EPFL

Thèse n° 8818

Model-based 3D Face Reconstruction

Présentée le 17 décembre 2021

Faculté des sciences et techniques de l'ingénieur
Laboratoire de traitement des signaux 5
Programme doctoral en génie électrique

pour l'obtention du grade de Docteur ès Sciences

par

Christophe René Joseph ECABERT

Acceptée sur proposition du jury

Dr J.-M. Vesin, président du jury
Prof. J.-Ph. Thiran, directeur de thèse
Prof. H. Ekenel, rapporteur
Dr M. Sorci, rapporteur
Dr S. Marcel, rapporteur

École
polytechnique
fédérale
de Lausanne

2021

Persistence is very important. You should not give up unless you are forced to give up.
— Elon Musk

To my parents, and family...

Acknowledgements

This thesis is the final milestone of my journey at EPFL started quite a while ago. What should have been a short stay for getting my Master's degree has turned into an exciting adventure spread over the last ten years. These years have been filled with learning experiences, not only on the scientific aspect but also on the human side. I have met many amazing people and countless memories along the way, and I would like to take the opportunity to thank them here.

First and foremost, I would like to express my deepest gratitude to my Ph.D. advisor, Prof. Jean-Philippe Thiran. Without him, none of this would have been possible. Thank you for your trust, for believing in me, and for giving me the opportunity to start this thesis. I am grateful for all the support through the good and the difficult moments met along the way and for guiding me through the years to the successful completion of this work.

I am very thankful to the jury members who have excepted to be part of my thesis. Prof. Hazim Ekenel, Dr Sébastien Marcel, and Dr Matteo Sorci who took on the evaluation of the work carried out in this thesis. Their insightful comments and questions during the defense and their feedbacks on this manuscript were very valuable. It helped me improve the quality and clarity of my thesis. I want to thank the president of my jury, Dr. Jean-Marc Vesin, who made sure everything went smoothly. Honorable mention to Christian, Gabriel, and Laura, who spend invaluable time proofreading this entire thesis.

I have joined the Signal Processing Laboratory 5 (LTS5) family as a master student and started working on what has paved the way for this thesis. On my journey, I have met exceptional human beings that I would like to acknowledge, my former colleagues from the *Face group* - Anil, Damien, Gabriel, Hua, Marina, and Murat. Being the last member of this group, a chapter is closing. Of course, the lab is not limited to the Face group, and I would like to thank Adrien, Christian, Dimitri, Devavrat, Florien, Francesco, Guillaume, Johnatan, Rémi, Roser, Saeed, Saleh, Samuel, Sandra, Simon, Thomas, and those who I forgot. I have really appreciated sharing our research journeys, and you made the lab a great place to work in. Kudos to everyone!

I would also like to thank the people who made me balance my work over the past four years with eventful moments: André, Baptiste, Claude, Cyril, Gaëtan, Grégoire, Laura, Loïc, Pascal, Sébastien, Stéphanie, Victoria. Thank you very much for the priceless laughter and unforgettable memories.

And finally, I would like to thank my family, who unconditionally supported me over the past years. Without them, this work would not have been possible. My sister, my Mom, and Dad

who have always been there for me and encouraged me through all the events despite the fact that the specifics of my work remained a mystery. I could never stress enough how much it meant for me.
To everyone whose been part of this long journey, Merci du fond du coeur!

Lausanne, October 2021 Christophe Ecabert

Abstract

The human face plays an essential role in social interactions as it brings much information about someone's identity, state of mind, or mood. People are, by nature, very good at catching this non-spoken information. Therefore, scientists have been interested in replicating this skill to improve human-machine interactions over the last decades. This area of research is usually referred to as facial image analysis. It covers a wide range of domains such as face detection, facial landmarks localization, face recognition, or age and gender estimation, to name a few. Even though facial analysis algorithms are successfully used on a daily basis, for instance, to unlock our smartphones with face recognition, much remains to be done to make them more robust in a natural environment where head pose variations and illumination conditions can change drastically. Two decades ago, scientists started to show an increasing interest in algorithms driven by 3D models to overcome the intrinsic drawbacks of classical 2D algorithms. This transition led to techniques synthesizing 3D human faces out of images captured with conventional cameras. The reconstruction task is by nature an ill-posed problem due to the loss of information happening when cameras turn objects into images. After reconstruction, the virtual representation of a face gives access to additional information, such as distances, that would typically be lost with images.

Recent advances in deep learning have disrupted the 3D face reconstruction field unlocking new possibilities and research directions. The work carried out in this thesis identifies and focuses on different ways to increase the robustness of deep learning-based reconstruction systems and the fidelity of the synthesized faces. We first introduce a reference 3D reconstruction system composed of commonly used modules from the literature. The reference system is used to establish baseline results to compare against and assess the validity of the proposed methods.

We investigate ways to increase the robustness of the reconstruction system in the presence of significant head pose variations. We propose to modify the classical training strategy based on recent advances in contrastive learning to impose face parameterization consistency between different viewpoints of the same object. We validate our approach using two use-cases, with synthetic data and with real still images. The proposed method achieves similar performance to our baseline while being more consistent across a wide range of head poses.

The quality and resolution of images used while reconstructing 3D faces may not always be high; surveillance camera footage is a perfect example of this situation. Therefore, we investigate the possibility of learning visual representations of images regardless of the image resolutions to make the reconstruction systems robust and less sensitive to image resolutions.

Abstract

The proposed approach is able to perform similarly well across a wide range of image sizes. The last part aims at increasing the fidelity of the synthesized 3D face. Because of how the underlying statistical face models are built, the reconstructed facial geometry is very smooth and lacks depth cues linked to facial expressions, such as wrinkles. However, these cues play an important role in perceiving the correct facial expression. Therefore, we propose to recover this missing information and apply it on top of reconstructed geometry as corrections in the form of displacement maps. The proposed method is validated on a dataset of facial expression images and showed improvement over the standard approach.

Keywords: *Computer Vision, Machine Learning, Face Analysis, Face Reconstruction, 3D Morphable Model, Face Modeling, Inverse Rendering, Computer Graphics*

Résumé

Le visage humain joue un rôle essentiel dans les interactions sociales car il apporte de nombreuses informations sur l'identité, l'état d'esprit ou l'humeur d'une personne. Les gens sont, par nature, très doués pour capter ces informations non verbales. C'est pourquoi, au cours des dernières décennies, les scientifiques ont cherché à reproduire ce comportement afin d'améliorer les interactions homme-machine. Ce domaine de recherche est généralement appelé "analyse d'images faciales". Il couvre un large éventail de domaines tels que la détection et la reconnaissance de visages, la localisation de repères faciaux, ou l'estimation de l'âge et du sexe d'une personne, pour n'en citer que quelques-uns.

Bien que les algorithmes d'analyse de visages soient utilisés avec succès au quotidien, par exemple pour déverrouiller rapidement notre smartphone avec notre visage, il reste beaucoup à faire pour les rendre plus robustes dans un environnement naturel où le positionnement et l'orientation de la tête de l'utilisateur peuvent varier et les conditions d'éclairage peuvent changer. Il y a deux décennies, les scientifiques ont commencé à montrer un intérêt croissant pour les algorithmes se basant sur des modèles 3D afin de surmonter les inconvénients intrinsèques des algorithmes 2D classiques. Cette transition a conduit à des techniques de synthèse de visages humains en 3D à partir d'images capturées par des caméras conventionnelles. La tâche de reconstruction est par nature un problème mal posé en raison de la perte d'informations qui se produit lorsque les caméras transforment les objets en images. Après la reconstruction en 3D, la représentation virtuelle du visage donne accès à des informations supplémentaires, telles que les distances morphologiques, qui seraient typiquement perdues avec les images.

Les récentes avancées dans le domaine de l'apprentissage automatique ont bousculé le domaine de la reconstruction de visage en 3D débloquant de nouvelles possibilités et directions de recherche. Le travail effectué dans cette thèse identifie et se concentre sur différentes façons d'augmenter la robustesse des systèmes de reconstruction basés sur l'apprentissage automatique et la fidélité des visages synthétisés. Nous introduisons d'abord un système de reconstruction 3D de référence composé de modules couramment utilisés dans la littérature. Le système est utilisé pour établir des résultats de référence auxquels la validité des méthodes proposées sera comparée et évaluée.

Nous étudions les moyens d'accroître la robustesse du système de reconstruction en présence de variations importantes de la pose de la tête. Nous proposons de modifier la stratégie d'entraînement classique en nous appuyant sur les avancées récentes de l'apprentissage contrastif

pour imposer la cohérence du paramétrage du visage entre différents points de vue du même objet. Nous validons notre approche dans deux situations, avec des données synthétiques et avec des images réelles. La méthode proposée atteint des performances similaires à notre reférence tout en étant plus cohérente sur une large gamme de poses de la tête.
La qualité et la résolution des images utilisées lors de la reconstruction de visages 3D ne sont pas toujours élevées ; les images des caméras de surveillance sont un parfait exemple de cette situation. Nous étudions donc la possibilité d'apprendre des représentations visuelles d'images indépendamment de leur résolution, afin de rendre les systèmes de reconstruction robustes et moins sensibles aux résolutions d'images. L'approche proposée est capable d'obtenir des performances similaires sur une large gamme de tailles d'images.
La dernière partie vise à augmenter la fidélité du visage 3D synthétisé. En raison de la façon dont les modèles statistiques de visages sont construits, la géométrie faciale reconstruite est très lisse et manque d'indices de profondeur liés aux expressions faciales, comme les rides. Or, ces indices jouent un rôle important dans la perception de l'expression faciale correcte. Par conséquent, nous proposons de récupérer ces informations manquantes et de les appliquer sur la géométrie reconstruite comme corrections. La méthode proposée est validée sur des images de personnes exprimant une variété d'expression faciales et a montré une amélioration par rapport aux systèmes de reconstruction classiques.

