quality texture.

Our rig was composed of 20 Canon EOS T5i cameras ready to handle the photogrammetry sections. All cameras were connected to a computer through USB ports. The shots were triggered from the software interface to ensure that all cameras shoot at the same time. The time difference between the shots must be as small as possible to avoid subtle movements from the subject that could interfere in the photo alignment. The registered time difference was not larger than 20 milliseconds.

The frontal tripods were mounted with three cameras each, while the back tripods had two cameras. We used this configuration in order to have more pixel information over the face rather than the back of the head, as can be seen in Figure 2.1. In fact, the photos of the head and hair were used only as reference, since hair was sculpted manually by an artist in a post process. Due to the reflective nature of the hair, it would require more advanced techniques to achieve satisfactory results using photogrammetry of regions like scalp and beard.

The total time spent in the acquisition phase, from the moment the subject sits down to be photographed and the final verification of the downloaded photos into the computer, was no longer than 45 seconds per facial expression.

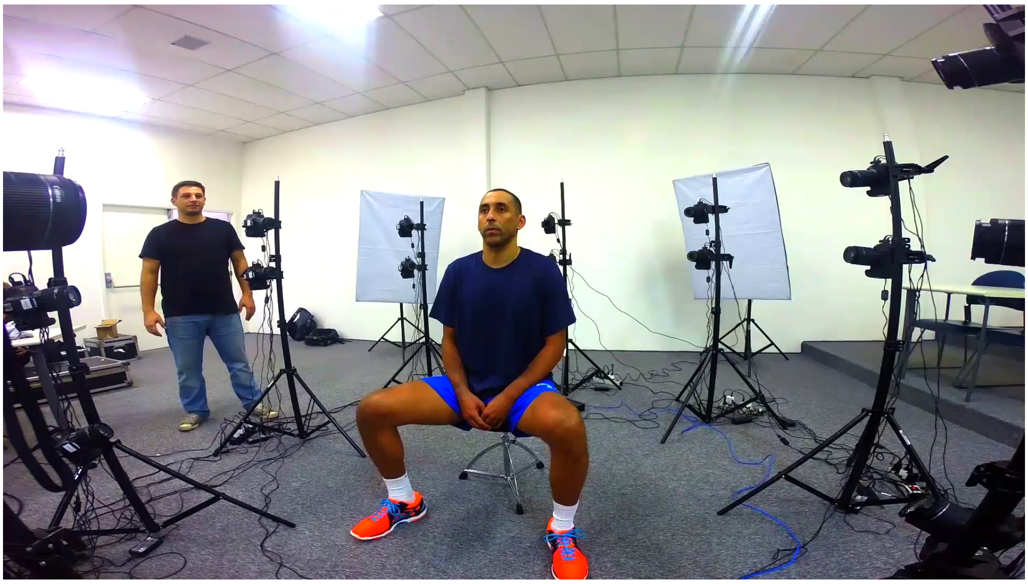


Figure 2.1: Photogrammetry Rig with 20 Canon EOS T5i cameras. The subject was a professional volleyball player from the Brazilian team.

2.2 Photogrammetry Pipeline

After capturing the photos, the alignment process begins. For this stage we used the software Agisoft Photoscan [14], considered one of the best photogrammetry tools in the market. In our case, we used Photoscan to align the photos and build the first raw version of the three-dimensional facial model. If the shots were acquired carefully in ideal conditions, there were hardly any problems with alignment. Examples of situations that may interfere negatively in the matching of images is a heterogeneous setup of lights, or if the actor is wearing makeup resulting in a more reflective skin.

Once the images were aligned, a three dimensional digital model of the actor was generated. This raw high resolution geometry was sent to Zbrush [15] where an artist would work manually to cleanup the mesh. More specifically, he would remove holes or noise that were common for the photogrammetry process. The main work in this phase was related to regions with hair (head, bear, eyebrows) or high specularity (eyes).

Figure 2.2 illustrates the whole pipeline, from the acquisition phase to the final asset ready to production.

2.3 Warping a Template Mesh

Notwithstanding, the final geometry from the acquisition process and post process was usually not suitable for artists to continue working with. Indeed, the final geometry could be completely arbitrary. It was desirable that the geometry from

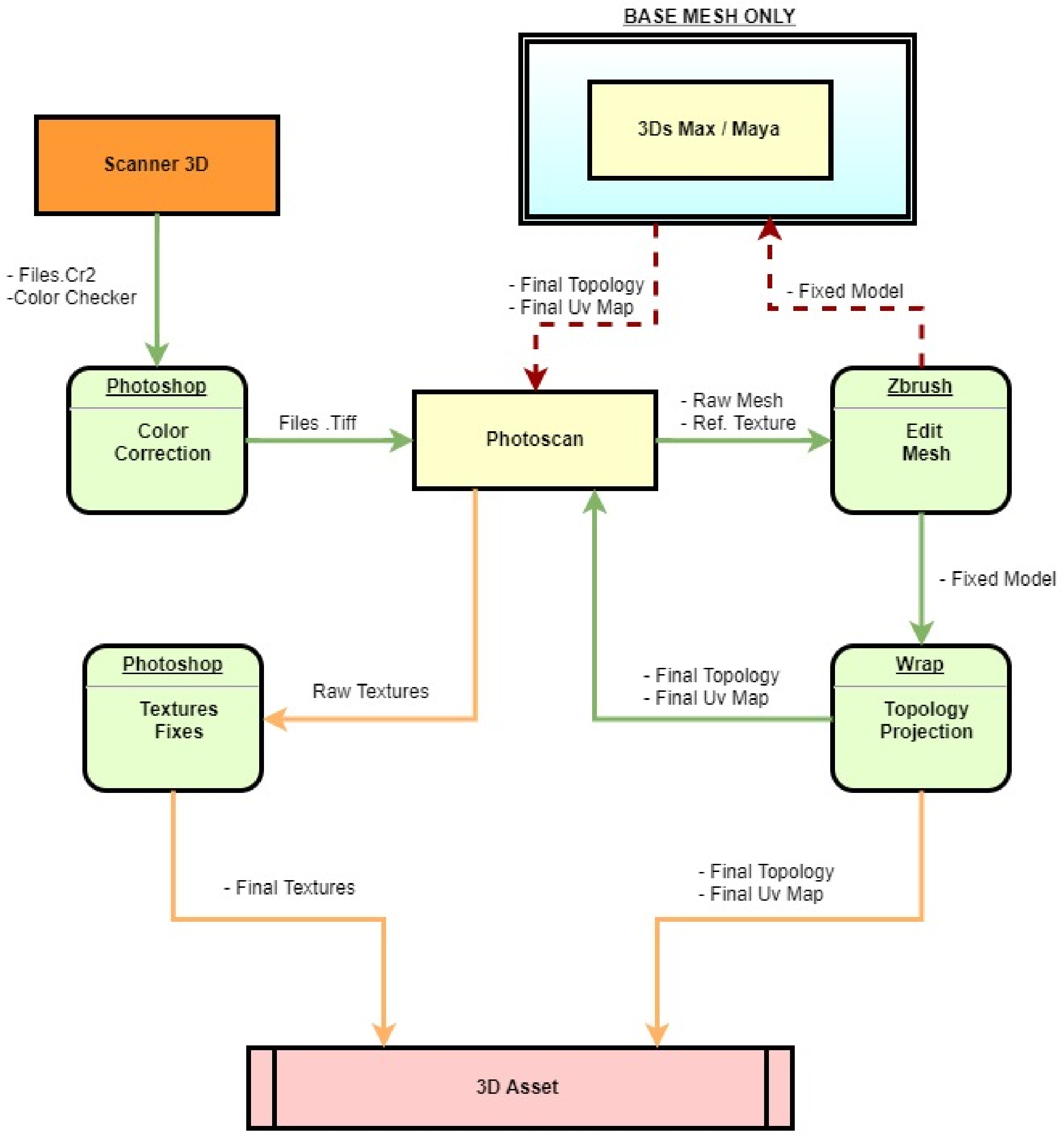


Figure 2.2: Photogrammetry Pipeline

different face models share the same topology as it would benefit the post processing stages such as texturing or animation, when the artist could reuse the same effort between multiple models. Therefore, the models must be converted from the arbitrary geometry to a common template. For the specific case of facial models, this template would present some attributes that were very important for later stages, like number of vertices and edge-loops.

The goal was then to warp different subjects in arbitrary poses to this common a template face. The key challenge was to handle the large variations of facial appearances and geometries, as well as the complexity of facial expression and large deformations.

We used the R3DS Wrap 3.3 [16], a commercial software product with functionality to non-rigidly register surfaces, to warp the template mesh into the new model. The method uses landmark points to compute an optimal alignment between two non-rigid surfaces.

In order to deform a model, a set of landmark correspondences points between the model and the template are selected, a first rigid transformation is applied and then a non-rigid alignment is computed between the face models. The method uses N-ICP [17] in a coarse-to-fine manner to establish an approximate non-rigid alignment, which is subsequently refined in further steps. Figure 2.3 shows a screenshot of the R3DS Wrap interface and Figure 2.4 shows an example of a template head mesh being warped to fit the target head mesh.

It would be also possible to use R3DS Wrap for others scan related tasks, such as fixing non-manifold topology, remove spikes, remove small components or decimate the raw mesh. Nevertheless, in our pipeline, these tasks were done by digital artists in Zbrush [15], since they have more detailed control over the final mesh.

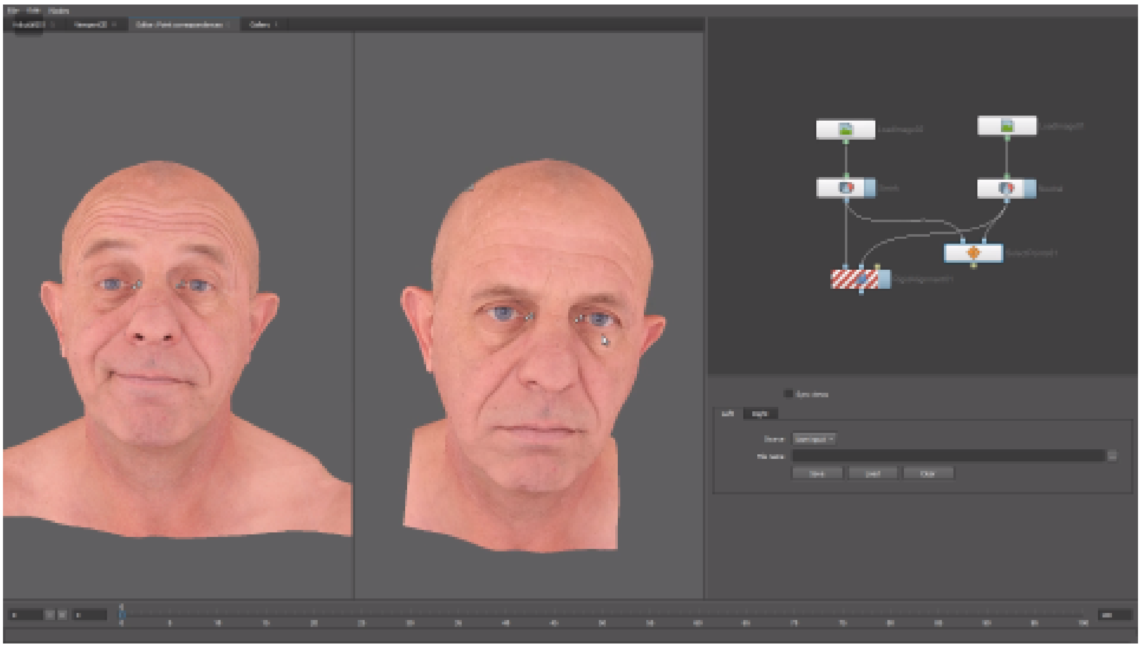


Figure 2.3: R3DS Wrap interface. Two models being aligned as well as a few correspondence points for the rigid alignment process. On the right corner the nodes used for loading meshes and textures aside from nodes to apply rigid alignment and set correspondence points.

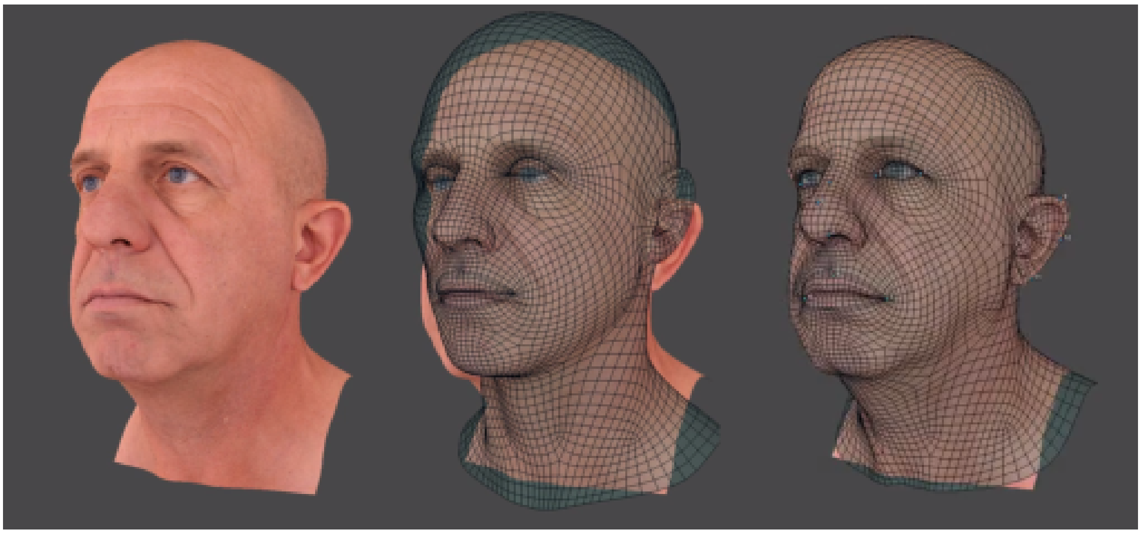


Figure 2.4: A template mesh being warped to another head model.

Chapter 3

Related Works

In this chapter, we will describe some of the techniques and research related directly or indirectly to our work. Note that there is a specific work, Yoon et al. [7], which is most related to ours, that will be described in more details in Chapter 4.

3.1 Blend Shapes

Blendshapes is one of the most used approaches for facial animation and deformation [18]. Although there are several others alternatives, such as methods based on principal component analysis [19, 20], physically-based-models [21, 22], motion capture driven meshes [23, 24], and interpolation of an abstract pose [25, 26], blendshapes are still popular due to its simplicity and ease of understanding. Even in cases when more sophisticated techniques are applied, blendshapes are frequently used as a base layer over which non-linear or physically based deformations are performed. See Figure 3.1 for an illustration of a head with five facial expressions blendshapes.

According to Lewis et al. [18], "blendshapes are linear facial models in which the individual basis vectors represent individual facial expressions". As a consequence, the basis is not orthogonal in general [19]. From an artist's point of view, the interpretability of the blendshape basis is a defining feature.

A face model is built from a linear combination of a number of facial expressions or deformations, which are called blendshape targets. A relevant research in facials poses has been done by Ekman and Rosenberg [27]. They defined a set of targets that, in some combination, could encompass all possible human facial expressions, approximating the linearized effect of individual facial muscles. With little computation, it is possible to achieve a wide range of facial expressions just by varying the weights of the linear combination. Blendshapes has several advantages when compared to other representations:

- the weights have intuitive semantic parameterization since they represent the influence of each facial expression;
- it is possible to achieve the target shape just by applying the maximum weight of its influence parameter;
- it is easy to avoid undesired deformations.

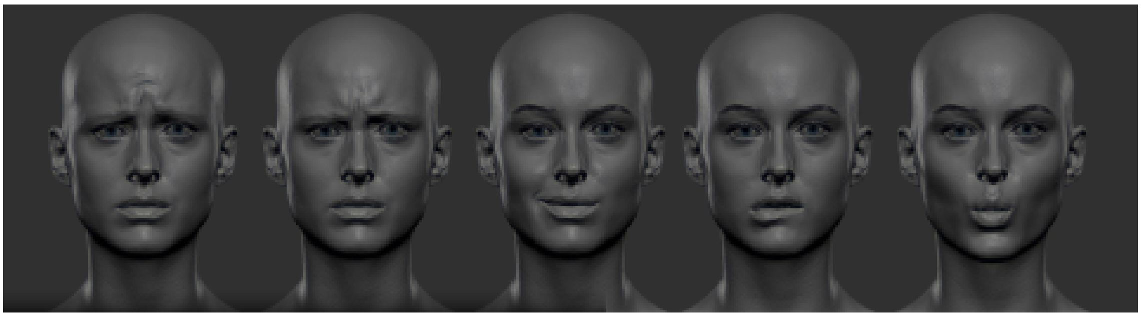


Figure 3.1: Example of five blendshape targets for a human head showing facial expressions.

Even with the conceptually simple framework of blendshapes, it still demands a laborious intensive effort. For this reason, photogrammetry has become an useful tool in facial modeling, as will be seen next.

3.2 Facial Scanning

Tian et al. [1] present a method for face reconstruction using images from three viewpoints. They extract 3D shapes and poses by incorporating the feature prior constraint and the texture constraint, which are learned from multi-view images. While the feature prior constraint is used to estimate accurate 3D facial contours, the texture constraint extracts a high-precision 3D facial shape. They claim that their method is more robust than traditional methods particularly when there are limited number of feature points. See figure 3.2.

Fyffe et al. [28] describe a video based facial capture which deforms a common template model to match multi-view input images of the subject. For each frame, they compute the cross-view, cross-subject, and cross-pose consistencies using a combination of 2D landmark detection, optical flow, and surface and volumetric Laplacian regularization. Since they process each frame independently, their method can be easily parallelizable. In addition, they use a PCA-based dimension reduction and denoising scheme to extract the head pose.

Zhu et al. [2] propose a robust method using multi-feature framework which included SIFT features [29], pixel intensity, and contours 3.3. Because the SIFT

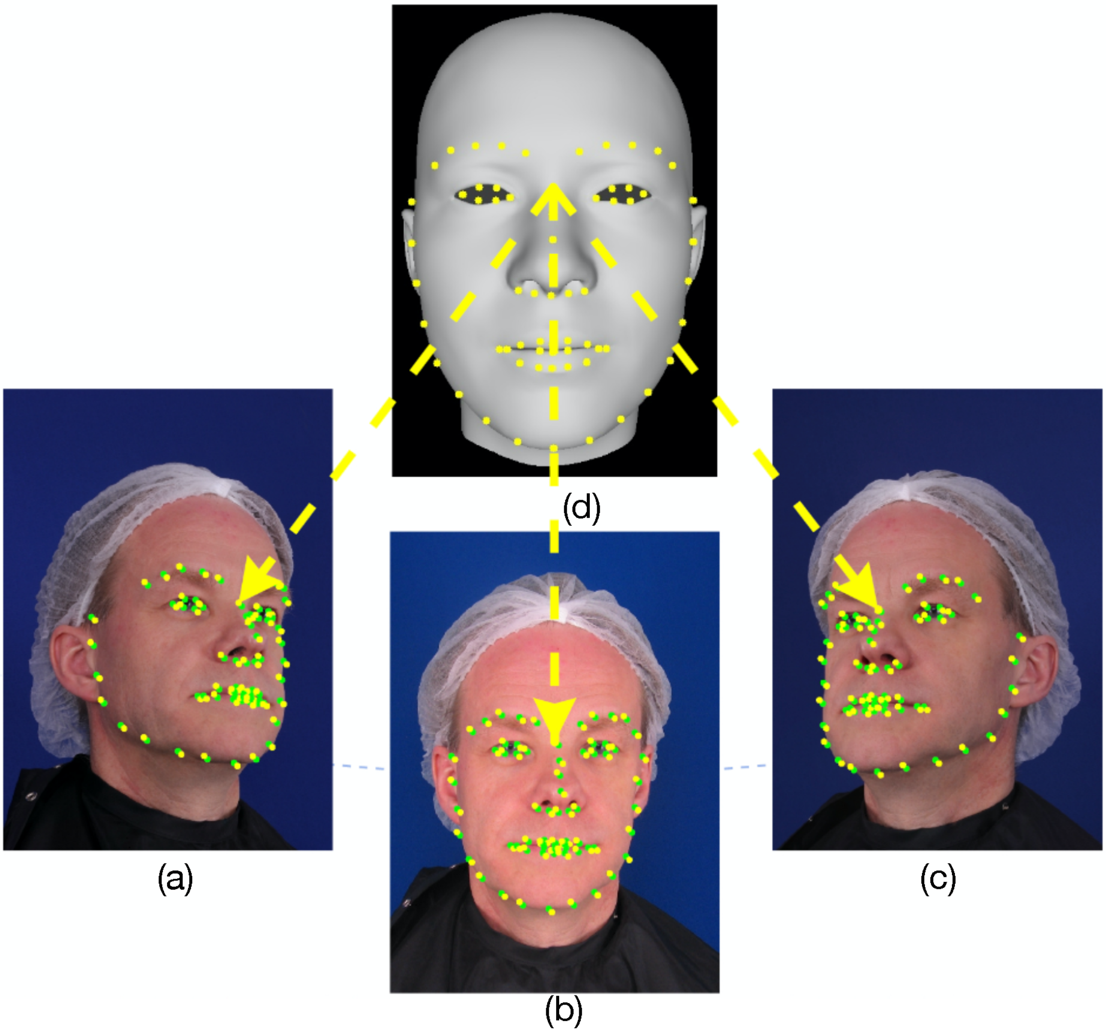


Figure 3.2: The schematic diagram of the feature prior constraint. In each viewpoint image, the projection of landmarks (yellow pixels in (a–c)) should have the same coordinate as the extracted feature pixels (green pixels). Illustration reproduced from [1]

feature is invariant to uniform scaling, orientation and partially invariant to affine distortion and illumination changes, their method can locate the facial components successfully and reconstruct 3D shapes. In an optimized approach, Lin et al. [30] propose a method where they significantly reduce the amount of required facial feature points using self-adaptative morphable models. Dai et al. [31] propose a coarse-to-fine multi-view 3D face reconstruction method by taking advantage of the complementarity between facial feature points and occluding contours. Multi-view face images with visual angle differences are employed to calculate the 3D coordinates of facial feature points to generate 3D face models.

Beeler et al. [32] and Klaudiny et al. [33] use a similar approach, however, they rely on optical flow to track a face model, for which computation is only possible sequentially. They have to break up the performance into short clips and use anchor

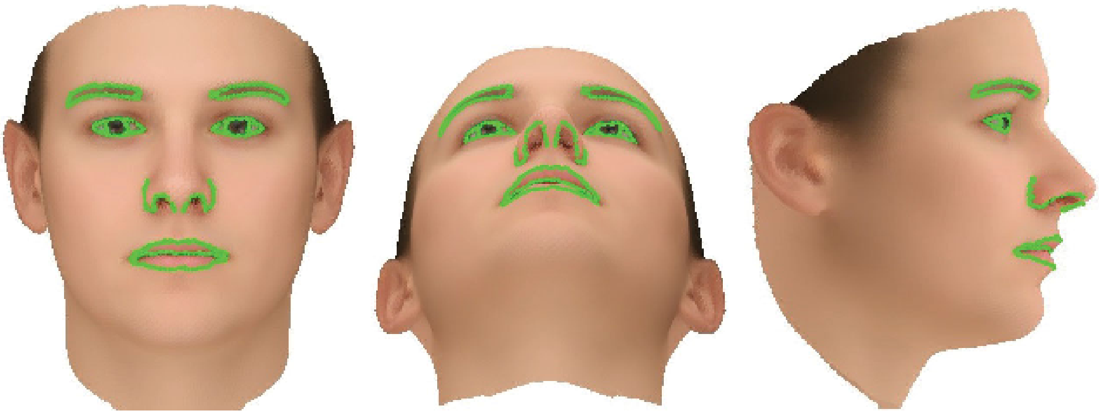


Figure 3.3: Feature detection on 3d face models. Illustration reproduced from [2]

frames or key frames with a common appearance to increase performance. The main drawback is the requirement to repeatedly return to a common expression throughout the performance. In a previous work, Beeler et al. [24] present a passive stereo vision system that computes the 3D geometry of the face with reliability and accuracy on a par with a laser scanner or a structured light system. This is a good example of a low-cost system, they use four cameras with results achieving a submillimeter accuracy. Their main contributions are a modification of the standard stereo refinement methods to capture pore-scale geometry, and a calibration method suited to facial capture systems.

On the topic of facial scanning, Paul Debevec's outstanding work has had a significant impact on the area. In collaboration with Graham Fyffe and Paul Graham [3, 28], they presented a facial capture technique that combines the benefits of single-shot techniques (simple setup of low-cost cameras where each camera takes precisely one photo) and multi-shot techniques (specular reflectance information and high-resolution photometric normals). The main idea is to fire the cameras at nearly the same instant, but with a slight offset, in order to capture under several different lighting conditions before the subject blinks, rather than a single lighting condition. They use a 24-camera entry-level DSLR photogrammetry setup with six ring flashes to light the face. This setup is similar to the one we adopted, except that in our case we capture all images simultaneously and do not acquire reflectance data. See figure 3.4 for an illustration of this work.

3.3 Mesh Smoothing

We call mesh smoothing the process of changing vertex positions in a mesh in order to improve the mesh following some given criterion. It may be useful

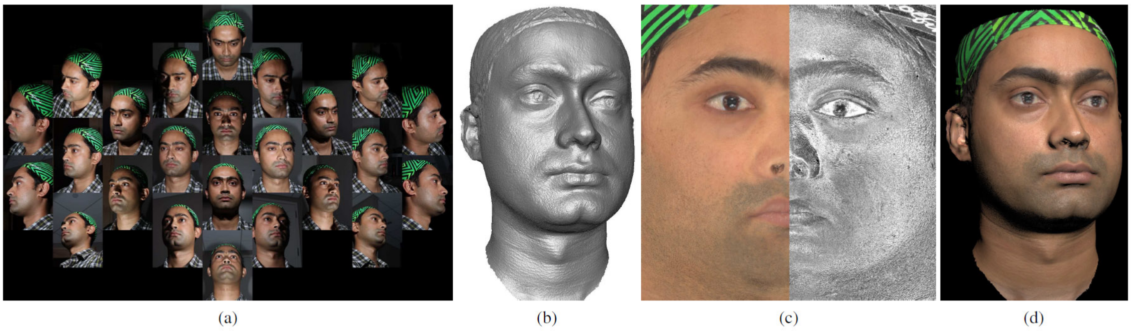


Figure 3.4: (a) Multi-view images shot under rapidly varying flash directions. (b) Refined geometry (c) Diffuse/specular maps (d) Rendering. Illustration reproduced from [3]

to improve mesh quality [34], remove noise [35], or to change its topology [36]. Mesh smoothing methods can be classified as optimization-based [37, 38], geometry-based [39, 40], physics-based [41], and some combination thereof [42, 43].

Mesh improvement methods used in common applications generally require topology modifications of the mesh by adding or removing vertices, or even changing the connectivity of the vertices given by the edges. Sometimes, a preprocessing step may be necessary to avoid those procedures. In our case, we require methods that do not change the mesh topology to ensure that all meshes maintain the vertex correspondences even after the applying a smooth operator.

Due to its simplicity and speed, Laplacian smoothing can be considered one of the most popular smoothing methods [34]. It can be derived from a finite difference approximation of the Laplace operator [44]. Vartziotis et al. [40] discuss the efficiency and effectiveness of Laplacian smoothing and introduce the class of geometric element transformation method. It turns out that Laplacian smoothing on surface meshes maximizes a concave quality function, which is intimately related to the mean ratio quality measure. The Laplacian operator also conserves mesh topology.

Others common approaches for mesh smoothing use global operators, such as radial basis function [45], B-splines [7] or wavelets [46]. These are mathematical functions able to interpolate, on a distance basis, scalar information known only at discrete points (source points) [47]. The quality and the behavior of the interpolation depend both from the function and from the kind of chosen basis function [48]. In RBF interpolation, for instance, given data samples X_i and respective function values defined at the samples, an approximant is constructed in such a way that it interpolates the function f. The approximant is formed by a finite linear combination of translations of radially symmetric functions $\phi(\|.\|)$, where $\|.\|$ is the Euclidean norm in $\mathbb{R}^m$. According to Buhmann [49]: "radial symmetry means that the value of the function only depends on the Euclidean distance of the argument from the origin". Since the translations of ϕ are defined by the samples, the interpolation function s is defined at an arbitraty point x by

$$s(x) = \sum_{i=1}^{N} \lambda_i \phi(\|X_i - x\|),$$

where λ_i are real-valued coefficients.

While RBF is a global operator, for our proposal, the local behaviour of laplacian smoothing tends to preserve in a better way the cavities and curvature of human face parts. With Laplacian operator we were able to separate the details from the main anatomy of the face keeping a rough design of the facial parts, which may be difficult with RBF smoothing.

3.4 Facial Detail Transfer

Transferring details from a source to a target shape is a well studied problem for general digital 3D models. The well known work of Sorkine et al. [50] exemplifies how details can be extracted from a single model via local vertex displacements and coordinates. Sumner et al. [51] describe a way to encode details and relate them to a reference pose. Although these methods were not particularly design for facial detail, they can be customized for this purpose.

Shin et al. [52] propose a method to extract and transfer expression wrinkles from a high resolution example face model to a target model for enhancing the realism of facial animations. Using multiscale detail maps, they isolate the high frequency details that define the expression wrinkles. The detail maps actually contain the surface normal perturbations that will design the expression wrinkles.

Trying to provide an intuitive way to blend detail between faces, Ma et al [4] proposed a facial composite editor. It is an interactive editing system that, starting from a small number of given face models, allows digital modelers to create new blendshape face models for primary or background characters. To avoid the limitation of generating linear combinations of the given models, they propose an approach similar to forensic software, in which face features (eyes, nose, mouth, etc.) from different individuals are assembled to create a composite likeness. To handle differences in scale and position, they adopt a gradient-domain assembly and blending of the individual features, and produce the final face by integrating the resulting Poisson equation. Their method requires vertex-wise correspondence across each face and its Blendshapes. See figure 3.5.