Mots clés : *Vision par ordinateur, apprentissage automatique, analyse de visage, reconstruction de visage, modèle 3D déformable, modélisation de visage, rendu inverse, infographie.*

Contents

List of Figures

List of Tables

List of Acronyms

3DMM	3D Morphable Model.
AAM	Active Appearance Model.
BFM	Basel Face Model.
BiSeNet	Bilateral Segmentation Network.
BRDF	Bidirectional Reflectance Distribution Function.
CNN	Convolutional Neural Network.
EMA	Exponential Moving Average.
FACS	Facial Action Coding System.
FAN	Face Alignment Network.
GAN	Generative Adversarial Network.
GCN	Graph Convolutional Network.
GPMM	Gaussian Process Morphable Model.
GPU	Graphics Processing Unit.
HG	Hour-Glass.
ICP	Iterative Closest Point.
IoU	Intersection over Union.
KL	Kullback-Leibler.
MH	Metropolis–Hastings.
MLP	Multilayer Perceptron.
MSE	Mean Square Error.
NDC	Normalized Device Coordinate.
NICP	Nonrigid Iterative Closest Point.
NMR	Neural 3D Mesh Renderer.
PCA	Principal Component Analysis.
PDM	Point Distribution Model.
PRT	Precomputed Radiance Transfer.
PSF	Point Spread Function.
PSNR	Peak Signal-to-Noise Ratio.
PSPNet	Pyramid Scene Parsing Network.
S^3FD	Single Shot Scale-Invariance Face Detector.
SDM	Supervised Descent Method.
SH	Spherical Harmonics.
SIFT	Scale Invariant Features Transform.

SSD	Single Shot Detector.
SSIM	Structural Similarity Index Measure.
SVD	Singular Value Decomposition.
VR	Virtual Reality.

Introduction

Context and Motivations

For several decades, facial image analysis has attracted a lot of interest from scientists world-wide and has been a very active field of research. Facial image analysis is the field in computer vision dedicated to investigating the information provided by the human face. We know from social interactions that the human face plays an essential role as it brings a lot of information about someone's identity, state of mind, or mood.

The methods developed in this research area cover a wide range of domains such as face detection, facial landmarks localization, facial expression analysis, face recognition, and virtual face synthesis to name a few. Many applications in different domains such as biometric, visual speech understanding, and smart human-computer interaction were developed based on algorithms developed for facial image analysis thanks to advances in machine learning and modeling techniques.

The performances of the state-of-the-art algorithms are almost perfect on frontal images. However, these methods have limitations inherent to images such as self-occlusion, sensitivity to head pose variations or change of illumination conditions. They suffer performance when applied to a natural scene. With the increasing availability of consumer-level RGB-D scanners (*e.g. colors and depth sensors*), scientists have shown a growing interest in algorithms driven by 3D models to overcome the intrinsic drawbacks of classical 2D algorithms. This transition led to the beginning of the 3D face reconstruction research field.

Over the last two decades, face reconstruction systems have become more robust, the fidelity of the generated objects is better, and the fitting time has been drastically lowered. However, there are still some limitations. The reconstruction problem is intrinsically ill-posed because of the loss of information during the 3D to 2D projection. Ideally, the reconstruction system should preserve the intrinsic facial information (*e.g. the identity and the expression of the person*) while being insensitive to external environmental elements.

This thesis focuses on reconstruction methods based on deep learning coupled with explicit statistical face models. With this setup, we have identified three limiting factors impacting the reconstruction process at different levels. More specifically, we have investigated ways to

reduce the impact of head pose variations, the robustness of reconstruction networks against a wide range of image sizes, and the possibilities to improve the fidelity of the synthesized face by recovering missing mid-level facial details.

Thesis Outline

The work in this thesis is organized in the following way:

Chapter 1: Background. This chapter provides an overview of the different algorithms required and used at various stages of a standard monocular face reconstruction pipeline. These methods range from detecting faces in any image and localizing precise facial landmarks, segmenting components of the human face, and techniques to register 3D surfaces together. The face detection and alignment methods are not used directly during the reconstruction process but instead to prepare and build the training dataset. The same goes for the segmentation algorithms. These methods are needed to extract valuable information required during the reconstruction process. Finally, the registration algorithms are needed either to build statistical face models beforehand or at evaluation time to align the 3D ground truth and the reconstructed surfaces.

Chapter 2: Monocular 3D Face Reconstruction. This chapter goes through the different components of a standard monocular 3D reconstruction pipeline and provides a comprehensive literature review for each of them. The commonly used objective functions and the various fitting strategies are reviewed as well. The chapter concludes with a discussion on the identified limiting factors present in current reconstruction pipelines. These factors are the starting point for our contributions.

Chapter 3: Experimental Setup. This chapter outlines the experimental setup used in this work and establishes baseline results. More specifically, it defines the types of statistical face and illumination models and the details of each component of the reconstruction network. Details on the various objective functions and datasets used through our experiments are also given for reproducibility purposes. It also introduces the different evaluation protocols used to assess the reconstructed facial geometry and appearance quality. Finally, this baseline is compared against other approaches from the literature.

Chapter 4: Cross-Pose Consistency. With the current approaches of 3D face reconstruction, there is no guarantee that the model predicts the same face parameterization across a wide range of head rotations. This chapter presents a new approach for learning a pose-invariant feature subspace based on recent advances in contrastive learning. The pairs of augmented images are generated on the fly using the generative face model following the idea of [Genova et al., 2018]. The validity of the proposed method is tested on synthetic data as well as natural image sequences. Finally, the process is compared against the baseline established in Chapter 3.

Chapter 5: Resolution-aware 3D Reconstruction. Generally, the monocular 3D face reconstruction problem is tackled by assuming the images are of relatively high resolutions. This chapter introduces a modified reconstruction network following the work of [Xu et al., 2021] to be robust against a wide range of image resolutions and still generate accurate reconstructions.

Chapter 6: Fine Facial Details Recovery. Because of the way standard statistical face models are built, most of the mid-level facial details such as the wrinkles are lost. However, this information is crucial to reconstruct realistic facial expressions. This chapter proposes to recover them using the concept of displacement maps the computer graphics industry commonly uses. The effectiveness of the proposed method is demonstrated qualitatively and quantitatively on a dataset of facial expressions.

Chapter 7: Conclusion. This chapter reviews the fundamental discoveries from the work carried out of this thesis and outlines future research directions.

Contributions

In the scope of this thesis, the main contributions are summarized below:

1. Provide a comprehensive literature review of all the components of a standard monocular 3D face reconstruction pipeline.
2. Design a new training scheme to learn pose-invariant feature subspace to increase the consistency of the predicted parameters across sequences of natural images.
3. Implement a resolution-aware reconstruction network compliant with a wide range of image sizes.
4. Propose a new approach to model and recover mid-level facial details through the use of displacement maps.

1 Background

The monocular 3D face reconstruction frameworks rely not only on the raw pixels of an image but also on higher semantical information to perform the reconstruction of human faces. In this chapter, the different methods used to access and extract this information will be presented and discussed in the subsequent sections.

The first step of the reconstruction pipeline is to identify relevant regions in the images where faces are located. Section 1.1 presents two methods for detecting faces in images. Once the position of the face is defined, morphological information can be extracted through the localization of fiducial points (*i.e. landmarks*). These landmarks will be used to align the 3D facial geometry with the image. Section 1.2 introduces two regression-based algorithms to localize these landmarks from an image. Section 1.3 discusses image semantic segmentation applied to facial images to provide structural knowledge of the human face. The segmentation masks will define regions on the face where objective functions can be evaluated meaningfully. These different algorithms are not directly used by the reconstruction pipelines but during the training data preparation.

Lastly, Section 1.4 covers and explains the different aspects of the process of aligning, rigidly or nonrigidly, 3D surfaces. It is an important process used in two crucial steps of the 3D Morphable Model (3DMM) framework: when building a statistical model to have the data with the same topological representation, at evaluation time to ensure the reconstructed surface and the ground truth data share the same coordinate system (*i.e. both surfaces are aligned*) to measure meaningful metrics

1.1 Face Detection

In many facial analysis systems, face detection is a crucial first step. It checks for the presence of faces in a given image as well as their locations and sizes. The possible locations or scales of the faces are not known beforehand. Therefore the detector needs to perform an exhaustive inspection on the whole image. A greedy approach is to use a sliding window to perform the

search. The window template, defined as having approximately the same size as the face to be detected, is slid successively over the whole image. The image patch covered by the template is cropped at each step and considered a possible face. Features are extracted from the cropped region and fed into a classifier trained to predict whether or not there is a face in the area covered by the window. Because the object scale is not known in advance, the operation is repeated multiple times on downsampled version of the original image patch. That multiscale approach allows searching for faces of different sizes with the same classifiers by keeping the dimensions of the feature space fixed.

Because the search space covers the whole image, the detection process is computationally expensive. A face detector needs to have a good balance between algorithm complexity and features descriptiveness power to reach real-time execution and have a high detection rate. Therefore a trade-off has to be made between accuracy and execution speed.

The information extracted by the detector is solely based on the texture of the image. However, the appearance of the face can drastically change under various conditions, such as large head pose, facial expressions, occlusions, or illumination changes, making the detection task challenging. To be robust to these changes, a face detector requires a substantial amount of positive and negative images (*i.e. patches of image with faces and no faces*) to learn from a representative subset of appearances. Moreover, having a face detector with high recall and precision is critical for a facial analysis pipeline. Indeed, the consequences of not finding an existing face hinder the performance of the whole pipeline.

In the subsequent parts of this section, the first real-time detection framework and probably the most well-known face detector will be presented: the Viola-Jones face detector [Viola and Jones, 2001]. Then a more up-to-date method, the Single Shot Scale-Invariance Face Detector (S^3FD) of Zhang *et al.*, based on recent advances in deep learning for object detection will be discussed [Zhang et al., 2017]. The detector is directly derived from the multi-box object detection framework of Liu *et al.* applied to faces [Liu et al., 2016]. Both methods are radically different in their design considerations. The Viola-Jones face detector is tailored for fast execution but is not very resilient against the appearance challenges mentioned earlier. The S^3FD, on the other hand, is computationally heavier but can detect faces with higher accuracy even in the situation where the facial appearance is largely degraded (*i.e. higher system capacity*).

A broader review of the traditional face detection methods and the available databases can be found in [Zafeiriou, 2015] and the survey of Minaee *et al.* will cover the rapid progress of deep learning-based detectors [Minaee et al., 2021].

1.1.1 Viola-Jones Face Detector

Detecting objects in real-time is a challenging task. The goal is not only to classify the object but also to localize it in the image space. It has to be done as quickly as possible. The work

of Viola and Jones, presented in [Viola and Jones, 2001, Viola and Jones, 2004], was the first framework to achieve real-time detection. It became very popular and is still in use today by many applications. The study is oriented around face detection, but the method is not limited to faces and can be used for any object.

The method relies on three main components acting at various stages in the detection pipeline to process images extremely rapidly and achieve a high detection rate. The first element is the type of features, similar to the Haar basis function, used to represent the face coupled with an intermediate image representation, the *integral image*, for efficient and fast computation. The second element is a method for constructing a classifier by selecting a few discriminative features using AdaBoost. The last component is using a cascade of classifiers which dramatically increases the detection speed by focusing only on promising regions of the image and discarding background patches very early on. Each component will be discussed in more detail in the following sections.

Haar-like Features

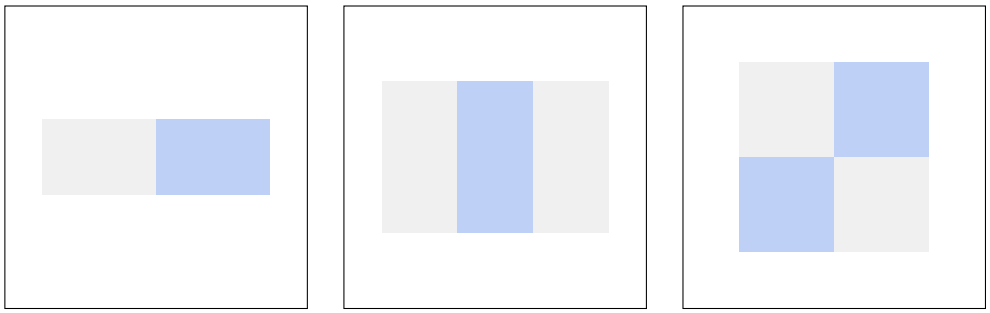
The Haar-like feature, used by the detector, is defined as the difference of the sums of every pixel within neighboring rectangles as illustrated in Figure 1.1. To effectively and rapidly compute rectangular features, an intermediate image representation called *integral image* is used. The integral image at location x, y contains the sum of the pixels above and to the left of x, y, as defined in Equation 1.1:

$$\boldsymbol{I}_{\mathrm{i}}\left(x,y\right)=\sum_{i\leq x,j\leq y}\boldsymbol{I}(i,j), \tag{1.1}$$

where $\boldsymbol{I}_{\mathrm{i}}$ is the integral image, $\boldsymbol{I}$ is the original grayscale image and $\left(x,y\right)$ are the coordinates on the image plane. Using this intermediate image representation, the features are efficiently evaluated. Figure 1.2 shows how the sum of all the pixels within any rectangle can be computed using an integral image by summing only the values at each corner of the rectangle. The computational footprint is thus very small and constant in time, making the computation of the Haar-like features extremely fast.

AdaBoost-based Feature Selection

With the Haar-like features and a set of positive and negative images, Viola and Jones used a variant of AdaBoost to select a small subset of discriminative features *and* train the classifier [Freund and Schapire, 1997]. The base resolution of the search window is 24x24 pixels giving an exhaustive set of features of more than 180000 combinations, making the set of rectangle features overcomplete (i.e. $180000 \gg 24^2$). Even though each feature can be computed efficiently, computing the whole set of features is still very expensive and time-consuming. Only a small number of these features can be combined to form an effective classifier. The



(a) Two rectangles (b) Three rectangles (c) Four rectangles

Figure 1.1 – Rectangle features shown within the area of detection. For every types, the sum of all the pixels in the gray rectangles is subtracted from the sum of all the pixels in the blue rectangles.

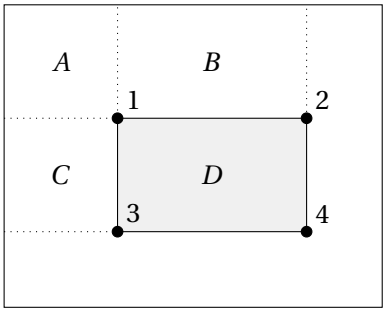


Figure 1.2 – The value the integral image at location 1 is the sum of all pixels within rectangle A. Similarly, the value at location 2 is the sum of all pixels in rectangle $A+B$, location 3 is for rectangle $A+C$ and finally location 4 is the sum of all rectangles $A+B+C+D$. The sum of all pixels within D can be computed as $\boldsymbol{I}_i(4)+\boldsymbol{I}_i(1)-\boldsymbol{I}_i(2)-\boldsymbol{I}_i(3)$.

challenge is then to find a limited number of discriminant rectangle features.

The feature selection process capitalizes on the idea of combining several *weak* classifiers to form a *strong* classifier. Moreover, the weak learning algorithm is designed to select a single rectangle feature that best separates the positive and negative examples. It performs the feature selection by forcing each weak classifier to work with a single rectangle feature. Only the most discriminative one will be retained. Therefore a weak classifier $h_j(x)$ is composed of a scalar feature ϕ_j, a threshold θ_j and a parity indicator p_j selecting the direction of the inequality in Equation 1.2:

$$h_j(x) = \begin{cases} 1 & \text{if} \quad p_j\phi_j < p_j\theta_j \\ 0 & \text{otherwise} \end{cases}, \tag{1.2}$$

where x is the region of the image from where the feature vector ϕ_j is extracted.

Cascade of Classifiers

The detector uses a cascade of classifiers to achieve an increased detection performance while maintaining low computational complexity. The main idea is to construct simpler classifiers to reject as many negative sub-windows as possible while detecting almost every positive instance, thus having a false negative rate close to zero. Once the majority of sub-windows have been rejected in the early stage of the cascade, stronger classifiers (*i.e. using more than one feature*) are used in the following stages to reach a low false-positive rate. At each stage in the cascade, the sub-windows identified as negative are directly discarded, thus avoiding extra computation. The ones classified as positive are forwarded to the next classifier in the chain.

To summarize the overall framework, the simple yet discriminative features coupled with weak classifiers arranged in a cascade manner make the face detector of Viola and Jones quite fast with decent performances. However, their approach is not robust against radical changes in appearance due to large head pose, facial expressions, or occlusions. The following section will introduce a deep learning-based method that tackles this issue at the cost of computational efficiency.

1.1.2 Single Shot Scale-invariant Face Detector

With the rise of deep learning over the recent years, Liu *et al.* presented a framework for real-time object detections that achieves high accuracy detection rates, namely the Single Shot Detector (SSD) [Liu et al., 2016]. The proposed scheme reaches competitive accuracy compared to methods that utilize an additional object proposal step, such as the Faster R-CNN method of [Ren et al., 2015]. The SSD is much faster while providing a unified framework for training and inference.

Following the SSD architecture, Zhang *et al.* presented the S^3FD, a detector tailored for detecting small and large faces [Zhang et al., 2017]. There are three main contributions in their work. The first element is a scale-equitable framework to deal with faces of various scales. The second element is a new matching strategy to combine predicted anchors and the true object bounding box. The last factor is a new mechanism to label background pixels to reduce the false positive detection rate. Each component will be discussed in more detail in the following sections.

Scale-equitable Face Detection Framework

The SSD framework uses a feed-forward convolutional network to produce multi-scale feature maps. These maps are in turn used to regressed fixed-size bounding boxes (*i.e. anchors*) and scores for the presence or not of an object instance within the bounds of these boxes. A non-maximum suppression step follows to produce the final detections. The main drawback of the method is the performance drop as the face becomes smaller [Huang et al., 2017]. The two main reasons for the performance reduction are the spatial positioning of the anchors

that are too coarse with not enough discriminative features available and the scales that are not adequate for small faces.

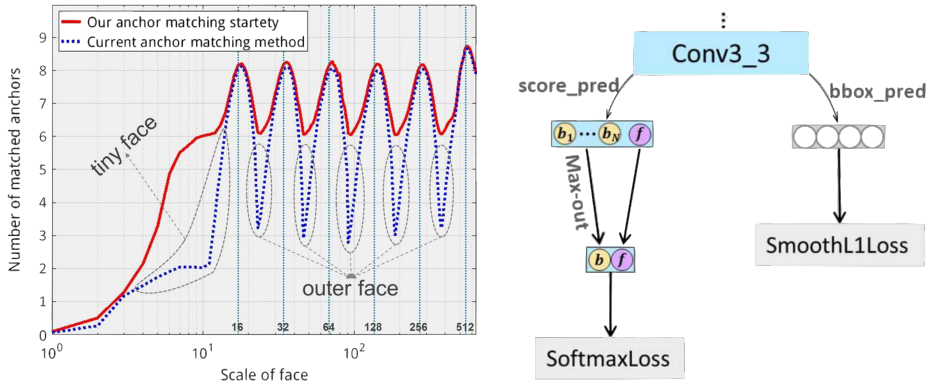
A rework of the architecture is proposed to solve these issues. The network has a wide range of detection layers that predict the bounding boxes and scores, whose stride size gradually doubles from 4 to 128 pixels. This ensures that different scales of faces have adequate features for detection. Once the locations of anchors are defined, the scales of anchors are selected based on the size of the effective receptive field of the filters. As pointed out in [Luo et al., 2016], a unit in a Convolutional Neural Network (CNN) has two types of receptive fields. The *theoretical receptive field* indicates the input region that can be affected by the value of the unit (*i.e. the filter*). However, not every pixel in this region contributes equally to the final output. Generally, the pixels located at the center have a much larger impact than those located near the outer boundary. Therefore only a fraction of the area has an effective influence on the output value, which is another type of receptive field named the *effective receptive field*. Based on this observation, the anchors are selected to match the effective receptive fields of the anchor-associated layers. The scales range from 16 to 512 pixels with an aspect ratio of $1:1$ since the bounding box of the face is approximately square.

The scale-equitable framework solves for the decreasing performance as the objects become smaller. Using a wide range of anchor-associated layers and a series of reasonable anchors scales, the detector can handle a wide span of faces scales.