Romeiro et al. [5] propose a way to reconstruct faces from skulls using mesh template deformation and transfer details over Hermite radial basis function (HRBF). Their goal is to provide a semi-automatic facial reconstruction with all the soft tis-

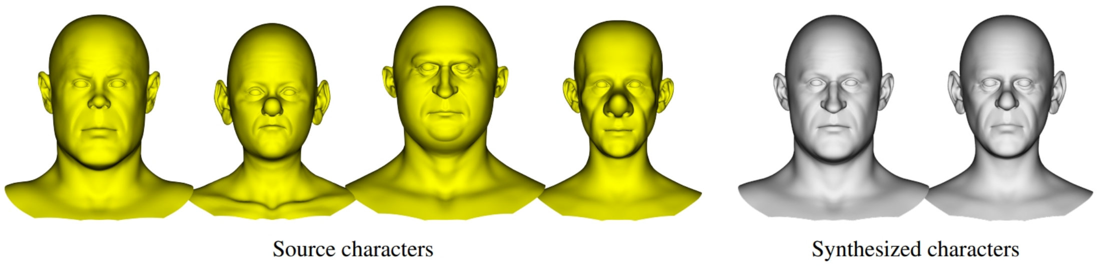


Figure 3.5: Composition of two new models from various source models. Illustration reproduced from [4]

sue structures without being biased toward predefined templates. They start by identifying the craniometric points on the skull manually, and then by adapting a broad set of anatomical rules in conjunction with the normals obtained from the skull model to produce an initial set of target face points. They create two HRBFs, one from the set of target face points, and another from the set of origin template points. The differences between the template and the HRBF surface created from the set of origin template points are then added to the HRBF surface created from the set of target face points, thus yielding the final result of the facial reconstruction. Ours method is similar to theirs regarding the composition of details over the template mesh.

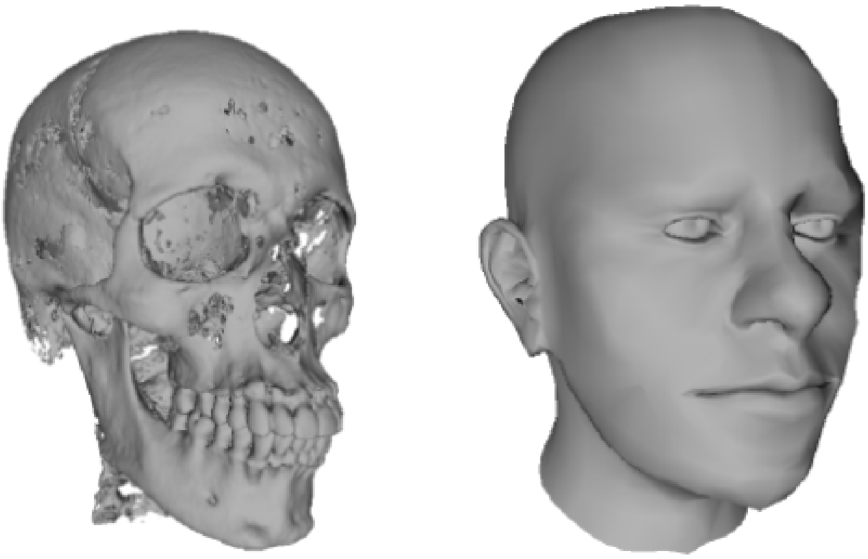


Figure 3.6: Example of an input skull model and the output facial reconstruction obtained by [5]. Images reproduced from [5].

Booth et al. [53] present an automated pipeline to construct 3D morphable models from thousands of distinct facial identities. First, they establish dense corre-

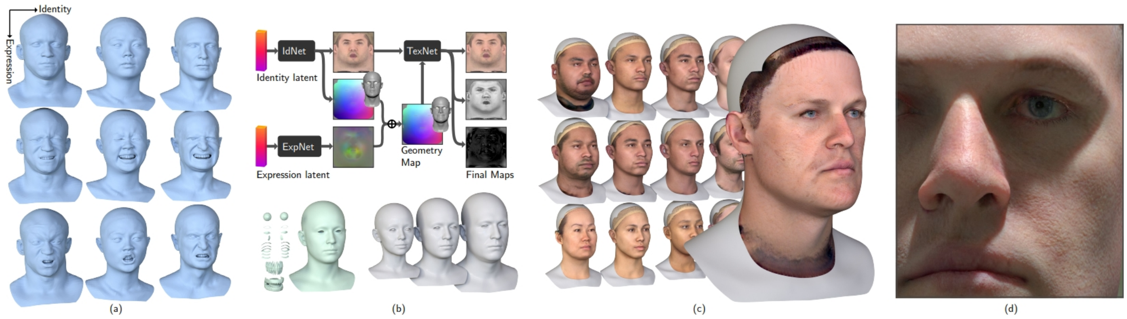


Figure 3.7: Framework for learning physically based face models present by Li et al. [6]. (a) source face models; (b) combination of anatomical and physically based attributes; (c) output faces; (d) high resolution output. Illustration reproduced from [6].

spondence using UV based interpolation methods. Then, they propose an approach for 3D landmark localization followed by dense correspondence estimation. Finally, they detect and exclude the cases of failures of dense correspondence and use PCA to construct the deformation basis. For a more extensive reading on 3D morphable models, please refer to the survey from Egger et al. [54].

Ploumpis et al. [55] propose methodology to fuse large-scale statistical model of the human head in terms of ethnicity, age and gender using both, a regression method based on latent shape parameters and a covariance combination approach. They utilize the combined models to perform full head reconstruction from unconstrained single images. Their approach builds new morphable models from meshes with different topology and that only partly overlap.

Guo et al. [56] presented a method to reconstruct face geometry and appearance from sequence of images. Albeit having a different goal since they do not generate new faces, they do employ a two scale process to recover the geometry where, in a similar manner to our work, the fine scale represents the geometric details in a displacement map.

More recently, Li et al. [6] presented a framework to generate face models from scan data. They base their method on a learning approach, training over an augmented dataset from 178 scanned faces. Their model combines anatomical and physically based face attributes to generate the new digitized faces at fine geometrical resolution. See Figure 3.7.

The main difference from our proposed method in regards to the works above, is that we aim at providing a more handcrafted way to generate new faces while being intuitive and providing control. In fact, the inspiration for our method comes from the content generation pipeline from an artistic point-of-view, and the necessity to generate new models in a small amount of time. Moreover, since we do not need training data our method works with as few as two faces.

Chapter 4

Synthesizing a Face Using Multiscale Face Models

In this chapter, we will describe the work proposed by Yoon et al [7] entitled "Blending Face Details: Synthesizing a Face Using Multiscale Face Models", which is the most related method to our proposal. As in our proposal, they do not use any special capturing setup to achieve multiscale face details. Instead, their technique starts from existing dense face models that could be sculpted by artists or obtained from conventional capturing systems.

The first step of their method is to parameterize the 3D face models on a 2D unit domain $[0,1] \times [0,1] \subset \mathbb{R}^2$. This parameterization serves as a common domain for detail transfer between faces, and also to manually create correspondences among faces when necessary. Note that differently from our case, they do not have a common template, so there is no a priori correspondences between vertices from different models.

Next, they build a hierarchical representation of the face using the uniform cubic B-spline technique [57]. This generates a multiscale representation where the first level is a very coarse and smooth approximation of the face. Then, as the level is increased, more details are inserted. They start with a coarse grid of control points as the first approximation of the face model $\mathcal{M}$ composed of vertices $\mathbf{v}$. The first B-spline surface F_1 then defines an approximation error : $e^1 = \mathbf{v} - F_1(u,v)$.

Following, they repeatedly refit the errors by using higher resolution grids, creating better approximation surfaces. They suggest a first grid of resolution 4x4, and then at each level increase each dimension by 1 unit (i.e. 5x5, 6x6 ...). For example, the second residual would be the difference from the first surface and the second approximation error: $e^2 = e^1 - F_2(u,v)$. At each level the approximation surface

$S_l(u,v)$ can then be defined as:

$$S_l(u,v) = \sum_{i=1}^{l} F_i(u,v)$$

Figure 4.1 illustrates the results of approximation to the face model at different levels.

From the approximation surfaces they then build a Multiscale Face Model (MFM). This model will only use two approximation surfaces from the initial hierarchical representation and the rest will be discarded. One of the coarsest approximation levels is chosen as the base surface S_b of the MFM, and the surface S_d one level higher is used to create an initial displacement vector field, defined as $D_0 = S_d - S_b$, where $d = b + 1$. They observe that if no self-intersection occurs, the original face model can be reconstructed by displacing the surface S_d along its normal direction until reaching $\mathcal{M}$.

As with the first hierarchical representation, this vector displacement is again approximated by a multilevel B-spline function f_l. Finally they define a set of multiscale Continuous Displacement Maps (CDMs), to store the level of details to be transferred between faces. A CDM D_l is an artist defined grouping of subsequent displacement functions f_*. In the paper they always choose to create 3 groupings for the examples. The final Multiscale Face Model can be then defined as:

$$\mathcal{M} = S_b + D_0 + N \sum_{l=1}^{3} D_l \tag{4.1}$$

The idea is that by successively adding the details to the approximation surface S_d in the normal direction it is possible to return to the original face surface. Note that D_0 is a vector CDM, while $D_{l>0}$ are scalars CDMs. Note also that in Equation 4.1 $S_b + D_0$ could apparently be replaced directly by S_d, but as will be explained next, S_b and S_d from different faces will be merged to form new faces. Figure 4.2 illustrates the MFM defined by Equation 4.1.

The MFM is created for different face models that are to be combined. The new face is then created by choosing a S_b from one surface and mixing the other components of the MFMs ($\{D_0, N, D_{l=\{1,2,3\}}\}$ from different models. Given a set of X original faces, and weights α_l^x corresponding to the weight of face x at CDM with level l, the new face model is defined as:

$$\mathcal{M}^{new} = S^A + N^A \sum_{\mathcal{M}^x \in X} \sum_{l=0}^{3} \alpha_l^x D_l, \tag{4.2}$$

where $S^A = S_b^A + \sum_{\mathcal{M}^x \in X} \alpha_0^x D_0$. Figure 4.3 shows some results of blending

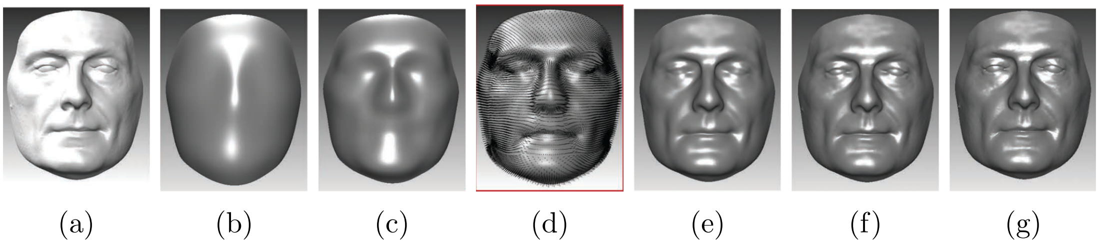


Figure 4.1: Surface approximation by multilevel B-spline: 4.1a is the source face model, (b-g) are examples of approximation surfaces $S_l(u,v)$ at different levels $l = 4, 5, 6, 7, 8, 9$. The surface (d) is shown with normal vector fields. Figure reproduced from [7].

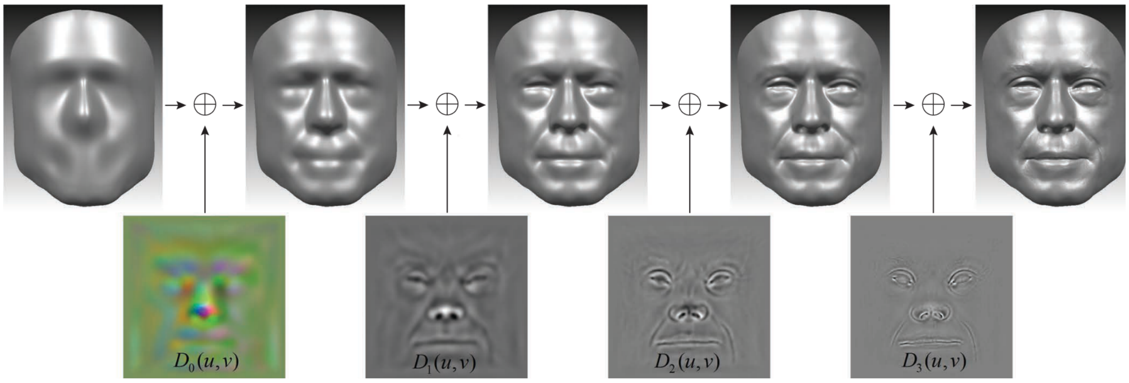


Figure 4.2: Illustration of the process to create an MFM. The leftmost face is S_b, that combined with vector field D_0 produces S_d (second face). Then, the scalar CDMs $D_{l=\{1,2,3\}}$ area use to displace S_d along its normal direction. The final face model $\mathcal{M}$ is shown on the far right. Figure reproduced from [7].

multiscale human faces.

4.1 Discussion

The work proposed by Yoon et al [7] has as main target designing faces, not expressions. Their method has similarities with the Blendshapes approach although being more complex and embracing. While synthesizing a new 3D face model using weighted blending of multiscale details across different faces including human and nonhuman characters, they decompose face models into component scales with a correspondence of salient facial features across faces, building a hierarchical representation of the face's spatial details. The parametrization of the face is done in 2D space and decomposed into a base surface and multiscale continuous displacement maps. There are full correspondence between the displacement maps and the hierarchical representation of the face details. Their method can blend details between faces without vertex correspondence. However, it suffers when dealing with holes in the mesh. In some situations this is a considerable drawback since the face model

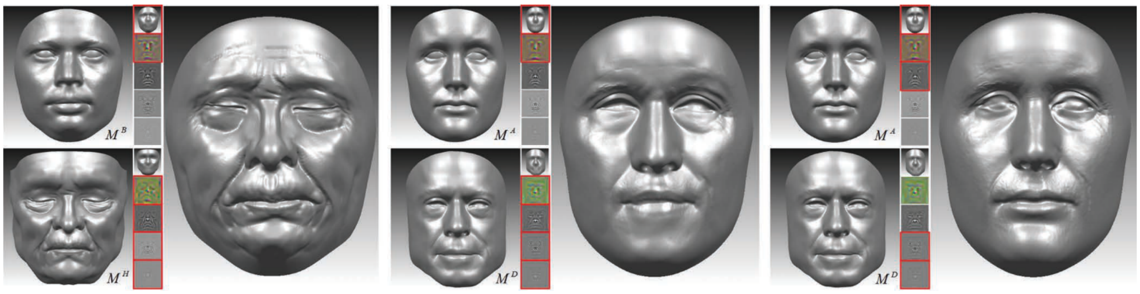


Figure 4.3: Three results of blending multiscale human faces. Each new face (larger images) is created from two original models (smaller images). Besides each original face its CDM is illustrated, and the red squares mark which components were used, i.e., have weights larger than zero. For the leftmost example, with original faces $\mathcal{M}^B$ and $\mathcal{M}^H$, the base surface S_b^B of face B is chosen and the blending weights are: $\vec{\alpha}_l^B = (0.5, 0, 0, 0)$ and $\vec{\alpha}_l^H = (0.5, 1, 1, 1)$. The weights for the other examples were not specified in the original paper. Figure reproduced from [7].

may actually contain cavities on eyes, mouth, and nostrils. Our method can handle these issues seamlessly. Besides, their method requires intense user intervention, both for parameterization adjustments and for mesh editing.

Many details are omitted in the original paper and some definitions are not very precise. In this chapter we have tried to describe the original paper as best as possible and filling the missing parts with the most plausible interpretation we could find. Apart from some missing details, the paper is unfortunately not reproducible since there is no accurate definition of how the CDMs are build (which levels are grouped together). The authors simply state that the CDMs were composed by artists.