Scale Compensation Anchor Matching Strategy

Every cell in the feature maps will produce a fixed set of object's bounding boxes. During training, correspondences between these anchors and the true face bounding box (*i.e. ground truth label*) need to be defined. However, anchor scales are discrete, while face scales are continuous. As shown by the blue dashed curve in the Figure 1.3a, the faces with scales distributed away from the anchor scales will not be matched with enough anchors or even be ignored, leading to a low recall rate. The issue is solved by using a scale-compensated matching strategy. The matching step is decomposed into two stages. The first stage consists of matching anchors and bounding boxes with a Jaccard overlap higher than 0.35 instead of the traditional 0.5 threshold. Lowering the decision threshold increases the average number of matched anchors. The second stage deals with faces that are not matched with enough anchors, such as tiny and outer faces. First, the matching is done by picking out anchors whose Jaccard overlap with the faces are higher than 0.1, then sorting them to select top-N as matched anchors. The value of N is set to the average number of matches from stage one.

The proposed matching strategy greatly increases the number of matched anchors of tiny and outer faces as shown by the red curve in Figure 1.3a and notably improves the recall rate for these faces.



(a) (b)

Figure 1.3 – (a) The number of matched anchors for different scales of faces are compared between traditional matching method and the scale compensation anchor matching strategy. (b) Max-out background label mechanism. Image from [Zhang et al., 2017]

Max-out Background Label

Anchor-based face detection methods can be regarded as binary classification problems determining if an anchor covers a face or background. Most of the pre-set anchors belong to the negative anchors (*i.e. background*), and only a few of them are positive anchors (*i.e. face*). Therefore the binary classification task is highly imbalanced. Small anchors are densely tiled on the image to detect small faces, causing a sharp increase in the number of negative anchors. Therefore the smallest anchors contribute the most to the false positive detections. A more sophisticated approach is used on the lowest layer to handle complicated backgrounds from small anchors to address the issue. For the smallest anchors, N_m scores are predicted for background labels instead of a single one. The final score is then defined by selecting from the one with the highest score as illustrated in Figure 1.3b. It integrates some local optimal solutions into the S^{3}FD model and reduces the false positive rate for small faces.

The S^{3}FD has been used through all the experiments presented in this thesis because of its robustness to different face sizes and significant appearance changes. However more recent methods such as *BlazeFace* [Bazarevsky et al., 2019] or *RetinaFace* [Deng et al., 2020] could be used instead.

1.2 Face Alignment

For many facial analysis tools, having the position of the face in the image is not enough. Higher semantical knowledge about the face structure or appearance is required. One can retrieve such morphological information by localizing facial landmarks such as the eye corners, the tip of the nose, or the contour of the mouth. Over the years, numerous databases with different semantical landmarks and different numbers of facial points have been proposed such as the Multi-PIE [Gross et al., 2010], the XM2VTS [Messer et al., 2009], and the IBUG

[Sagonas, 2016]. However, the number and type of landmarks used will mainly depend on the task or application being targeted.

Over the recent years, regression-based landmarks detectors have become very prominent and well studied. Two of these methods will be discussed in the following sections: a traditional regression-based model using hand-crafted features and a deep learning-based model that directly predict the position of the landmarks from an image. Face alignment is still an ongoing field of research. A more up-to-date overview can be found in the survey of Gogić [Gogić et al., 2021].

1.2.1 Supervised Descent Method

The problem of facial landmarks localization can be solved through a nonlinear optimization method like many other problems in computer vision (*i.e. camera calibration, optical flow, structure from motion*). A robust way to solve them is by using second-order methods such as Gauss-Newton or Levenberg-Marquardt. However, these methods have two main drawbacks, the objective function might not be twice differentiable, and the Hessian might be large and not positive definite. To address these issues, Xiong and De La Torre presented the Supervised Descent Method (SDM) framework that alleviates the need to compute the Jacobian and the Hessian explicitly [Xiong and De la Torre, 2013]. They demonstrate that the proposed method can achieve state-of-the-art performance for the face alignment task.

Localizing a set of L landmarks, $\boldsymbol{s} = \left(x_1, y_1, ..., x_L, y_L\right)^\top$, in an image $\boldsymbol{I}$, can be defined by the following objective function

$$f\left(\boldsymbol{s}_0 + \Delta\boldsymbol{s}\right) = \left\|\Phi\left(\boldsymbol{I}, \boldsymbol{s}_0 + \Delta\boldsymbol{s}\right) - \phi_*\right\|_2^2, \tag{1.3}$$

where Φ is a shape-indexed feature extraction function and $\phi_* = \Phi\left(\boldsymbol{I}, \boldsymbol{s}_*\right)$ are the local descriptors extracted around the true manually labeled landmarks. From the training data, SDM learns a series of descent direction producing a sequence of shape updates, $\boldsymbol{s}_{k+1} = \boldsymbol{s}_k + \Delta\boldsymbol{s}_k$, mapping the initial shape $\boldsymbol{s}_0$ to the target shape $\boldsymbol{s}_*$.

Assuming the function Φ is twice differentiable, following Newton's method, a second-order Taylor expansion is applied to Equation 1.3 leading to

$$f\left(\boldsymbol{s}_0 + \Delta\boldsymbol{s}\right) \approx f\left(\boldsymbol{s}_0\right) + \boldsymbol{J}_f\left(\boldsymbol{s}_0\right)^\top \Delta\boldsymbol{s} + \frac{1}{2}\Delta\boldsymbol{s}^\top \boldsymbol{H}\left(\boldsymbol{s}_0\right)\Delta\boldsymbol{s}, \tag{1.4}$$

where $\boldsymbol{J}_f\left(\boldsymbol{s}_0\right)$ and $\boldsymbol{H}_f\left(\boldsymbol{s}_0\right)$ are the Jacobian and Hession matrices of f evaluated at $\boldsymbol{s}_0$. In order to simplify the notation, $\boldsymbol{s}_0$ will be dropped in the following development.

The optimal shape update $\Delta\boldsymbol{s}$ can be found be differentiating the Equation 1.4 with respect to $\Delta\boldsymbol{s}$ and setting it to zero. This gives the first update

$$\frac{\partial f(\boldsymbol{s}_0 + \Delta\boldsymbol{s})}{\partial \Delta\boldsymbol{s}} = \boldsymbol{J}_f + \boldsymbol{H}\Delta\boldsymbol{s} = 0 \tag{1.5}$$

$$\Leftrightarrow \Delta\boldsymbol{s} = -\boldsymbol{H}^{-1}\boldsymbol{J}_f = -2\boldsymbol{H}^{-1}\boldsymbol{J}_h^\top\left(\phi_0 - \phi_*\right), \tag{1.6}$$

where $\boldsymbol{J}_f = \boldsymbol{J}_h^\top\left(\phi_0 - \phi_*\right)$ is defined using the chain rule and $\phi_0 = \Phi(\boldsymbol{I}, \boldsymbol{s_0})$ is the features extracted with the initial shape $\boldsymbol{s}_0$. One can see the first Newton update as a projection of $\Delta\phi_0 = \phi_0 - \phi_*$ onto the row vectors of the matrix $\boldsymbol{R}_0 = -2\boldsymbol{H}^{-1}\boldsymbol{J}_f^\top$. In the remaining part of the section, $\boldsymbol{R}_0$ will be referred as a *descent direction.*

Defining the descent direction requires the function Φ to be twice differentiable or expensive numerical approximation of the Jacobian and Hessian. Furthermore, the update $\Delta\boldsymbol{s}$ is also function of the features ϕ_* extracted at the manually labelled landmarks which are not available at inference time (*i.e. fitting*). This limitations can be overcome by reformulating Equation 1.6 as a generic linear combination of feature vector ϕ_0 and a bias term $\boldsymbol{b}_0$ learned from the training data,

$$\Delta\boldsymbol{s}_1 = \boldsymbol{R}_0\phi_0 + \boldsymbol{b}_0. \tag{1.7}$$

The SDM approximates the first step of the optimization procedure of the alignment problem by learning a linear regressor using training samples. However, it is very unlikely that the algorithm converges in a single step since f is not necessarily quadratic under $\boldsymbol{s}$. Therefore SDM will generate a sequence of regressor $\{\boldsymbol{R}_k, \boldsymbol{b}_k\}$, each representing a different descent direction (*i.e. update step*).

The descent direction $\boldsymbol{R}_k$ and their biases $\boldsymbol{b}_k$ are learned from a set of face image $\{\boldsymbol{I}^i\}$ and the corresponding manually labeled landmarks $\{\boldsymbol{s}_*^i\}$ by minimizing the objective function

$$\underset{\boldsymbol{R}_k, \boldsymbol{b}_k}{\arg\min} \sum_{\boldsymbol{I}^i} \sum_{\boldsymbol{s}_k^i} \|\Delta\boldsymbol{s}_*^{ki} - \boldsymbol{R}_k\phi_k^i - \boldsymbol{b}_k\|_2^2, \tag{1.8}$$

where $\Delta\boldsymbol{s}_*^{ki}$ is the optimal update step to go from the current estimate of the shape to the annotated landmarks, and ϕ_k^i is the feature vector extracted at the position $\boldsymbol{s}_k^i$ updated by the previous linear regressor $\boldsymbol{R}_{k-1}$ and $\boldsymbol{b}_{k-1}$. Adding more regressors will decrease the error monotonically and refine the approximation. The feature descriptor Φ is generic and can be of any type. In the original work, they used Scale Invariant Features Transform (SIFT) features [Lowe, 2004]. A broad comparison of features, as well as improvement to increase the robustness of SDM, can be found in the work of Qu *et al.* [Qu et al., 2015].

Once the sequence of descent directions has been learned, it can be applied to any sample to localize the landmarks. In an iterative fashion, the shape update is regressed from the

feature vector $\Phi(\boldsymbol{I}, \boldsymbol{s}_{k-1})$ extracted at the current estimation of the position of the landmarks (*i.e. computed at the previous iteration*) as defined in Equation 1.9

$$\boldsymbol{s} = \boldsymbol{s}_0 + \sum_{k=1}^{N} \boldsymbol{R}_{k-1}\Phi(\boldsymbol{I}, \boldsymbol{s}_{k-1}) + \boldsymbol{b}_{k-1}. \tag{1.9}$$

The SDM framework tackles the face alignment problem by learning a series of linear regressors that map appearance-based feature vectors to landmark displacement, greatly simplifying the optimization process. However, for the method to work correctly in every possible case, it requires features to be robust enough to learn such mapping. Moreover, with the rise of datasets with more than a few thousand samples, the method will not necessarily scale in terms of generalization or robustness (*i.e. large head pose, self-occlusion,...*). The method presented in the next section, the Face Alignment Network (FAN), tackles these issues.

1.2.2 Face Alignment Network

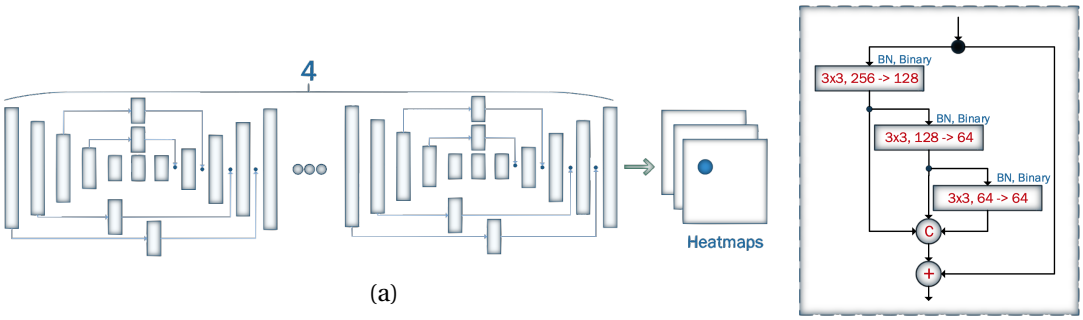
Cascaded regression method, such as SDM, is prone to performance deterioration in case of inaccurate initialization or when a significant number of landmarks are self-occluded due to large head poses. Therefore, Bulat and Tzimiropoulos proposed to mitigate these issues by building upon the recent advances in human pose estimation [Bulat and Tzimiropoulos, 2017b] and introduced the Face Alignment Network (FAN).

The landmarks are encoded into heatmaps consisting of 2D Gaussians centered on the position of the facial point (*i.e. mean value*) with a standard deviation of a few pixels. A single fiducial point is stored in a map. Thus, there are as many heatmaps as the number of landmarks. A convolutional neural network is then used to predict this new representation of the landmarks from a given image.

The proposed network architecture capitalizes on two components, the state-of-the-art Hour-Glass (HG) network from [Newell et al., 2016] and the hierarchical, parallel and multi-scale block of [Bulat and Tzimiropoulos, 2017a]. More specifically, the complete architecture is composed of a stack of four HG networks as shown in Figure 1.4a where each bottleneck block is replaced by a hierarchical block depicted in Figure 1.4b for increased performance while keeping the number of parameters constant.

The network is trained in an end-to-end manner with online data augmentation on the *300-W-LP* dataset of [Zhu et al., 2016]. This dataset is synthetically augmented with a face profiling technique to increase the head pose diversity in the data and avoid the dominance of frontal samples. The generalization power of the proposed architecture is investigated across multiple datasets, namely the *300-W* of [Sagonas, 2016], the *300-VW* of [Shen et al., 2015] and the *Menpo* challenge of [Zafeiriou et al., 2017].

Overall more than 220000 images are used to evaluate the proposed method. The outcome is



(a) (b)

Figure 1.4 – Detailed FAN architecture: (a) The regressor is made by a stack of four HGs networks with modified bottlneck block. (b) Internal structure of hierarchical, parallel and multi-scale bottleneck block, [Bulat and Tzimiropoulos, 2017b].

that FAN reaches near saturating performance on the datasets mentioned above and shows its robustness to various appearance changes. Therefore the FAN model has been selected for all the experiments in this thesis.

1.3 Image Semantic Segmentation

Semantic image segmentation is a computer vision task that consists of assigning a *category* (*i.e. label/class*) to every pixel of an image. Thus, it is harder than image classification, which predicts a single label for an entire group of pixels. The labeling at pixel-level turns groups of neighboring pixels with the same semantic into objects or parts of objects. It provides high-level information about the image content (*i.e. segmentation mask*). Figure 1.5 shows examples where components of the face are segmented. The segmentation algorithms are effective in a range of applications such as medical image analysis, autonomous vehicles, and facial segmentation.

In the scope of this thesis, an image segmentation model is used to extract specific semantic components of the face such as the nose, eyes, eyebrows, ears, mouth, lip, and more as depicted in Figure 1.5. These components highlight the semantic structure of the face and will be used to handle occlusions in the monocular reconstruction pipeline. More details on the usage of the segmentation maps will be given in Section 2.4.

The regions are generated with a Bilateral Segmentation Network (BiSeNet) proposed by Yu *et al.* designed for accuracy and speed [Yu et al., 2018]. The method is developed around two concepts, keep the rich spatial information down the network and receptive fields of various sizes to have enough context information. The next section will describe the overall architecture of the BiSeNet as well as the key design concepts.

Image semantic segmentation is a very active field of research. For an up-to-date survey of existing methods, refer to the survey of [Shijie Hao et al., 2020] for a broader review of the new

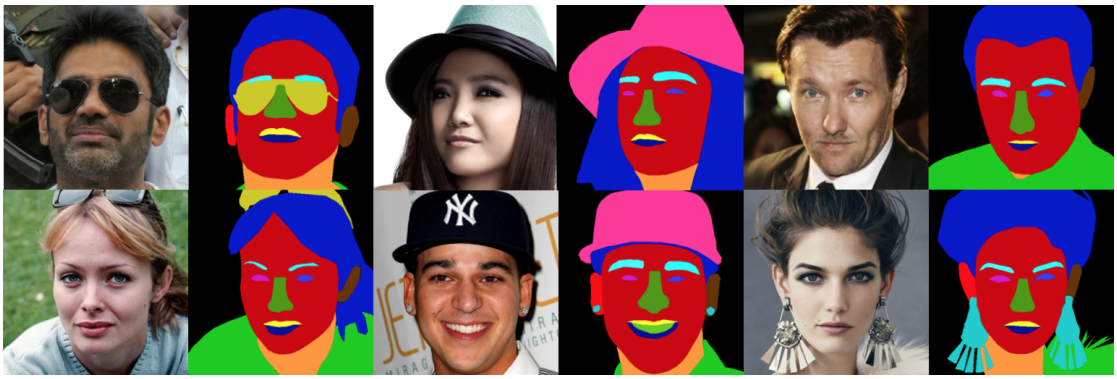


Figure 1.5 – Image samples and their corresponding segmentation masks from [Lee et al., 2020]

techniques.

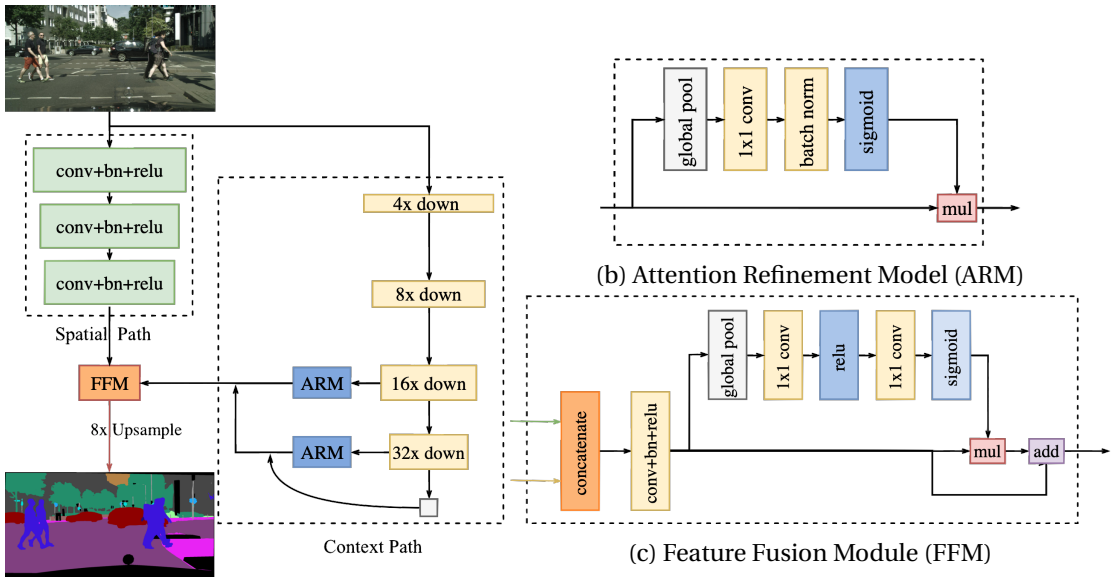
1.3.1 Bilateral Segmentation Network

The BiSeNet architecture divides the input signal into two distinct streams, the Spatial Path (SP) and the Context Path (CP). The former encodes positional information while the latter extracts contextual information. Both data signals are independent of each other and are aggregated together only in the last module of the network as depicted in the Figure 1.6a.

The Spatial Path purpose is to encode the affluent structural information while preserving the spatial size of the original input image. Its shallow architecture consists of three blocks composed of a convolutional layer with a stride of 2, batch normalization, and ReLU activation function. Therefore the spatial feature maps are 1/8 of the input dimensions and contain rich spatial information (*i.e. low compression rate*).

The Context Path is responsible for extracting contextual information to generate a high-quality segmentation mask. It capitalizes on a lightweight model, such as ResNet18 [He et al., 2016a] or Xception [Chollet, 2017], and global average pooling to create large receptive fields using fast downsampling structure to provide high-level semantic information. Moreover, adding a global average pooling layer at the end provides the largest receptive field and encodes a global context for the whole image. At each stage, the features are refined using an Attention Refinement Module (ARM). The module provides an attention vector to weight the importance of each feature map and guides the context learning process. The detailed structure of the module is shown in Figure 1.6b. Finally, the global and local contextual features are assembled in a multi-scale fashion.