Chapter 5

Our Method

In this chapter we will describe the method developed to generate new faces by blending existing ones. Given a data set of faces, the goal is to create new ones taking into considerations specific characteristics, that may come from a racial or ethnic background, or from other factors such as age. See figure 1.1.

As a chronological retrospective of the research, we begin by looking for ways to mix two faces into a new one. Although some methods could give us interesting results, their capacity to create original models was limited to the linear combination between the existing ones. Besides, we were not searching just for an interpolator for 3D models, rather for a tool to allow artists to design new faces. It was required, for instance, to be able to keep the base anatomy of one face and apply a layer of details from another face. This kind of combination is not possible using Blendshapes [9, 19].

Our first inspiration was the method by Romeiro et al. [5] to add details from a template face to a coarse facial reconstruction from a skull and craniometric points. However, they use HRBFs to create the coarse facial reconstruction and as base to generate their displacement vectors. In our case, it is important to use the original mesh and to preserve the topology during any operation. In addition, we have a different problem where we have high-resolution meshes with much more details than in the case of facial reconstruction. Having a common topology also entails in not needing to create correspondences between faces, as was manually done by Yoon et al [7] (Chapter 4). We then opted for a mesh smoothing operator that would gradually remove details from the face without changing the underlying topology and, as described next, has other important properties for our requirements.

5.1 Laplacian Smoothing

The idea behind the Laplace equations for mesh smoothing is that the solution of the Laplace equations satisfies the minimum/maximum principle. In other words, this means that the values of the interior displacements are bounded by the values

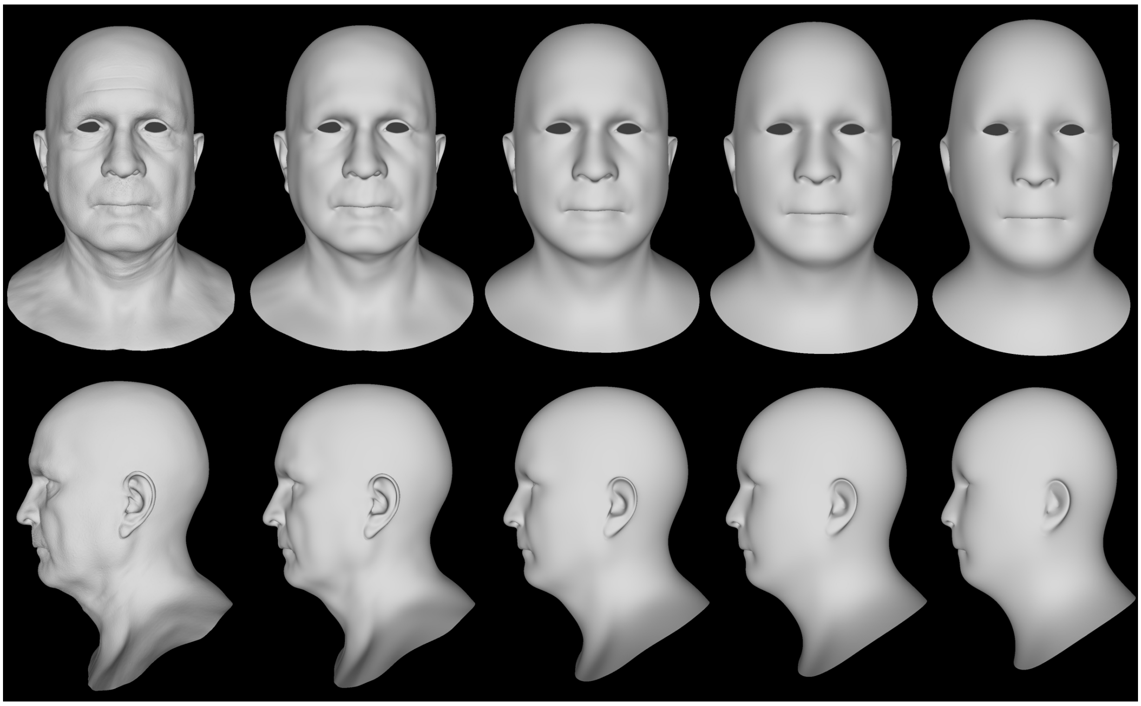


Figure 5.1: Example of the Laplacian smoothing operator applied to a face mesh. Four levels of smoothing for a mesh with approximately 1.2 million vertices. From the left to right, the original model is followed by smoothed meshes using 500, 2500, 5000 and 10000 iterations, respectively.

on the boundary surfaces. This ensures that the interior nodes will not cross the mesh boundaries, a fundamental property in our case since we want to preserve the boundaries of the eyes, mouths and nostrils, for example.

Briefly, the Laplacian Smooth operator works by iteratively displacing each vertex based on local information, such as the position of neighbors. According to Hansen et al [58], the Laplacian smoothing operation on vertex v_i may be defined as:

$$\bar{x}_i = \frac{1}{N} \sum_{j=1}^{N} \bar{x}_j, \tag{5.1}$$

where N is the valence of vertex i, $\bar{x}_j$ is the position of the j-th adjacent vertex of v_i, and $\bar{x}_i$ is the new position for v_i.

Figure 5.1 shows an example of Laplacian smoothing being applied to a facial mesh. This mesh has approximately 1.2 million vertices. The amount of smoothness is defined by the number of iterations over the mesh.

5.2 Blending details between faces

In this section we describe our method to blend multiple faces by carrying the details of one or more source meshes to a target one. We start by describing how to blend two faces. Given a source C and a target B, we produce a new 3d model A by transporting the details from C to B in the following way:

$$A = B^L + (C - C^L), \tag{5.2}$$

where B^L is the face B smoothed by L Laplace iterations, and likewise for C^L.

Since the meshes have the same topology and a one-to-one correspondence between vertices, which also share the same uv texture coordinates, we only need a single parameter to control the blending results, the number of iterations L for the Laplacian smoothing. Higher values of L lead to more smoothing of the source and target faces and, consequently, to more details being transferred from the target to the source mesh. Figure 5.2 illustrates the process of blending two faces.

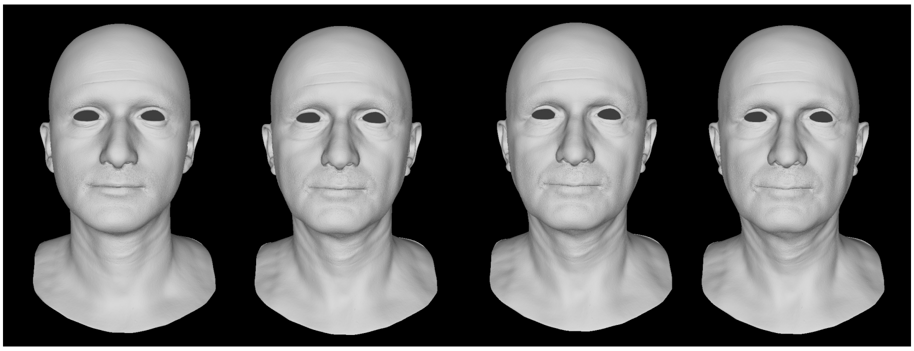


Figure 5.2: Blending the source and target faces from Figure 1.1 with the following number of smoothing iterations L from the left to right: 10000, 14000, 18000 and 20000.

The method may be extended for multiple faces. In this case, Equation 5.2 becomes:

$$A = B^L + \frac{\sum_{i=1}^{n} \omega_i (C_i - C_i^L)}{\sum_{i=0}^{n} \omega_i}, \tag{5.3}$$

where n is the number of source faces to be used in the composition and ω_i is the weight for source face i. We normalize the contributions to avoid aberrations.

We define the weight ω_i of each face using Euclidean distances measured on a 2d plane. Faces are manually arranged on a 2d plane (see figure 6.1), where t_i is the position of face i. Then, for any 2d blending position m the weight of face i is defined as the inverse of the squared distance between m and t_i:

$$\omega_i = \frac{1}{||m - t_i||^2}. \tag{5.4}$$

To avoid computing B^L and C^L every time a new smoothness factor L is selected, we define approximated smoothed faces $\tilde{B}^L$ and $\tilde{C}^L$, as described in Section 5.2.1. In the same way, we define the approximated displacement vector that inserts details in $\tilde{B}^L$ as

$$\tilde{D} \approx \frac{\sum_{i=1}^{n} \omega_i (C_i - \tilde{C}_i^L)}{\sum_{i=0}^{n} \omega_i}. \tag{5.5}$$

Finally, the final mesh for any smoothness level L becomes

$$A = \tilde{B}^L + \tilde{D}. \tag{5.6}$$

Notice that the weights are normalized but, by removing this restriction, we can also create faces that extrapolate the details. This interesting side effect is achieved by modulating the weights by some factor, as illustrated in Figure 5.3.

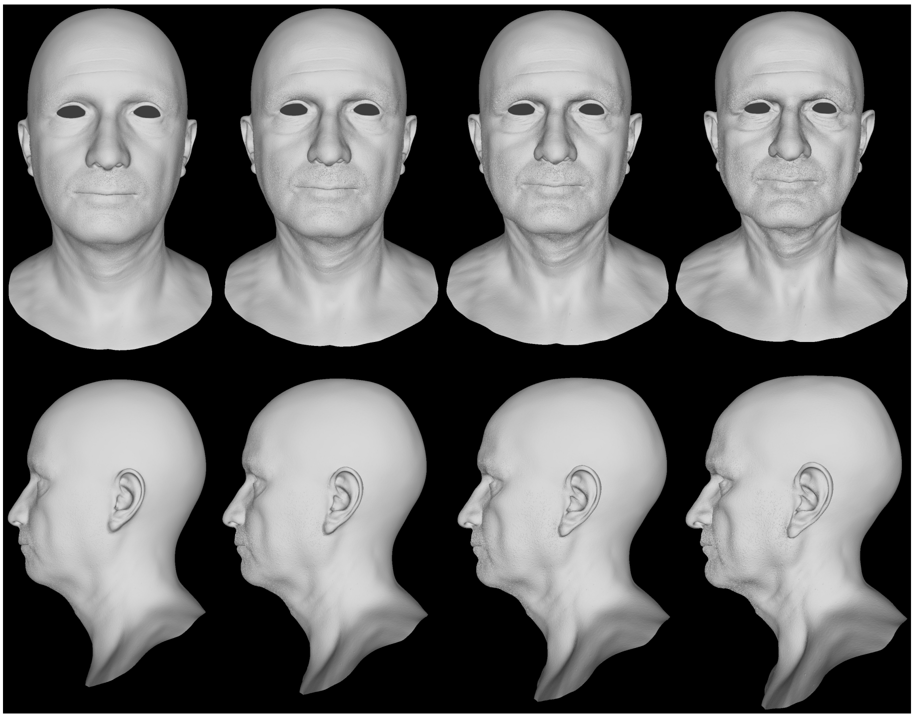


Figure 5.3: Detail extrapolation by manipulating the weights from the example shown in Figure 1.1. From the left to right scaling all weights by 0.75, 1.0, 1.2 and 1.4. Notice that not only details are intensified but the global shape of the face is also deformed for high values.

5.2.1 Approximating smoothness levels

To compute approximations $\tilde{B}^L$ and $\tilde{C}^L$ of the smoothed levels, we first define a maximum level L_{max}. A naive way to approximate the intermediate smoothed levels is to linearly interpolate between the original face and L_{max}. Naturally, linearly interpolating the displacement vector is not the same as computing the Laplacian smoothing operator a given number of times. With this approximation, some fine details remain after smoothing the mesh and are not properly transferred. Consequently, $\tilde{B}^L$ and $\tilde{C}^L$ result in poor approximations of B^L and C^L. Instead, we approximate the displacement of each vertex along with the multiple smoothing operations by a cubic polynomial using a least-squares method [59–61]. Moreover, since it has a fixed number of parameters, we also have a constant representation of the displacement vectors independently of L_{max}.

Nevertheless, fitting the curve using all L_{max} samples would impose a large memory overhead as we need to store all intermediate smoothed faces. Considering only the vertices, 1.2 million per face, we would require approximately 13.73Mb per face. Thus, we select a few levels to be stored in order to accelerate the pre-processing and avoid storage issues. Since the finer details are smoothed out after a few iterations, as illustrated in Figure 5.1, we use more levels from the first iterations and less nearer L_{max} in order to preserve these details. In our experiments, we use around 40 levels to fit the curve, which is a massive reduction since L_{max} is typically larger than 10000.

The maximum error between the curve fitted with all levels and the original smoothed faces is very low, around 1×10^{-7} cm. This error is the distance between the vertices of each laplacian iteration of a face and the vertices resultant from the curve interpolation fitted using all smoothed levels. Using only 40 levels to fit the curve results in a slightly higher error, around 3×10^{-6} cm. Nevertheless, it is still negligible for any visual purposes.

To illustrate it better, in Figure 5.4 we show two sample vertices extracted a face, one in the nose region and the other in the lips region. Their trajectory are approximated using five different methods: a linear interpolation, a linear interpolation using an additional sample in the middle of the trajectory path, a polynomial of degree two and a polynomial of degree three, as illustrated in Figure 5.5. The computed error distance for each of the methods illustrated is shown in a bar graph in image 5.6

5.3 Multiple Regions

To add more control and broaden the possible space of faces, we segment the face into separate regions, where for each region we can apply the method separately. This is, naturally, optional and one could decide to split the mesh into any number of regions. For our test cases we defined three horizontal regions by creating a colour map on the texture *uv* space. We assigned three different regions with transition bands between them to avoid abrupt changes, as shown in Figure 5.7. The regions were manually defined using a simple image editor software, and we used a gradient

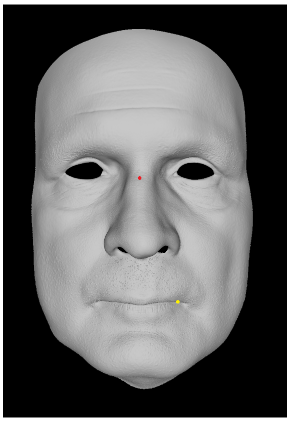


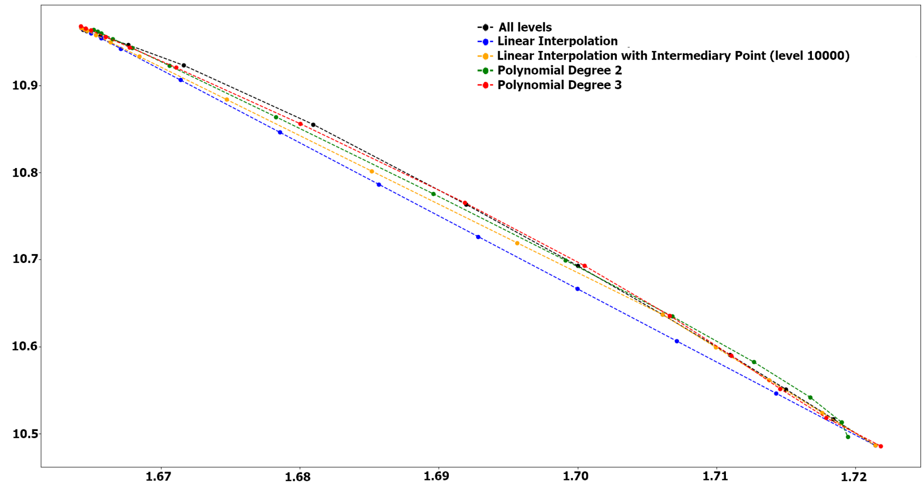
Figure 5.4: The image shows two sample vertices extracted from nose and lips region. These are the vertices we used to explain bellow our error measurements.

brush for the transition bands. Nevertheless, there are no restrictions on how the regions may be defined and, for example, a 3D brush can be used to paint directly over the mesh to assign regions to vertices.

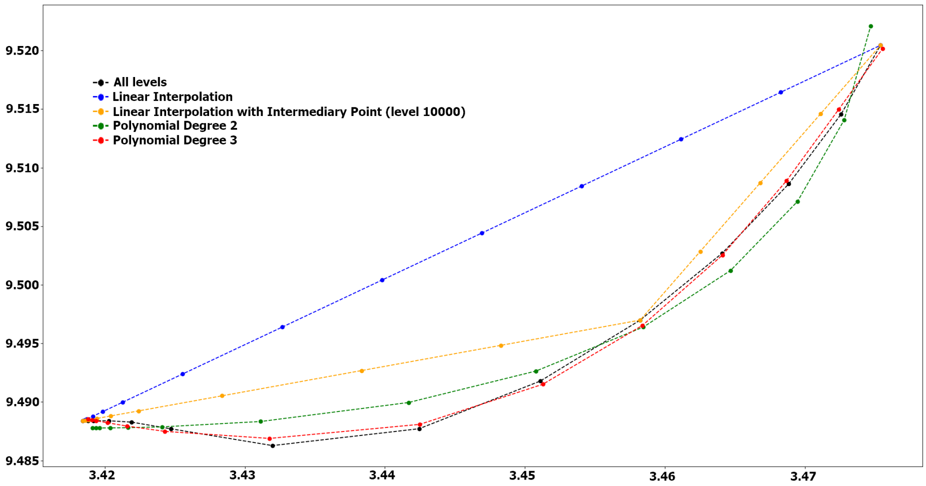
Taking into account the regions mask, Equation 5.6 now becomes

$$A = \sum_{k=1}^{r} (\tilde{B}^{L_k} + \tilde{D}_k) * \alpha_k, \tag{5.7}$$

where α_k is a scalar in the range $[0, 1]$ that defines the weight for region and is taken from the mask color channel, r is the number of defined regions, $\tilde{B}_k^{L_r}$ is the smoothed region k of the approximated target mesh $\tilde{B}$ and $\tilde{D}$ is the approximation displacement vector as defined in Equation 5.5. Note that we can define different levels of smoothness L_k for each region.

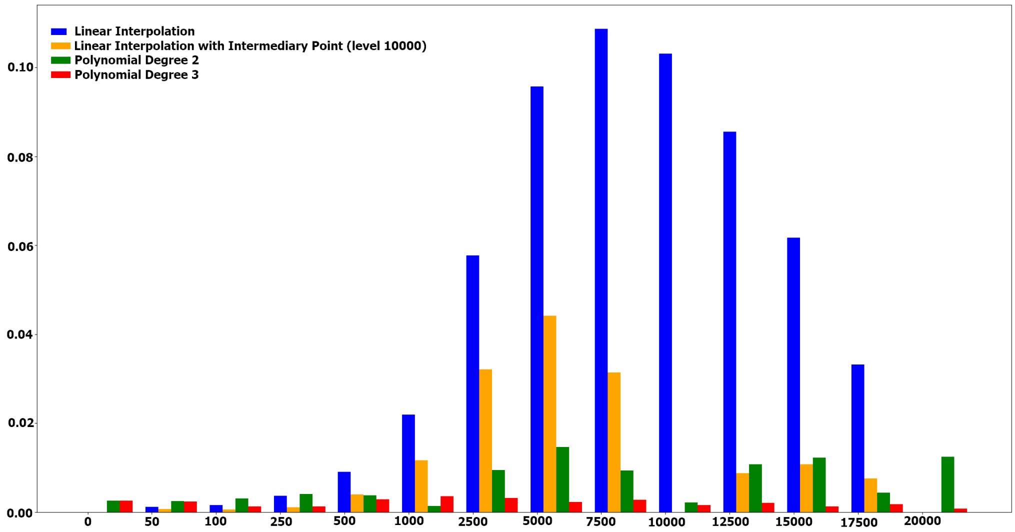


(a) The trajectory path for a sample vertex in the region of the nose. The abscissa axis correspond to the displacement of x coordinate and the ordinate axis shows the displacement for z coordinate. The displacement on y coordinate is omitted in this example due to negligible variation.

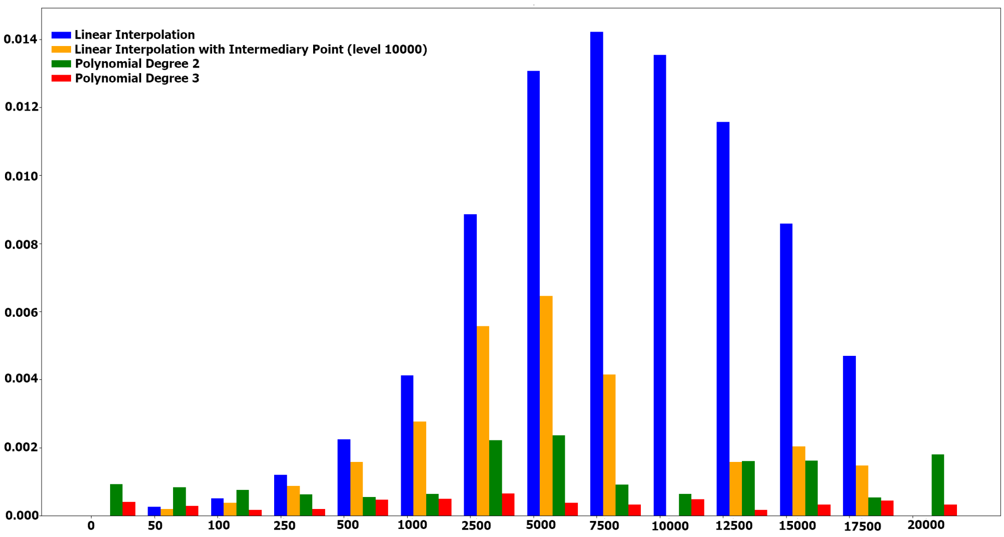


(b) The trajectory path for a sample vertex in the region of the lips. The abscissa axis corresponds to the displacement of x coordinate and the ordinate axis shows the displacement for z coordinate.The displacement on y coordinate is omitted in this example due to negligible variation.

Figure 5.5: Comparison between different interpolation methods and their distance measurement to the original vertices path in a test case with 20000 smoothing iterations.



(a) The distance error measured for the sample vertex in the region of the nose



(b) The distance error measured for the sample vertex in the region of the lips

Figure 5.6: The distance error measured between different interpolation methods for the two sampled vertices shown in Figure 5.4

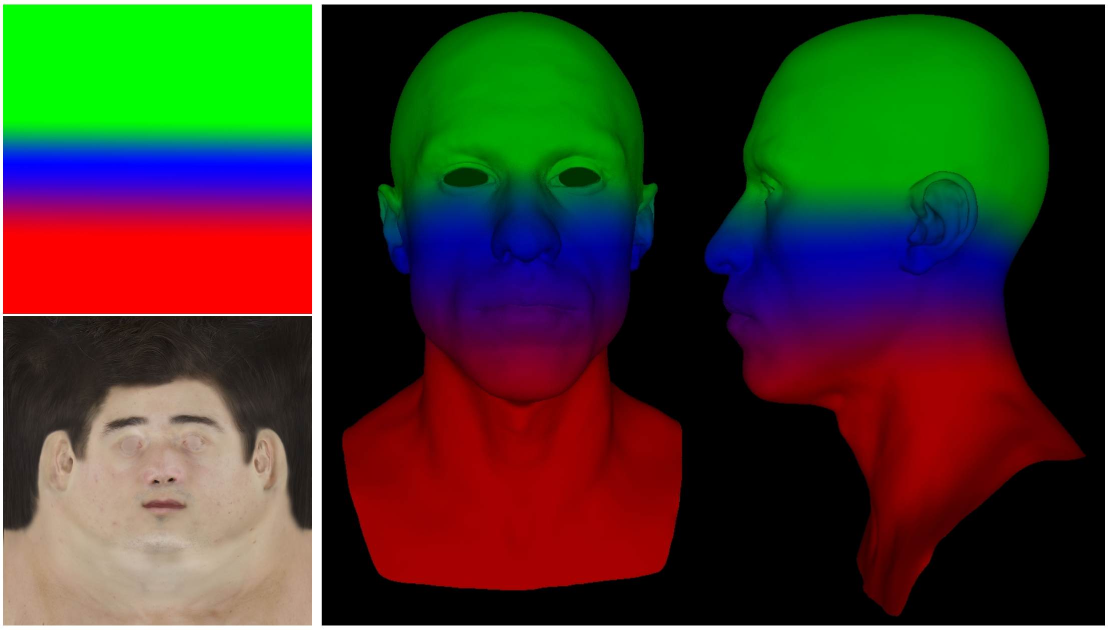


Figure 5.7: An example of using region masks manually defined. On the left, one of the original textures and the painted regions in texture space. Note the blending region between the top (red) and middle (blue), and middle (blue) and bottom (green) masks to achieve a smooth transition. On the right, the resulting regions applied on one of the heads.

Chapter 6

The Software

In order to perform tests and validate our method, we have developed a real-time visualization system with a simple and intuitive interface. With this tool, it is possible to load the head meshes and textures, select the meshes we want to blend and dynamically move the controllers and change the blend parameters. For example, parameter L in Equations 5.2 and 5.3 is controlled by a slider as a percentage value $\frac{L}{L_{max}}$. The blending position m that defines the blending weights is controlled by moving the position of a circle. The location of each thumbnail representing a model can also be dynamically changed by moving them and, thus, influencing more or less the final blending. Figure 6.1 shows the interface and some of its functionalities.

In our prototype we have included a controller for each face region, in order to manipulate each weight independently. Furthermore, we include one slider to set the smoothness level L_k for each region.

The only pre-processing required is smoothing each face with L_{max} iterations and fitting the cubic curves to approximate the vertex displacements. In our tests with L_{max} equals to 15k iterations, this pre-processing time was under 5 minutes per face. We have not made any efforts to further optimize this pre-processing since it can be seen as a final step of the acquisition pipeline which, in turn, may take in total a couple of days considering the manual artistic efforts involved. In addition, the face needs only to be prepared once and can be used multiple times within the system to generate new faces. Consequently, the time to prepare the smoothed head model is negligible within the production pipeline.

We use the original quad meshes for all operations and only transform into a triangular mesh for rendering purposes. In cases of very high resolution meshes or poor hardware, a potential strategy to keep a real-time responsive interface is to use only the front part of the subject's face instead of the whole head. This approach would reduce the computational costs but still maintain the most significant part of the face in regards to details. Once the desired face is achieved through the interface, the system can easily export the parameters and generate the final solution for the whole head and optionally use directly B^L and D^L in Equation 5.7, instead of the approximated $\tilde{B}^L$ and $\tilde{D}^L$ versions using the cubic curve. An example of a post-processed full head blend can be seen in Figure 1.1.

We have implemented and tested our method in a desktop with an Intel Core i7 7700 with 32Gb of Ram and an nVidia RTX 2080 Ti with 11Gb. After loading all smoothed meshes and associated displacement vectors, the system runs entirely in GPU using GLSL shaders. All blending operations are performed interactively with results being immediately composed and visualized. The maximum number of loaded faces is initially bounded by the graphics card video memory. Even though we have not reached this limit in any of our experiments, this restriction can be circumvented by employing a multi-pass strategy. Another option is to lower the mesh resolutions and only apply the blending to high-resolution meshes in CPU when the face creation process is finished. Nevertheless, so far we have not noticed the need for such solutions since we tested our method with up to 15 high-resolution faces while maintaining real-time frame rates.

The source code developed for this project is available at gitlab.com/diegomazala/faceblending under general public license.

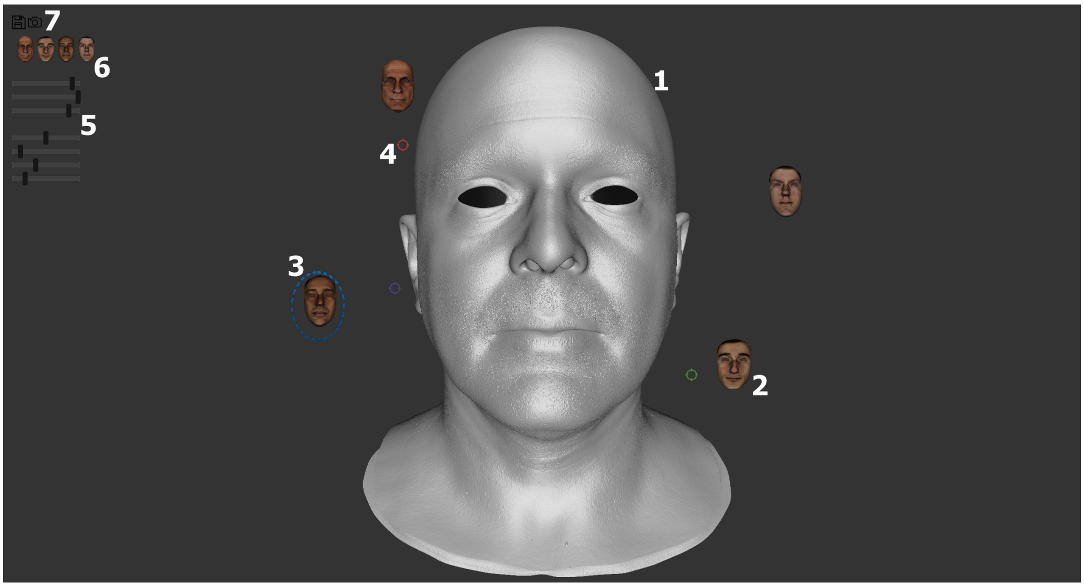


Figure 6.1: The main screen of our interface. (1) resultant mesh A; (2) one of the thumbnails representing a source face; (3) dashed blue circle indicates selected target face B; (4) a controller for a specific region defining position; (5) sliders to control the smoothness level for each region; (6) available textures that can be applied to the models; (7) buttons to save resulting model or image.

Chapter 7

Results and Comparisons

7.1 Results

In this section we present some results obtained using head meshes with 1.2 million vertices. Pre-processing each face takes approximately 12 minutes for each face but this step needs to be done only once.