The features encoded in both paths are semantically contrasting and at different levels of representation. The Spatial Path provides low-level features while the Context Path generates high-level features. Therefore, they must be carefully fused to retain the maximum information from both streams. The Feature Fusion Module (FFM) is responsible for aggregating the



(a) Network Architecture

(b) Attention Refinement Model (ARM)

(c) Feature Fusion Module (FFM)

Figure 1.6 – An overview of the Bilateral Segmentation Network. (a) Network Architecture. The width of the blocks indicates the spatial size and the height represents the number of channels in the feature map. (b) Internal structure of the Attention Refinement Module (ARM). (c) Internal structure of the Feature Fusion Module (FFM), [Yu et al., 2018].

knowledge from both feature representations. The features are first concatenated together then normalized with batch normalization. From the normalized features, a weight vector is computed. This weight vector defines the importance of each component and re-weights them accordingly, thus performing feature selection and combination. The whole process is illustrated in the Figure 1.6c.

Image semantic segmentation consists of assigning a class to each pixel in the picture. Therefore the classification model is trained with a softmax loss. The addition of auxiliary loss functions further supervises the output of the Context Path. The auxiliary tasks consist of predicting the segmentation masks from different levels of contextual features. For our use-case, the segmentation network has been trained on the dataset proposed by Lee *et al.*, namely the *CelebAMask-HQ* dataset [Lee et al., 2020].

1.4 Surface Registration

The process of surface registration or surface alignment consists of establishing dense correspondences between data points such as images, surfaces, or point clouds. Usually, registration is done between pairs of samples but can also be done on sequences of samples.

Given a source surface $\mathcal{S}$ and a target surface $\mathcal{T}$, registration will find a suitable deformation transformation $\boldsymbol{T}$ mapping points on $\mathcal{S}$ to points on $\mathcal{T}$ that have the same semantic. For

instance, when registering two faces, the nose tip of the source face should be mapped to the tip of the nose on the target face. Surface registration is required whenever a statistical shape model is built [Blanz and Vetter, 1999], when partial 3D scans are stitched together, or when comparisons between surfaces are needed.

The optimal deformation transformation $\boldsymbol{T}$ is estimated by minimizing the objective function defined in Equation 1.10,

$$E(\boldsymbol{T}) = \mathcal{D}(\boldsymbol{T}(\mathcal{S}), \mathcal{T}) + \lambda \mathcal{R}(\boldsymbol{T}), \tag{1.10}$$

where the data fidelity term $\mathcal{D}$ measures the distance between the two surfaces and the regularization term $\mathcal{R}$ penalizes unlikely solutions using prior information about the possible deformations. So far, the deformation transformation $\boldsymbol{T}$ is generic and could be anything. However, it will generally fall into two categories, *rigid-body* transformation when all vertices of the surfaces undergo the same deformation (*i.e. all points move in the same way*) and *nonrigid* transformation when each vertex of the surface are allowed to move independently.

Several ways have been proposed to solve the optimization problem defined in Equation 1.10, usually in an iterative manner. The most famous being the Iterative Closest Point (ICP) algorithm proposed by Chen and Medioni in [Chen and Medioni, 1991] for *rigid* transformation and the Nonrigid Iterative Closest Point (NICP) variant of Amberg *et al.* for *nonrigid* transformation [Amberg et al., 2007].

In the following parts of this section, the concept behind the ICP algorithm for rigid transformation will be presented, and each component will be discussed. Then the extension of the framework for nonrigid deformation will be introduced.

For more information about the registration framework, either applied to surfaces or images alignment, please refer to the work of Chen *et al.* [Chen et al., 2019a].

1.4.1 Iterative Closest Point

Chen and Medioni have proposed the original formulation of the ICP algorithm in [Chen and Medioni, 1991] followed by Besl and McKay in [Besl and McKay, 1992]. Since then, their method has become the default choice for aligning three-dimensional surfaces. This popularity can be explained by the simplicity of the algorithm and the fact it relies solely on geometrical information. Despite its simple design, the method is yet very efficient and robust. The ICP technique has attracted lots of interest in the scientific community. Thus multiple variants have been proposed to improve the original design. The algorithm is composed of the five steps listed below:

- **Selection**: Select a subset of points from the source surface $\mathcal{S}$ to be used during the

alignment process.

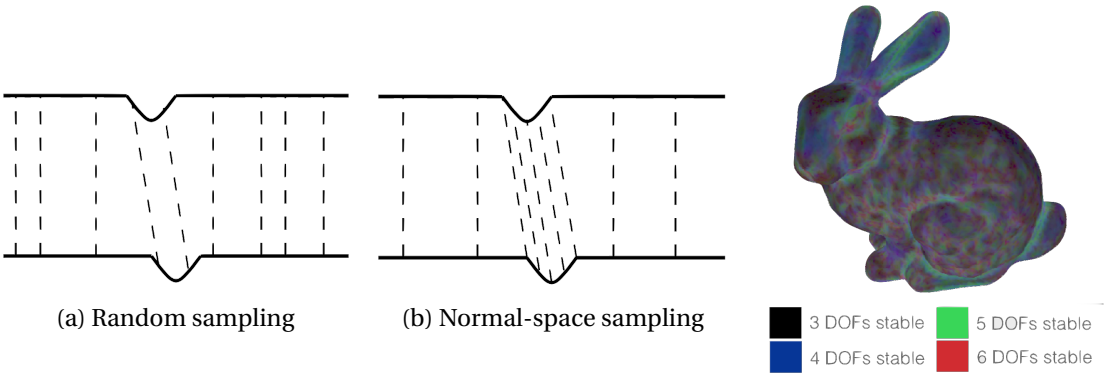
- **Matching**: Find correspondences between the subset of points on $\mathcal{S}$ and their equivalent on the target surface $\mathcal{T}$.
- **Weighting**: Assign weights to the matched pairs based on some properties defined on $\mathcal{S}$ and $\mathcal{T}$.
- **Rejecting**: Reject specific pairs based on individual metrics or by considering the entire set of pairs.
- **Error Metric and Minimization**: Define the error metric from the accepted pairs of points and minimize it.

These fives steps form the ICP algorithm. They are iteratively repeated until convergence is reached. Starting with an initial guess, the rigid-body transformation $\boldsymbol{T}$ will be refined at each iteration by minimizing an error metric with the estimated correspondence pairs. The initial estimation of the rigid transformation can be computed using various techniques such as matching features between the two surfaces [Stein and Medioni, 1992], calculating the principal axes of the scans [Dorai et al., 1998], or even provided by manual human annotations.

Selection The selection step consists of picking a subset of points from the source surface $\mathcal{S}$ to ensure all degrees of freedom of the transformation $\boldsymbol{T}$ will be constrained. Originally all the available points from $\mathcal{S}$ were selected [Besl and McKay, 1992]. This approach is suboptimal in the presence of noise and highly costly as the number of vertices increases (*i.e. linear complexity with the number of vertices*). To this end, alternate sampling schemes have been used. Turk and Levoy proposed to use a uniform sampling scheme, and Masuda *et al.* have used a random sampling strategy to select the points on the surface [Turk and Levoy, 1994, Masuda et al., 1996].

Nonetheless, Rusinkiewicz and Levoy proposed a new strategy, called *normal-space* sampling, based on the observation that for certain kinds of meshes, such as planar surfaces, small features or grooves are vital to find the correct alignment. The points are selected based on the orientation of the normals of the surface. The idea is to have a distribution of the normals of the selected points as large as possible [Rusinkiewicz and Levoy, 2001]. A random sampling strategy will not consider the topology of the surface. It will often select only a few samples in this area, as shown in Figure 1.7a, leading to an incorrect estimation of the component of the deformation transformation. The selection process builds a histogram of the normals of the points in angular space, then uniformly sample across all the bins. The result of sampling in the normal space is illustrated in Figure 1.7b.

Normal-space sampling ensures all the available orientations of the surface are used for the alignment. However, this selection strategy will only cope with translational uncertainties in the registration, leaving the rotational uncertainties present. The geometrically stable sampling method of Gelfand *et al.* aims at solving the issue [Gelfand et al., 2003]. The core idea



(a) Random sampling (b) Normal-space sampling (c) Stable sampling

Figure 1.7 – Illustration of different surface sampling strategies, [Rusinkiewicz and Levoy, 2001, Gelfand et al., 2003].

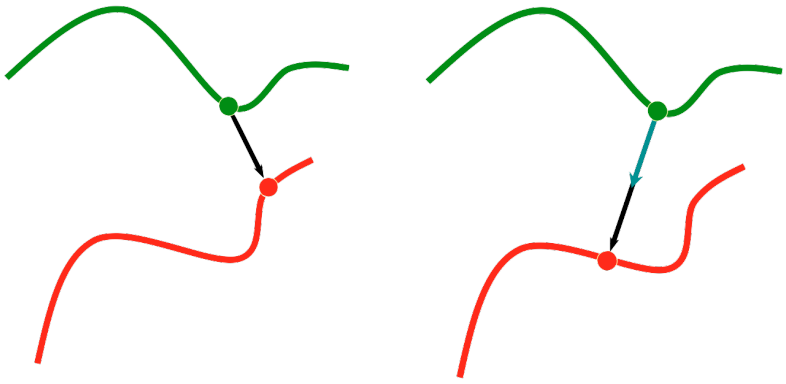
is to analyze the covariance matrix of the *torque* and *force* components contributed by each pair of points. The goal of the strategy is to select points such that the condition number of the covariance matrix is the closest to 1, such that each degree of freedom of the transformation is constrained. Figure 1.7c shows how each vertex of the mesh contributes to the stability of the registration, where constraints on the number of degrees of freedom are color-coded.

Matching Once some points have been picked on the source surface $\mathcal{S}$, correspondences on the target surface $\mathcal{T}$ need to be found for each one of them. Establishing such pairs is a vital step in the registration process, points with the same semantic need to be matched together to have proper alignment.

The naive approach to finding the correspondences for each selected points is by taking its closest point on the target surface $\mathcal{T}$ as done in [Besl and McKay, 1992]. The closest point matching principle is depicted in Figure 1.8a. An alternative method, named *normal shooting*, proposed by Chen and Medioni is to shoot a ray from the source vertex along the surface normal direction and find the closest point on the target near the intersection point as shown in Figure 1.8b [Chen and Medioni, 1991]. Usually, the normal shooting matching strategy will perform better than closest point matching for smooth surfaces. However, it is sensitive to noise and complex structures.

The two strategies mentioned above do not guarantee matched pairs are semantically correct correspondence, at least in the early iterations of the algorithm. However, after a few iterations, the transformation has been refined enough to bring semantically similar surface regions closed to each other. Therefore the semantic consistency of the matched pairs increases as the alignment process goes.