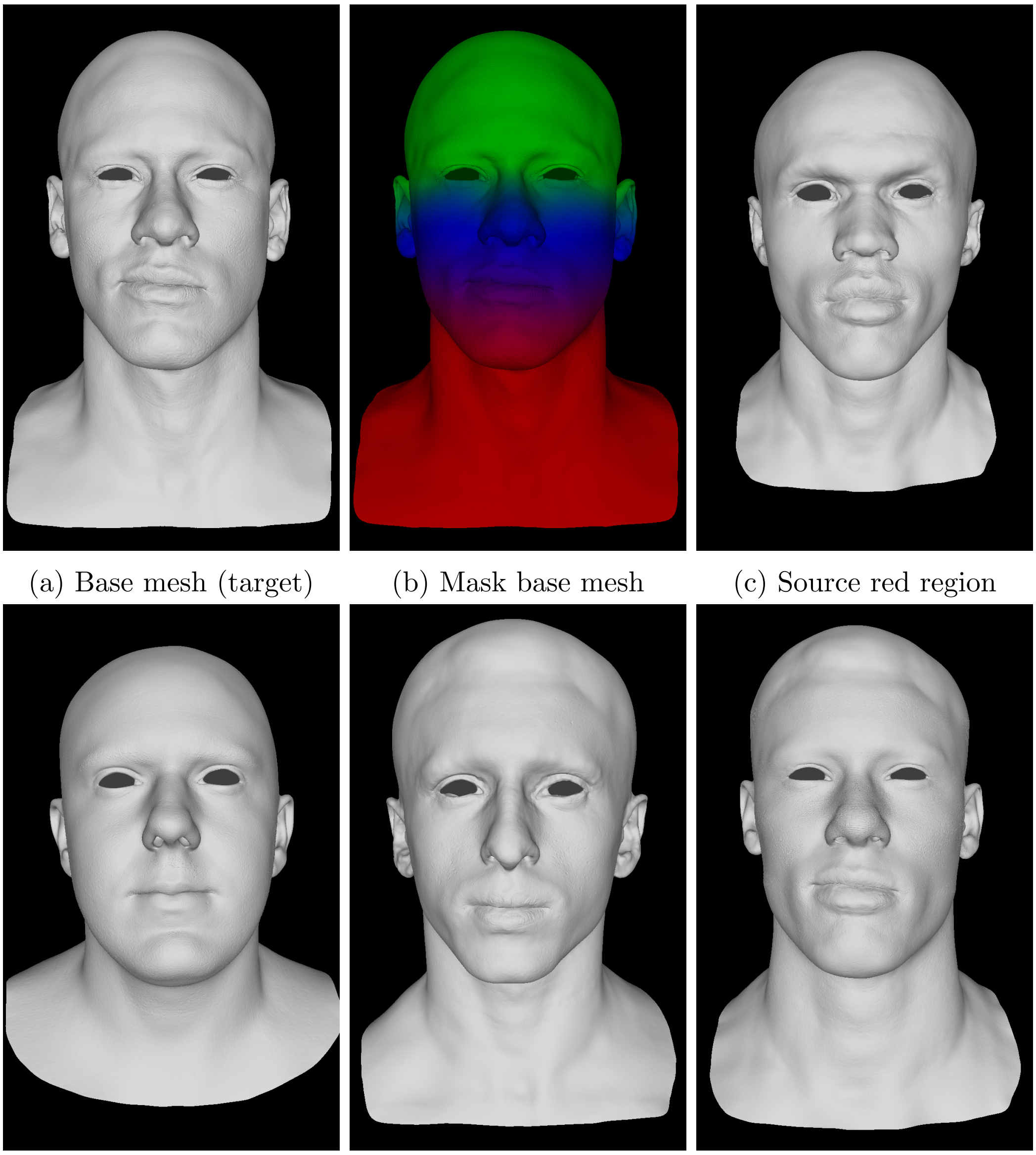
Although the method can produce general results as shown in Figures 7.2, specific combination of faces may also carry other high-level semantic meanings. Note in Figure 1.1 how by transferring details from an older to a younger person the result can be seen as an ageing effect.

Figure 7.1 shows a result of blending three different faces, where each face only influences one of the mask regions. Note how for the lower region (red) the lips become thicker but the crooked smiled is preserved, and the fine details from the chin are also transferred. For the nose and eye region (blue) the nose keeps its general shape but retrieves the details such as nasal bone format. The eyes maintain the thin format but receive the contouring lines from the source face. Finally, the forehead region (green) becomes more triangular shaped.

For all tests we have achieved real-time performance. Apart from a constant overhead, the time increases linearly with the number of faces, as expected. As shown in Table 7.1, even with 15 faces our systems runs at 40 FPS.

# faces	2	4	7	10	13	15
# vertices (millions)	2.4	4.8	8.4	12.0	15.6	18.0
ms/frame	14.1	15.6	17.5	20.4	23.2	25.0

Table 7.1: Performance when increasing the number of faces.



(a) Base mesh (target) (b) Mask base mesh (c) Source red region

(d) Source blue region (e) Source green region (f) Result

Figure 7.1: Example of using multiple faces with different weights per region. Here, each source face has full weight for only one specific region to better illustrate the effect.

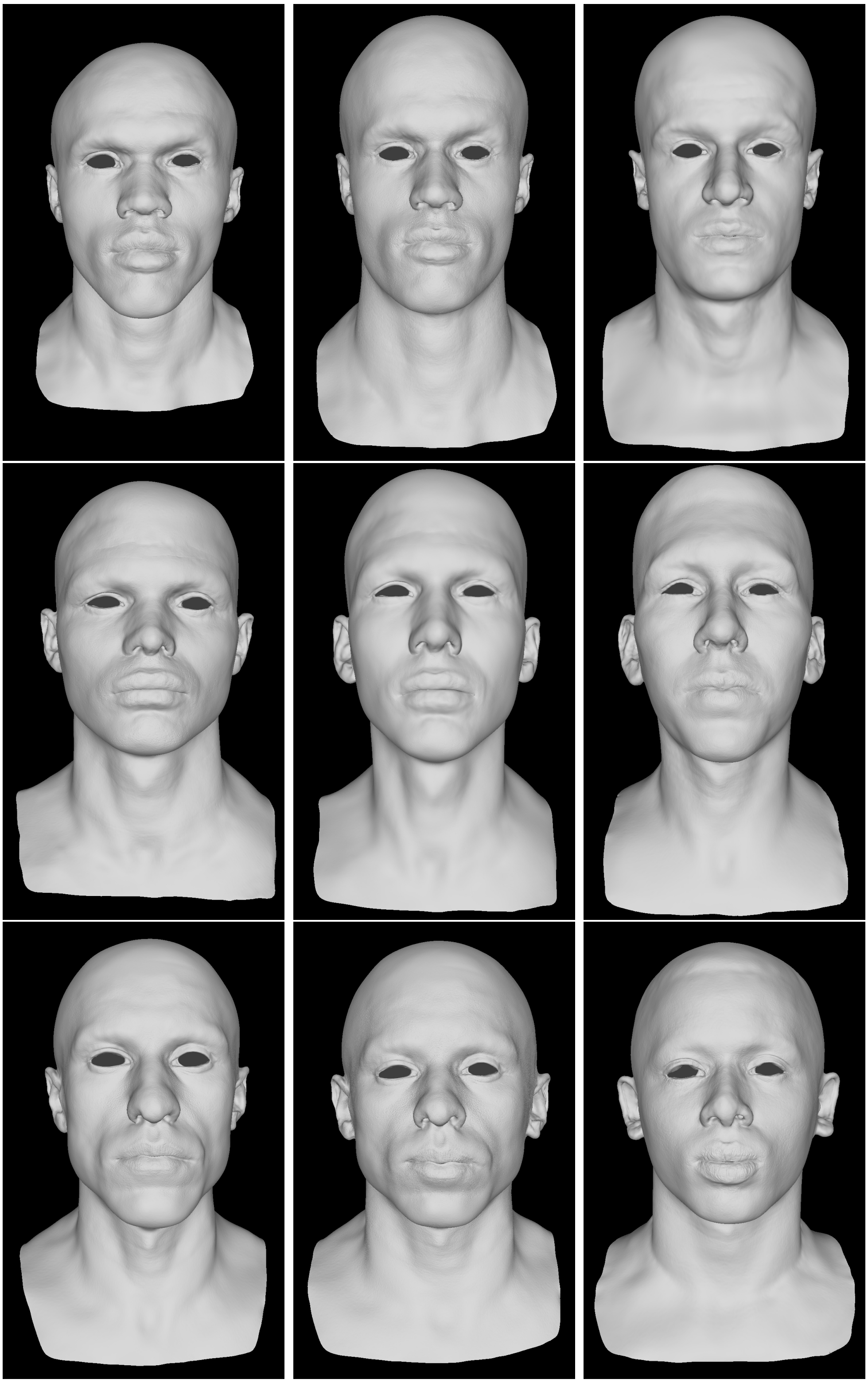


Figure 7.2: Some results of our approach applied on meshes with 1.2 million vertices. For each row, column (a) shows the source face from the details are extract, column (c) shows the target face and column (b) the blending result.

7.2 Comparisons

Here, we compare the results achieved from our method against those obtained by Yoon et al. [7] described in Chapter 4.

Figures 7.3, 7.4 and 7.5 show in a color scale the distance between the vertices of the models built using our method and the method by Yoon et al. [7]. We use a simple Euclidean distance:

$$D_i = \sqrt{(\mathcal{M}^0_x i - \mathcal{M}^1_x i)^2 + (\mathcal{M}^0_y i - \mathcal{M}^1_y i)^2 + (\mathcal{M}^0_z i - \mathcal{M}^1_z i)^2},$$

where D_i is the distance for the ith vertex of the our model $\mathcal{M}^0$ and Yoon's model $\mathcal{M}^1$. Remember that $\mathcal{M}^0$ and $\mathcal{M}^1$ are vertex wise correspondent meshes. The color scale is composed by four equally distant colors . It starts in $RGB(0,0,1)$, going through $RGB(0,1,0)$ and $RGB(1,1,0)$, and ending in $RGB(1,0,0)$.

Since there is no formal way to correlate the amount of details transferred between our method and theirs, we select the correspondence between the meshes visually. That is, we have chosen the CDM and the weights for Yoon's method and found the best match in order to have a fair comparison with our results.

Looking at the results (Figures 7.3, 7.4 and 7.5), it may be seen that in the cases with less smoothing there are more significant differences on the nose and over the eye-browns. When more smoothing is applyed, the differences are more distributed over the whole face.

Figures 7.3, 7.4 and 7.5 show the same process for three different models. The parameters used in the tests are the same for the three cases. Every sub-image (a) shows our result with 100 smoothing iterations; (d) with 7500 iterations; and (g) with 19000 iterations. Every sub-image (c) shows Yoon's result with weights 0.92 for the base face and 0.08 for the detail head; (f) equal weights 0.5 for each face; (i) weight 0.08 for the base head and 0.92 for the detail face. Sub-images (b), (e) and (h) show a distance representation between the vertices of the both meshes.

There is a unwanted result in Yoon et al. [7] method: it does not handle boundaries correctly. Figure 7.6 depicts this problem. Indeed, they do not offer the source code for their software and their paper lacks relevant information to allow a faithful implementation. However, as much as we could, we tried to implement their proposal and reproduce the results for comparison in a fair way. It is not clear if the boundary problem is a true limitation of their method, or if the problem did not arise before since all their meshes are apparently watertight. Notwithstanding, we made an effort to correct this issue in our implementation of their method, but found no straightforward way to handle it.

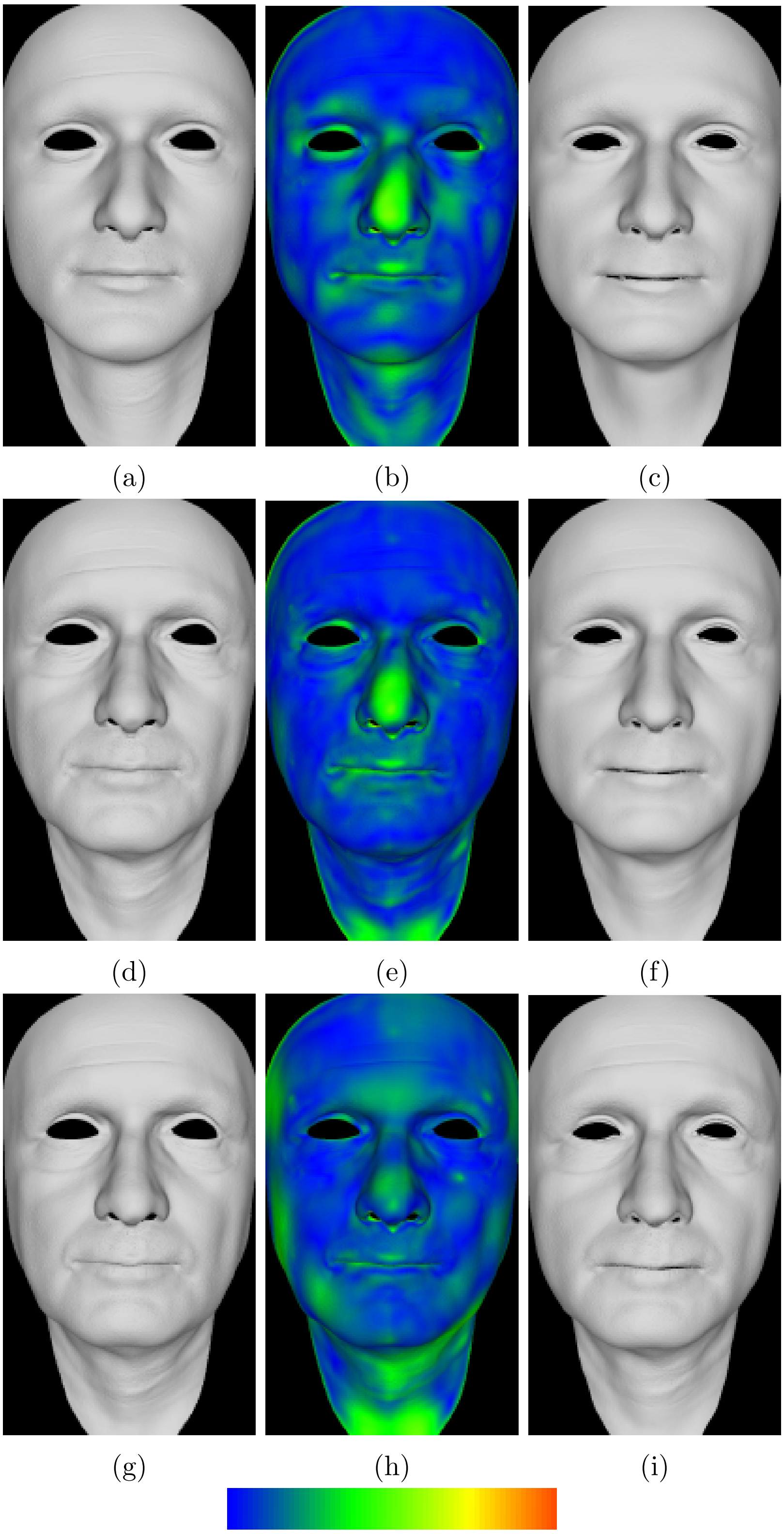
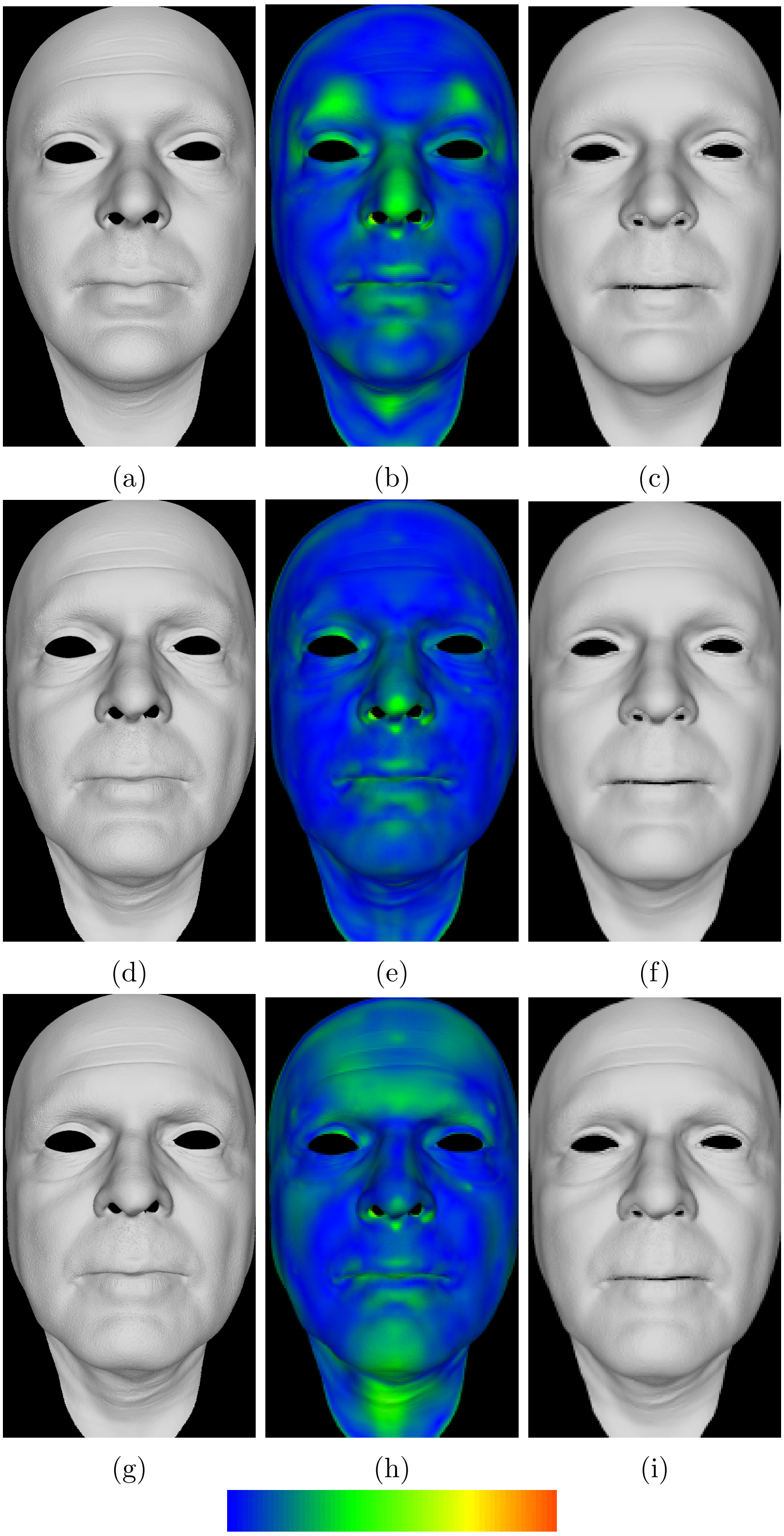


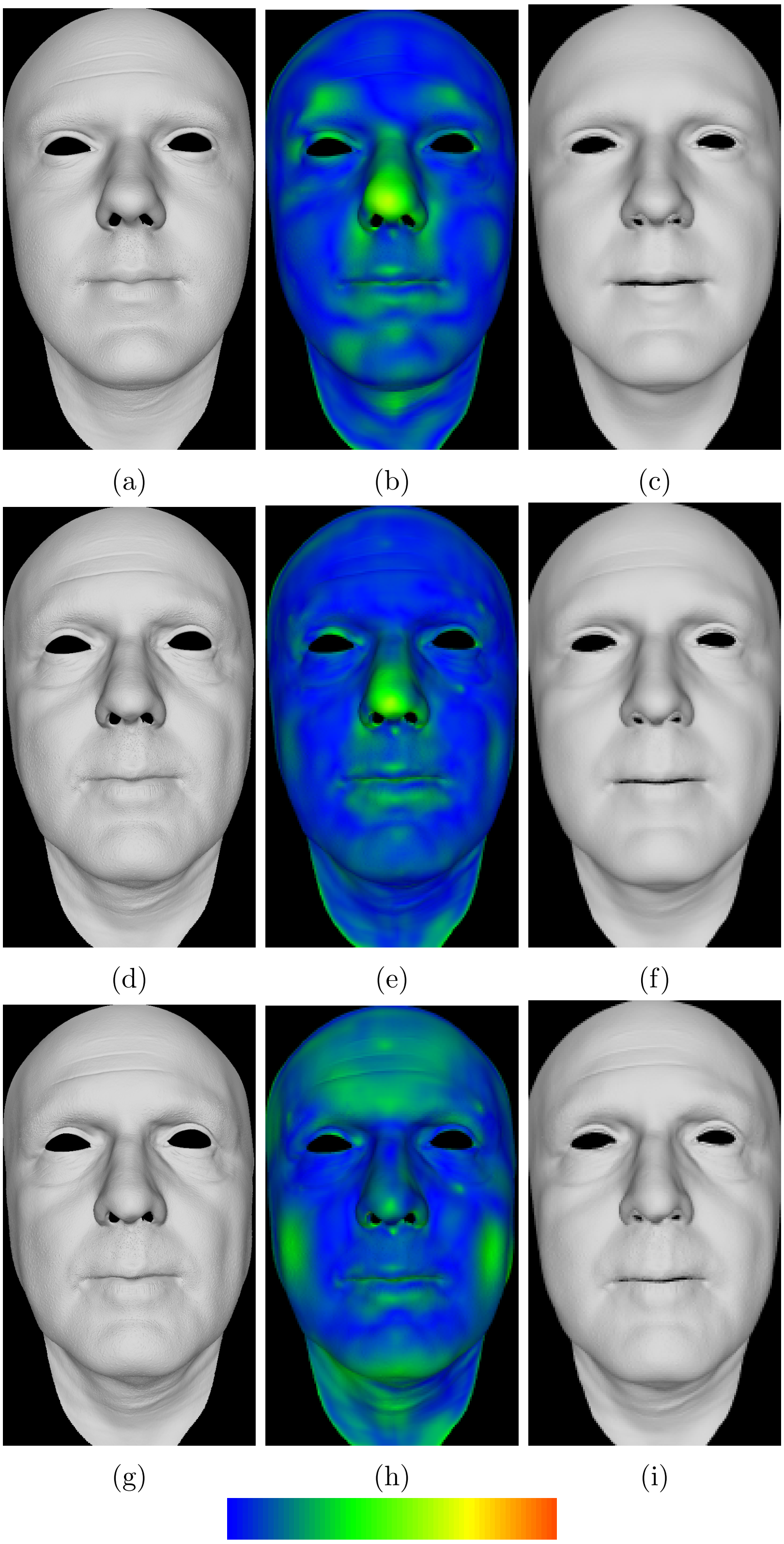
Figure 7.3: Comparison between our method and Yoon [7] proposal. The colors show the distance between the vertices of the meshes. The left column are results from our proposal; the right column from Yoon's method; and the center column is the difference between both.


(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure 7.4: Comparison between our method and Yoon [7] proposal. The colors show the distance between the vertices of the meshes. The left column are results from our proposal; the right column from Yoon's method; and the center column is the difference between both.



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure 7.5: Comparison between our method and Yoon [7] proposal. The colors show the distance between the vertices of the meshes. The left column are results from our proposal; the right column from Yoon's method; and the center column is the difference between both.

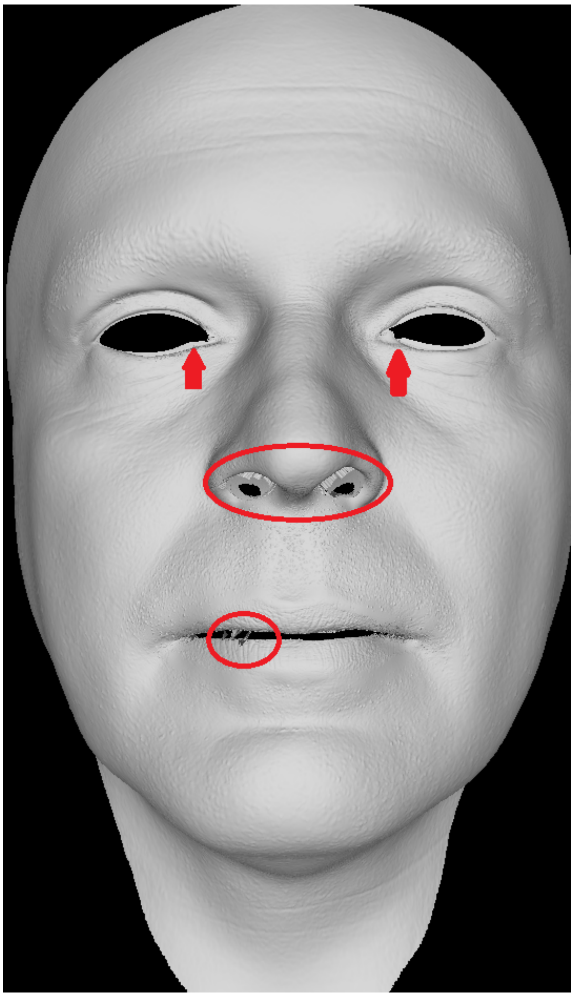


Figure 7.6: The circles and arrows in red show the boundaries problems we found in our implementation of Yoon et al [7].

Chapter 8

Conclusion

We have presented a method to generate new faces from a set of digitized high-resolution models. Our method provides an intuitive and simple way to create new faces by transferring details from one or more source faces to a target. Our approach is inserted into a content creation pipeline where faces are digitized to produce background characters. In this way, our method provides a fast way to generate new realistic characters. We place our method in a category of hand-crafted methods, where more control is given and less data is needed, in contrast to methods that rely on learning approaches. We have also provided a real-time interface for rapid creation of new faces.

Although we presented interesting results, we see some limitations that could be addressed in future steps. For instance, our method cannot handle faces with different mesh topologies. This is not a problem in many content creation pipelines since all faces are digitized the same way and are mapped to a common parametrization, but it still imposes a limitation when incorporating external models. Even though this was not the focus of this paper, any reparametrization solution could be coupled to our method to solve this issue.

Despite the fact that we support segmenting the face into regions, our method is oblivious to other semantics at other levels of the details, thus it becomes hard to extract selected features such as wrinkles or facial hair. Finally, region masks need to be created manually, and a manner to define masks during blending could make the creation process more agile.

In a further step, we would look into blending the textures as well as the geometry. The challenge is that the texture details must be aligned and properly identified to avoid blurring the texture during blending. Furthermore, even if a new parametrization is considered as preprocessing in our case, it might be interesting to investigate more automatic solutions in order to allow incorporating models that are not produced by our pipeline, or even non-humanoid faces.

Another future step which we have already carried out an initial research is how exploring multidimensional spaces via inverse projections could help us in blending multiple faces. One approach that seems very promising is the iLAMP developed by Amorim et al, [62]. This technique performs inverse projection by local affine mappings that preserve the distance between new samples [63]. Thus, one may interactively create instances in the original database, producing synthetic multidimensional data besides the already existent in the original set. It would be interesting to compare the results of this approach and the one we have presented in this thesis. Amorin et al. [64] further proposed another work where one may have control points and, thus, to achieve more precision in handling multidimensional data. An extension of this work using radial basis function (RBF) was discussed in [65] and applied to human face modeling. Definitely, this is a work where we find inspiration and a good reference to our research. We believe that, in our case, the multidimensional space would be composed by the multiple face models and, using a 2D interface, similar to the one what we have presented in this thesis (see Figure 6.1), we could swipe the mouse over the faces and have the new model being designed based on the position of the mouse related to the multiple face samples.

Moreover, we believe that with further research we can tune the method for specific purposes. For example, to automatically produce a new set of original faces to create more diversity among the background characters. Alternatively, it should also be possible to produce faces that maintain some resemblance to produce, say, characters from the same ethnic background, or that might belong to the same family. Finally, the current method has been applied to faces, we can explore the same strategies to transfer details between different models such as other body parts or even non-animated objects.

Bibliography

[1] TIAN, L., LIU, J., GUO, W. “Three-Dimensional Face Reconstruction Using Multi-View-Based Bilinear Model”, *Sensors*, v. 19, pp. 459, 01 2019. doi: 10.3390/s19030459.

[2] ZHU, X., YI, D., LEI, Z., et al. “Robust 3D Morphable Model Fitting by Sparse SIFT Flow”. In: *2014 22nd International Conference on Pattern Recognition*, pp. 4044–4049, Aug 2014. doi: 10.1109/ICPR.2014.693.

[3] FYFFE, G., GRAHAM, P., TUNWATTANAPONG, B., et al. “Near-Instant Capture of High-Resolution Facial Geometry and Reflectance”, *Computer Graphics Forum*, 2016. ISSN: 1467-8659. doi: 10.1111/cgf.12837.

[4] MA, W.-C., BARBATI, M., LEWIS, J. P. “A Facial Composite Editor for Blendshape Characters”. In: *Proceedings of the Digital Production Symposium*, DigiPro ’12, pp. 21–26, New York, NY, USA, 2012. ACM. ISBN: 978-1-4503-1649-1. doi: 10.1145/2370919.2370923. Disponível em: <http://doi.acm.org/10.1145/2370919.2370923>.

[5] ROMEIRO, R., MARROQUIM, R., ESPERANCA, C., et al. “Forensic Facial Reconstruction Using Mesh Template Deformation with Detail Transfer over HRBF”. In: *Brazilian Symposium of Computer Graphic and Image Processing*, pp. 266–273, 08 2014. doi: 10.1109/SIBGRAPI.2014.25.

[6] LI, R., BLADIN, K., ZHAO, Y., et al. “Learning Formation of Physically-Based Face Attributes”. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3407–3416, 2020.

[7] YOON, S., LEWIS, J., RHEE, T. “Blending Face Details: Synthesizing a Face Using Multiscale Face Models”, *IEEE Computer Graphics and Applications*, v. 37, n. 6, pp. 65–75, November 2017. ISSN: 0272-1716. doi: 10.1109/MCG.2017.4031069.

[8] STATHAM, N. “Use of Photogrammetry in Video Games: A Historical Overview”, *Games and Culture*, v. 0, n. 0, pp. 1555412018786415, 2018.

doi: 10.1177/1555412018786415. Disponível em: <`https://doi.org/10.1177/1555412018786415`>.

[9] LEWIS, J. P., ANJYO, K., RHEE, T., et al. "Practice and Theory of Blendshape Facial Models". In: Lefebvre, S., Spagnuolo, M. (Eds.), *Eurographics 2014 - State of the Art Reports*. The Eurographics Association, 2014. doi: 10.2312/egst.20141042.

[10] ALKAWAZ, M. H., MOHAMAD, D., BASORI, A. H., et al. "Blend Shape Interpolation and FACS for Realistic Avatar", *3D Res.*, v. 6, n. 1, pp. 38:1–38:10, mar. 2015. ISSN: 2092-6731. doi: 10.1007/s13319-015-0038-7. Disponível em: <`http://dx.doi.org/10.1007/s13319-015-0038-7`>.

[11] PAWASKAR, C., MA, W., CARNEGIE, K., et al. "Expression transfer: A system to build 3D blend shapes for facial animation". In: *2013 28th International Conference on Image and Vision Computing New Zealand (IVCNZ 2013)*, pp. 154–159, Nov 2013. doi: 10.1109/IVCNZ.2013.6727008.

[12] D'APUZZO, N. "Modeling human faces with multi-image photogrammetry". In: *Three-Dimensional Image Capture and Applications V*, v. 4661, *Society of Photo-Optical Instrumentation Engineers (SPIE)*, 2002. doi: 10.1117/12.460168. Disponível em: <`https://doi.org/10.1117/12.460168`>.