The semantic coherence between matched points can be increased by slightly modifying the closest point matching strategy. The idea is to match only the nearest point on the target surface *compatible* with the point on the source surface. The compatibility can be based on



(a) Closest point (b) Normal shooting

Figure 1.8 – Examples of different matching strategies used in ICP

normals, curvatures, or any local surface descriptor similarity.

All the strategies mentioned above for matching pairs of points are quite computationally expensive and need to be performed at every algorithm iteration. Moreover, their complexity scales linearly with the number of selected points. However, they can be accelerated by using spatial partitioning data structures such as *k-dimensional* trees [Bentley, 1975] or *octrees* [Meagher, 1982].

Weighting The purpose of weighting the pairs of correspondences established in the previous two steps is to define how each pair can be trusted for the registration. This is similar to a selection step with the benefit of being continuous. This step is not strictly necessary, and one can simply assign equal weight to each of the pairs (*i.e. each pair equally contributes to the registration*). However, other weighting schemes have been proposed over the years. One approach by Godin *et al.* is to assign lower weights to pairs with a large distance between both points [Godin et al., 1994]. More formally, the scheme defines weight proportional to the distance as defined in Equation 1.11:

$$w_i = 1 - \frac{\delta\left(x_i^{\mathcal{S}}, x_i^{\mathcal{T}}\right)}{d_{max}}, \tag{1.11}$$

where δ measures the distance between two points formed by a given pair, $x_i^{\mathcal{S}}, x_i^{\mathcal{T}}$ are components of the i^{th} pair of points, and d_{max} is the largest distance among all the matches.

An alternate approach is to reuse the compatibility concept introduced in the previous step. The weight for a given pair can be defined from the compatibility of the normals between the source and target points as defined in Equation 1.12:

$$w_i = n_i^{\mathcal{S}} \cdot n_i^{\mathcal{T}}, \tag{1.12}$$

where $n_i^{\mathcal{S}}$ and $n_i^{\mathcal{T}}$ are surface normals.

Rusinkiewicz and Levoy have analyzed the impact on the convergence rate of various weighting schemes. They conclude that the effect is usually small and highly dependent on the kind of data being registered. Thus the weighting step is most of the time skipped due to its negligible impact on the final alignment accuracy [Rusinkiewicz and Levoy, 2001].

Rejecting The purpose of rejecting pairs of correspondences is to eliminate outliers from the set of pairs used for registering the surfaces. Wrongly matched vertices could have a large effect while solving the alignment problem in the least-square sense. Therefore various rejection strategies are used to minimize their impact on the alignment.

A straightforward approach to filtering the outliers is rejecting pairs of points with a distance larger than a user-specified threshold. However, looking at each pair individually is not necessarily the most robust way to detect outliers. Thus Dorai *et al.* proposed to reject pairs that are not consistent with their neighboring pairs [Dorai et al., 1998]. Considering two matched pairs $\left(x_i^{\mathcal{S}}, x_i^{\mathcal{T}}\right)$ and $\left(x_j^{\mathcal{S}}, x_j^{\mathcal{T}}\right)$, it will be defined as inconsistent if and only if:

$$\left|\delta\left(x_i^{\mathcal{S}}, x_i^{\mathcal{T}}\right) - \delta\left(x_j^{\mathcal{S}}, x_j^{\mathcal{T}}\right)\right| > \epsilon, \tag{1.13}$$

where $\delta(\cdot,\cdot)$ measures the distance between two given points and ϵ is a user-defined distance threshold. The main drawback of this strategy is its computational cost. It has a running time of $\mathcal{O}\left(n^2\right)$.

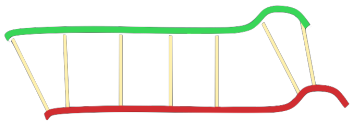
Alternatively, one can sort the matched pairs based on the distance between the points and reject the k% of the worst pairs as proposed by Chetverikov *et al.* in [Chetverikov et al., 2002]. The method is usually referred to as the *trimmed* ICP.

The main drawback of the rejection step is the need for some knowledge about the surface overlap to remove wrongly matched pairs of points properly. This information has to be estimated in some way but is usually user-supplied.

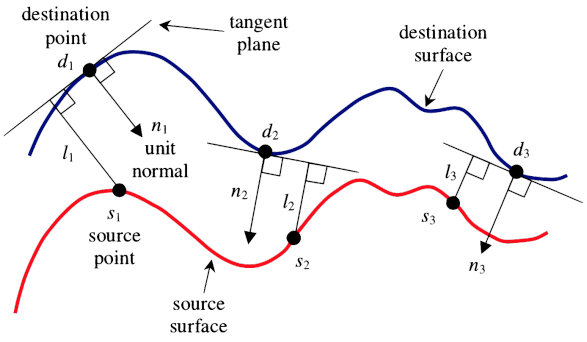
Error Metric and Minimization Once a set of valid pairs of points have been established, the final piece to estimate the rigid transformation $\boldsymbol{T} = (\boldsymbol{R}, \boldsymbol{t})$ is an objective function. One possible metric is the *point-to-point* distance defined in Equation 1.14:

$$E(\boldsymbol{R}, \boldsymbol{t}) = \sum_{i \in \mathcal{Q}} w_i \| (\boldsymbol{R}\boldsymbol{x}_i + \boldsymbol{t}) - \boldsymbol{y}_i \|_2^2, \tag{1.14}$$

where $\mathcal{Q}$ is a set of pairs of points, $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$ are points from the source surface $\mathcal{S}$ and target surface $\mathcal{T}$. The optimal rotational $\boldsymbol{R}$ and translational $\boldsymbol{t}$ components can be estimated by minimizing the objective function. The procedure listed in Algorithm 1 shows how to solve the optimization problem defined earlier using the Singular Value Decomposition (SVD) as



(a) *Point-to-point* distance between two surfaces



(b) *Point-to-plane* distance between two surfaces, [Low, 2004]

Figure 1.9 – Comparison between error metrics used in ICP

demonstrated by Arun *et al.* in [Arun et al., 1987].

Algorithm 1 ICP: Optimal point-to-point solution

1: $\bar{\boldsymbol{x}} = \frac{\sum_{i \in \mathcal{Q}} w_i \boldsymbol{x}_i}{\sum_{i \in \mathcal{Q}} w_i}, \quad \bar{\boldsymbol{y}} = \frac{\sum_{i \in \mathcal{Q}} w_i \boldsymbol{y}_i}{\sum_{i \in \mathcal{Q}} w_i}$ ▷ Compute each surface centroids

2: $\hat{\boldsymbol{x}}_i := \boldsymbol{x}_i - \bar{\boldsymbol{x}}, \quad \hat{\boldsymbol{y}}_i := \boldsymbol{y}_i - \bar{\boldsymbol{y}}$ ▷ Compute the centered vectors

3: $\boldsymbol{S} = \boldsymbol{X}\boldsymbol{W}\boldsymbol{Y}^\top$ ▷ Compute the 3×3 covariance matrix

4: $\boldsymbol{S} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$ ▷ Compute the singular value decomposition

5: $\boldsymbol{R} = \boldsymbol{V} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det\left(\boldsymbol{V}\boldsymbol{U}^\top\right) \end{bmatrix} \boldsymbol{U}^\top$ ▷ Compute the optimal rotation

6: $\boldsymbol{t} = \bar{\boldsymbol{y}} - \boldsymbol{R}\bar{\boldsymbol{y}}$ ▷ Compute the optimal translation

The major limiting factor of the point-to-point distance is that it does not allow points located on a flat region to slide over each other to reach the correct transformation. Consider the situation depicted in Figure 1.9a. Once the green surface has landed on red after a couple of iterations, the point-to-point metric will prevent it from moving in the horizontal direction. Therefore the transformation estimation will be wrong and the surface alignment incorrect.

To fix this issue, instead of measuring the distance between two points, the distance between one point and the tangent plane, perpendicular to the normal, located at the other point will be used (*i.e. on the target surface*). This is the *point-to-plane* distance, shown in Figure 1.9b, and it can be computed using Equation 1.15:

$$E(\boldsymbol{R}, \boldsymbol{t}) = \sum_{i \in \mathcal{Q}} \left(\left((\boldsymbol{R}\boldsymbol{x}_i + \boldsymbol{t}) - \boldsymbol{y}_i \right) \cdot \boldsymbol{n}_i \right)^2, \tag{1.15}$$

where $\boldsymbol{n}_i$ is the normal of the i^{th} vertex on the target surface $\mathcal{T}$.

The optimal solution for $\boldsymbol{R}$ and $\boldsymbol{t}$ can be obtained using any generic non-linear solver (*i.e.*

Figure 1.10 – Surface template, mesh acquired with scanner and the registration result, [Amberg et al., 2007]

Levenberg-Marquardt), which could potentially be costly. First, an alternative way is to linearize the rotation matrix using the small-angle approximation, $\sin\theta \approx \theta$ and $\cos\theta \approx 1$. Then rotation and translation can be determined using closed-form solution as shown in [Lowe, 2004].

The five steps of the ICP algorithm have been discussed in detail, with multiple solutions for each one of them. There is not a single variant of ICP that will work every time, and the choice of each internal component will most of the time depend on the data. To assist in this selection process, Pomerleau *et al.* have conducted a study aiming at comparing ICP variants [Pomerleau et al., 2013].

1.4.2 Nonrigid Iterative Closest Point

The surfaces acquired with a traditional 3D scanner will usually contain noise and missing parts (*i.e. holes*). The impact of such perturbations is significantly reduced by the alignment process and makes the scans usable. Nonrigid registration can be seen as the process of warping a template onto a target surface or a scan of a similar shape. The benefits are three folds. First, it smoothes out the surface. Second, the missing data are interpolated by using prior knowledge from the template (*i.e. structure*). Last, the mesh topology of the template is transferred to the target scan. Figure 1.10 shows an example of a scan before registration and once the alignment process is complete.

To find the correct deformation field that maps the template to the target surface, Amberg *et al.* proposed to extend the ICP framework to nonrigid registration while keeping the convergence properties of the original formulation [Amberg et al., 2007].