[13] PIGHIN, F., HECKER, J., LISCHINSKI, D., et al. "Synthesizing Realistic Facial Expressions from Photographs". In: *ACM SIGGRAPH 2006 Courses*, SIGGRAPH '06, New York, NY, USA, 2006. ACM. ISBN: 1-59593-364-6. doi: 10.1145/1185657.1185859. Disponível em: <`http://doi.acm.org/10.1145/1185657.1185859`>.

[14] "Agisoft Metashape". `https://www.agisoft.com/`, 2019. Accessed: 2019-04-15.

[15] "Pixologic Zbrush". `http://pixologic.com/`, 2019. Accessed: 2019-04-09.

[16] "R3DS Wrap 3.3". `https://www.russian3dscanner.com/`, 2019. Accessed: 2019-04-09.

[17] BOUAZIZ, S., PAULY, M. "Dynamic 2D/3D Registration for the Kinect". In: *ACM SIGGRAPH 2013 Courses*, SIGGRAPH '13, New York, NY, USA, 2013. Association for Computing Machinery. ISBN: 9781450323390. doi: 10.1145/2504435.2504456. Disponível em: <`https://doi.org/10.1145/2504435.2504456`>.

[18] LEWIS, J. P., ANJYO, K., RHEE, T., et al. "Practice and Theory of Blendshape Facial Models". In: Lefebvre, S., Spagnuolo, M. (Eds.), *Eurographics 2014 - State of the Art Reports.* The Eurographics Association, 2014. doi: 10.2312/egst.20141042.

[19] BLANZ, V., VETTER, T. "A Morphable Model for the Synthesis of 3D Faces". In: *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '99, pp. 187–194, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co. ISBN: 0-201-48560-5. doi: 10.1145/311535.311556. Disponível em: <http://dx.doi.org/10.1145/311535.311556>.

[20] BLANZ, V., BASSO, C., POGGIO, T., et al. "Reanimating Faces in Images and Video", *Comput. Graph. Forum*, v. 22, pp. 641–650, 09 2003. doi: 10.1111/1467-8659.t01-1-00712.

[21] ZHANG, Y., SUNG, E., PRAKASH, E. "A physically-based model for real-time facial expression animation". In: *Proceedings Third International Conference on 3-D Digital Imaging and Modeling*, pp. 399–406, May 2001. doi: 10.1109/IM.2001.924487.

[22] TERZOPOULOS, D., WATERS, K. "Physically-based facial modelling, analysis, and animation", *Journal of Visualization and Computer Animation*, v. 1, pp. 73–80, 1990.

[23] BEELER, T., HAHN, F., BRADLEY, D., et al. "High-quality passive facial performance capture using anchor frames", *ACM Trans. Graph.*, v. 30, pp. 75:1–75:10, August 2011. ISSN: 0730-0301. doi: http://doi.acm.org/10.1145/2010324.1964970. Disponível em: <http://doi.acm.org/10.1145/2010324.1964970>.

[24] BEELER, T., BICKEL, B., BEARDSLEY, P., et al. "High-quality Single-shot Capture of Facial Geometry". In: *ACM SIGGRAPH 2010 Papers*, SIGGRAPH '10, pp. 40:1–40:9, New York, NY, USA, 2010. ACM. ISBN: 978-1-4503-0210-4. doi: 10.1145/1833349.1778777. Disponível em: <http://doi.acm.org/10.1145/1833349.1778777>.

[25] RHEE, T., HWANG, Y., KIM, J. D., et al. "Real-time Facial Animation from Live Video Tracking". In: Bargteil, A., van de Panne, M. (Eds.), *Eurographics/ ACM SIGGRAPH Symposium on Computer Animation.* The Eurographics Association, 2011. ISBN: 978-1-4503-0923-3. doi: 10.2312/SCA/SCA11/215-224.

[26] MA, W.-C., JONES, A., CHIANG, J.-Y., et al. "Facial Performance Synthesis using Deformation-Driven Polynomial Displacement Maps", *ACM Trans. Graph.*, v. 27, pp. 121, 12 2008. doi: 10.1145/1457515.1409074.

[27] EKMAN, P., ROSENBERG, E. L. *What the face reveals : basic and applied studies of spontaneous expression using the facial action coding systems.* 2nd ed ed. , New York ; Oxford : Oxford University Press, 2005. ISBN: 0195179641 (hbk.). Disponível em: <http://www.myilibrary.com?id=42846>. Previous ed.: 1998.

[28] FYFFE, G., NAGANO, K., HUYNH, L., et al. "Multi-View Stereo on Consistent Face Topology", *Computer Graphics Forum*, v. 36, pp. 295–309, 05 2017. doi: 10.1111/cgf.13127.

[29] LOWE, D. G. "Distinctive Image Features from Scale-Invariant Keypoints", *Int. J. Comput. Vision*, v. 60, n. 2, pp. 91–110, nov. 2004. ISSN: 0920-5691. doi: 10.1023/B:VISI.0000029664.99615.94. Disponível em: <https://doi.org/10.1023/B:VISI.0000029664.99615.94>.

[30] LIN, K., WANG, X., TAN, Y. "Self-adaptive morphable model based collaborative multi-view 3d face reconstruction in visual sensor network", *Multimedia Tools and Applications*, v. 75, n. 18, pp. 11469–11491, Sep 2016. doi: 10.1007/s11042-015-2864-2. Disponível em: <https://doi.org/10.1007/s11042-015-2864-2>.

[31] DAI, P., WANG, X., ZHANG, W. "Coarse-to-fine multiview 3d face reconstruction using multiple geometrical features", *Multimedia Tools and Applications*, v. 77, n. 1, pp. 939–966, Jan 2018. doi: 10.1007/s11042-016-4325-y. Disponível em: <https://doi.org/10.1007/s11042-016-4325-y>.

[32] BEELER, T., HAHN, F., BRADLEY, D., et al. "High-quality Passive Facial Performance Capture Using Anchor Frames". In: *ACM SIGGRAPH 2011 Papers*, SIGGRAPH '11, pp. 75:1–75:10, New York, NY, USA, 2011. ACM. ISBN: 978-1-4503-0943-1. doi: 10.1145/1964921.1964970. Disponível em: <http://doi.acm.org/10.1145/1964921.1964970>.

[33] KLAUDINY, M., HILTON, A. "High-fidelity Facial Performance Capture with Non-sequential Temporal Alignment". In: *Proceedings of the 3rd Symposium on Facial Analysis and Animation*, FAA '12, pp. 3:1–3:1, New York, NY, USA, 2012. ACM. ISBN: 978-1-4503-1793-1. doi: 10.1145/2491599.2491602. Disponível em: <http://doi.acm.org/10.1145/2491599.2491602>.

[34] CHEN, L., ZHENG, Y., CHEN, J., et al. "An Improved Laplacian Smoothing Approach for Surface Meshes". In: Shi, Y., van Albada, G. D., Dongarra, J., et al. (Eds.), *Computational Science – ICCS 2007*, pp. 318–325, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg. ISBN: 978-3-540-72584-8.

[35] VOLLMER, J., MENCL, R., MÜLLER, H. "Improved Laplacian Smoothing of Noisy Surface Meshes", *Comput. Graph. Forum*, v. 18, pp. 131–138, 1999.

[36] CIGNONI, P., MONTANI, C., SCOPIGNO, R. "A Comparison of Mesh Simplification Algorithms", *Computers and Graphics*, v. 22, pp. 37–54, 1997.

[37] FREITAG, L., JONES, M., PLASSMANN, P. "An Efficient Parallel Algorithm for Mesh Smoothing". In: *INTERNATIONAL MESHING ROUNDTABLE*, pp. 47–58, 1995.

[38] PARTHASARATHY, V. N., KODIYALAM, S. "A Constrained Optimization Approach to Finite Element Mesh Smoothing", *Finite Elem. Anal. Des.*, v. 9, n. 4, pp. 309–320, set. 1991. ISSN: 0168-874X. doi: 10.1016/0168-874X(91)90004-I. Disponível em: <`http://dx.doi.org/10.1016/0168-874X(91)90004-I`>.

[39] FIELD, D. A. "Laplacian smoothing and Delaunay triangulations", *Communications in Applied Numerical Methods*, v. 4, n. 6, pp. 709–712, 1988. doi: 10.1002/cnm.1630040603. Disponível em: <`https://onlinelibrary.wiley.com/doi/abs/10.1002/cnm.1630040603`>.

[40] VARTZIOTIS, D., ATHANASIADIS, T., GOUDAS, I., et al. "Mesh smoothing using the Geometric Element Transformation Method", *Computer Methods in Applied Mechanics and Engineering*, v. 197, pp. 3760–3767, 08 2008. doi: 10.1016/j.cma.2008.02.028.

[41] SHIMADA, K., YAMADA, A., ITOH, T. "Anisotropic Triangulation of Parametric Surfaces via Close Packing of Ellipsoids." *Int. J. Comput. Geometry Appl.*, v. 10, pp. 417–440, 08 2000. doi: 10.1142/S0218195900000243.

[42] CANANN, S. A., TRISTANO, J. R., STATEN, M. L. "An Approach to Combined Laplacian and Optimization-Based Smoothing for Triangular, Quadrilateral, and Quad-Dominant Meshes". In: *IMR*, 1998.

[43] A. FREITAG, L. "On Combining Laplacian And Optimization-Based Mesh Smoothing Techniques", *American Society of Mechanical Engineers, Applied Mechanics Division, AMD*, v. 220, 08 1999.

[44] R. BUELL, W., A. BUSH, B. "Mesh Generation — A Survey", *Journal of Engineering for Industry*, v. 95, pp. 332, 02 1973. doi: 10.1115/1.3438132.

[45] BIANCOLINI, M. "Mesh Morphing and Smoothing by Means of Radial Basis Functions (RBF): A Practical Example Using Fluent and RBF Morph". In: *Handbook of Research on Computational Science and Engineering: Theory and Practice*, pp. p. 347–380, Hershey, PA, USA, IGI Global, 01 2011. doi: 10.4018/978-1-61350-116-0.ch015.

[46] KNIPPING, J., DU TOIT, J., VUIK, C. *Comparison of Wavelets for Adaptive Mesh Refinement*. Technical Report. NAG, 2020.

[47] GROTH, C., CHIAPPA, A., BIANCOLINI, M. "Shape optimization using structural adjoint and RBF mesh morphing", *Procedia Structural Integrity*, v. 8, pp. 379–389, 01 2018. doi: 10.1016/j.prostr.2017.12.038.

[48] BOER, A., SCHOOT, M., BIJL, H. "Bijl, H.: Mesh deformation based on radial basis function interpolation. Comput. and Struct. 85, 784-795", *Computers & Structures*, v. 85, pp. 784–795, 06 2007. doi: 10.1016/j.compstruc.2007.01.013.

[49] BUHMANN, M. D. *Radial Basis Functions*. New York, NY, USA, Cambridge University Press, 2003. ISBN: 0521633389.

[50] SORKINE, O., COHEN-OR, D., LIPMAN, Y., et al. "Laplacian Surface Editing". In: *Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, SGP '04, pp. 175–184, New York, NY, USA, 2004. ACM. ISBN: 3-905673-13-4. doi: 10.1145/1057432.1057456. Disponível em: <`http://doi.acm.org/10.1145/1057432.1057456`>.

[51] SUMNER, R. W., POPOVIĆ, J. "Deformation Transfer for Triangle Meshes", *ACM Trans. Graph.*, v. 23, n. 3, pp. 399–405, ago. 2004. ISSN: 0730-0301. doi: 10.1145/1015706.1015736. Disponível em: <`http://doi.acm.org/10.1145/1015706.1015736`>.

[52] SHIN, I.-K., ÖZTIRELI, A. C., KIM, H.-J., et al. "Extraction and Transfer of Facial Expression Wrinkles for Facial Performance Enhancement". In: Keyser, J., Kim, Y. J., Wonka, P. (Eds.), *Pacific Graphics Short Papers*. The Eurographics Association, 2014. ISBN: 978-3-905674-73-6. doi: 10.2312/pgs.20141262.

[53] BOOTH, J., ROUSSOS, A., PONNIAH, A., et al. "Large Scale 3D Morphable Models", *Int. J. Comput. Vision*, v. 126, n. 2–4, pp. 233–254, abr. 2018. ISSN: 0920-5691. doi: 10.1007/s11263-017-1009-7.

[54] EGGER, B., SMITH, W. A. P., TEWARI, A., et al. “3D Morphable Face Models - Past, Present and Future”, *ACM Transactions on Graphics*, v. 39, n. 5, ago. 2020. doi: 10.1145/3395208.

[55] PLOUMPIS, S., WANG, H., PEARS, N. E., et al. “Combining 3D Morphable Models: A Large scale Face-and-Head Model”, *CoRR*, v. abs/1903.03785, 2019. Disponível em: <`http://arxiv.org/abs/1903.03785`>.

[56] GUO, Y., ZHANG, J., CAI, J., et al. “CNN-based Real-time Dense Face Reconstruction with Inverse-rendered Photo-realistic Face Images”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 41, n. 6, pp. 1294–1307, 2019.

[57] LEE, S., WOLBERG, G., SHIN, S. Y. “Scattered data interpolation with multilevel B-splines”, *IEEE Transactions on Visualization and Computer Graphics*, v. 3, n. 3, pp. 228–244, July 1997. ISSN: 1077-2626. doi: 10.1109/2945.620490.

[58] HANSEN, G., W. DOUGLASS, R., ZARDECKI, A. *Mesh Enhancement: Selected Elliptic Methods, Foundations And Applications.* Imperial College Press, 03 2005. ISBN: 978-1860944871. doi: 10.1142/P351.

[59] BJORCK, A. *Numerical Methods for Least Squares Problems.* Society for Industrial and Applied Mathematics, 1996. doi: 10.1137/1.9781611971484. Disponível em: <`https://epubs.siam.org/doi/abs/10.1137/1.9781611971484`>.

[60] WEISSTEIN, E. W. “Least Squares Fitting–Polynomial”. https://mathworld.wolfram.com/LeastSquaresFittingPolynomial.html, 2021. Accessed: 2021-12-15.

[61] ARLINGHAUS, S. L. *Practical handbook of curve fitting.* Boca Raton, Fla, CRC Press, 1994.

[62] DOS SANTOS AMORIM, E. P., BRAZIL, E. V., DANIELS, J., et al. “iLAMP: Exploring high-dimensional spacing through backward multidimensional projection”. In: *2012 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pp. 53–62, Oct 2012. doi: 10.1109/VAST.2012.6400489.

[63] JOIA, P., COIMBRA, D., CUMINATO, J. A., et al. “Local Affine Multidimensional Projection”, *IEEE Transactions on Visualization and Computer Graphics*, v. 17, n. 12, pp. 2563–2571, Dec 2011. ISSN: 1077-2626. doi: 10.1109/TVCG.2011.220.

[64] AMORIM, E., BRAZIL, E. V., NONATO, L. G., et al. "Multidimensional Projection with Radial Basis Function and Control Points Selection". In: *2014 IEEE Pacific Visualization Symposium*, pp. 209–216, March 2014. doi: 10.1109/PacificVis.2014.59.

[65] AMORIM, E., VITAL BRAZIL, E., MENA-CHALCO, J., et al. "Facing the High-dimensions: Inverse Projection with Radial Basis Functions". In: *Computers & Graphics*, v. 48, pp. 35–47, 03 2015. doi: 10.1016/j.cag.2015.02.009.