The template mesh that will be deformed is composed of a set of n vertices and a set of m edges, $\mathcal{S} = (\mathcal{V}, \mathcal{E})$. The registration process aims at finding the deformation parameters $\boldsymbol{T}$ mapping the template to the target surface. Once the registration process is over, the aligned surface is projected onto the target surface along the normal of the deformed template to generate the final reparametrization of the original scan.

For nonrigid deformation, every vertex on the template surface can freely move around. The parametrization of the overall transformation is defined by one affine transformation matrix

$\boldsymbol{T}_i$ per-vertex on the template surface. Each affine transformation is stacked in a $4n \times 3$ matrix

$$\boldsymbol{T} := [\boldsymbol{T}_1, \ldots, \boldsymbol{T}_n]^\top . \tag{1.16}$$

The distance between the deformed template and the target should be small and is measured by,

$$E_d(\boldsymbol{T}) = \sum_{\boldsymbol{v}_i \in \mathcal{V}} w_i \delta(\mathcal{T}, \boldsymbol{T}_i \boldsymbol{v}_i)^2 , \tag{1.17}$$

where $\boldsymbol{v}_i = \left[x, y, z, 1\right]$ is a template vertex in homogenous coordinate, δ measures the distance to the closest point on the target, and $w_i \in \{0, 1\}$ indicates if a matching point has been found on the target surface.

In addition, a stiffness term is used to regularize the deformation and estimate a correct deformation field. The transformation of neighboring vertices should be close to having a smooth transformation across the entire template. The regularizer is defined as

$$E_s(\boldsymbol{T}) = \sum_{\{i,j\} \in \mathcal{E}} \left|\left|\left(\boldsymbol{T}_i - \boldsymbol{T}_j\right) \boldsymbol{G}\right|\right|_F^2 , \tag{1.18}$$

where i, j are indexes of neighboring vertices connected by an edge, $G = \mathrm{diag}\left(1, 1, 1, \gamma\right)$ is a weighting matrix balancing the differences in the rotational/skew parts versus the translational part.

The complete objective function is given in

$$E(\boldsymbol{T}) = E_d(\boldsymbol{T}) + \alpha E_s(\boldsymbol{T}) , \tag{1.19}$$

where the stiffness factor α controls how much the template is allowed to move. The complete registration process is listed in Algorithm 2. In the beginning, the deformation is strongly regularized (*i.e. large α factor*) to recover global deformation. As the number of iterations increases, the regularization is lowered, allowing for more localized deformations.

Once the correspondences are established and fixed, the objective function defined in Equation 1.19 becomes a sparse quadratic system that can be solved exactly and efficiently. The resulting quadratic function can be minimized directly using the closed-form solution. Note that, given a set of correspondences, the optimal transformation is estimated at every step.

Algorithm 2 Nonrigid optimal step ICP

1: Initialize $\boldsymbol{T}_0$
2: **for each** $\alpha^i \in \{\alpha^1, \ldots, \alpha^n\}, \alpha^i > \alpha^{i+1}$ **do**
3: **while** $\left|\left|\boldsymbol{T}^j - \boldsymbol{T}^{j-1}\right|\right| > \epsilon$ **do**
4: Find correspondences $\mathcal{Q}$
5: Find the optimal deformation $\boldsymbol{T}^j$ by solving Equation 1.19
6: **end while**
7: **end for each**
8: Project the deformed mesh $\boldsymbol{T}(\mathcal{V})$ onto the target surface

1.5 Summary

In this chapter, multiple tools required by various components and at different stages of the reconstruction process have been presented and reviewed. In Section 1.1, ways to find the location of the faces in the image have been presented. Given a detection bounding box of the face, Section 1.2 introduced two regression-based methods to find the position of facial anatomical points needed by the reconstruction pipeline to perform the initial alignment. Section 1.3 introduced the BiSeNet method to extract higher-level semantical information of the face at the pixel level. Such information will play a crucial role in the reconstruction pipeline. More specifically, the S^3FD, the FAN, and BiSeNet methods are used to automatically extract the required information to prepare the training data. The outputs of these state-of-the-art algorithms are assumed to be correct and are used as ground truth later in the fitting process. However, experiments have shown that this assumption does not hold true for in-the-wild data, which is a major flaw in the process. These algorithms produce a certain level of noise in their predictions, which can not be neglected. This noise affects the reconstruction task and has to remain low to limit its impact.

The last aspect discussed in Section 1.4 of this chapter concerns the ways to align 3D surfaces together. The discussion covered the well-known ICP algorithm first used for rigid surface alignment then extended to nonrigid deformation. In the context of 3DMM, this algorithm is used in two phases. The first use is when building the statistical shape model, where a mesh template needs to be non-rigidly aligned to the experimental data. The second usage is during evaluation time when the reconstructed surface needs to be rigidly aligned with the ground truth scan to measure meaningful metrics.

The next chapter will cover and review all the classical monocular 3D face reconstruction pipeline components, starting with modeling a complete 3D scene, followed by the different solutions to convert it to an actual image. The multiple objective functions used to constrain the reconstruction task, and the various fitting strategies will also be presented.

2 Monocular 3D Face Reconstruction

Monocular reconstruction consists of finding the 3D geometry and the color of an object that generates a given image and can be seen as inverting the image formation process. This topic has caught a lot of interest from the scientific community over the recent years and is still an active field of research. This is especially true with new emerging technologies such as Virtual Reality (VR), where 3D reconstruction can be used to simplify the process of creating VR assets and content.

Recovering a complete scene out of a single image is challenging. Thus different strategies have been proposed to solve the reconstruction problem. Multiview stereo vision, inspired by the human visual system, consists of using multiple images or *viewpoints* of the same scene to recover the 3D spatial location of each pixel using triangulation. However, this approach requires a complicated acquisition system and involves complex pixel matching algorithms prone to correspondence errors. Therefore alternate methods have been proposed using only a single camera as an acquisition system and the physical principles behind the image formation process. These methods fall in the Shape-From-X category, where X can be any 2D attribute such as shading, silhouette, or shadow.

Following the general idea of the Shape-From-X approach, prior knowledge on the geometry and the appearance of the object or illumination of the scene can be injected into the reconstruction problem using explicit models of these quantities. This is the approach taken in monocular 3D face reconstruction pipelines where the human face and the external factors are explicitly modeled. Thus, the complex scene can be represented with a small set of coefficients (*i.e. face, light, and camera parameters*). The reconstruction algorithm is responsible for finding the optimal set of parameters used to create the target image. The classical strategy to recover these parameters relies on the analysis-by-synthesis approach commonly used with a generative model. The general idea is to generate an instance of the model with the current estimation of the parameters and compare it against the target image. The error is then back-propagated, and the parameters are updated and refined accordingly. This process of synthesis, followed by an analysis step, is repeated until convergence.

This chapter will cover each component of the reconstruction pipeline in the subsequent sections. First, the different types of models used for the human face and the various ways to model external illumination will be presented. Then, the possible methods to convert a 3D scene into an image to compare it with the target image will be discussed. An overview of the commonly used objective function in the context of 3DMM fitting and the different strategies used to estimate the set of parameters of the generative scene model will follow. In this work, a generative scene model refers to a model that creates a textured instance of the face with illumination and places it in front of the camera with a given pose. The last section of this chapter reviews the different aspects of the whole reconstruction pipeline and identifies some of the limiting factors.

2.1 Morphable Face Model

The concept of 3DMM, a general face representation, was introduced more than 20 years ago. It is a generative model for facial appearance and geometry relying on two key components. The first component is the separating of the facial shape and texture and their disentanglement with respect to external factors such as illumination and camera parameters. The second component is the learning of the underlying distribution of all the possible variations a face can undergo [Egger et al., 2020].

The different types of modeling techniques will be reviewed in the following sections. First, the original formulation of the 3DMM will be presented, followed by a discussion on how it can be extended to bring prior variation information in the model without explicit examples. The different solutions to add facial expressions will be covered as well as more recent techniques to represent facial shape and appearance variations through the use of non-linear models based on deep learning. The last section will review the publicly available 3DMM models.

2.1.1 Parametric Face Model

In the original formulation by Blanz and Vetter, the 3DMM uses two linear subspaces to model the face shape and appearance [Blanz and Vetter, 1999] in a similar way as the *independent* Active Appearance Model (AAM) [Matthews and Baker, 2004]. The space of all possible shape deformation can be learned from a set of shapes $\{\Gamma_1, \ldots, \Gamma_n\}$ where each example is defined by a dense set of discrete vertices:

$$\Gamma_i = \{\boldsymbol{x}_k^i \mid \boldsymbol{x}_k \in \mathbb{R}^3, \quad k = 1, \ldots, N\}, \tag{2.1}$$

where N indicates the total number of vertices, usually in the range of thousands. All the samples are required to be in correspondence, meaning they have been registered beforehand (Section 1.4). Thus the k-th vertex x_k^i and x_k^j of the samples Γ_i and Γ_j represent the same anatomical point.

Given a single shape Γ_i, its vector representation $\boldsymbol{s}_i$ is defined by stacking each vertex $\boldsymbol{x}_k$ on top of each other: $\boldsymbol{s}_i = \left(x_1^i, y_1^i, z_1^i, \ldots, x_N^i, y_N^i, z_N^i\right)^\top$. The whole training data can then be defined in a matrix form as $\boldsymbol{s} = [\boldsymbol{s}_1, \ldots, \boldsymbol{s}_n]$. Adopting this new data representation, multivariate statistics are used to model a probability distribution over the shapes.

Shape variations are often assumed to follow a normal distribution $\boldsymbol{s} \sim \mathcal{N}\left(\mu, \Sigma\right)$. Thus the mean μ and the covariance matrix Σ can be easily estimated from the sample data as defined in:

$$\mu = \bar{\boldsymbol{s}} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{s}_i, \qquad \Sigma = \boldsymbol{S} = \frac{1}{n-1}\sum_{i=1}^{n} \left(\boldsymbol{s}_i - \bar{\boldsymbol{s}}\right)\left(\boldsymbol{s}_i - \bar{\boldsymbol{s}}\right)^\top. \tag{2.2}$$

Usually, the covariance matrix $\Sigma \in \mathbb{R}^{3N\times 3N}$ can not be explicitly computed due to memory limitation because of N being very large for a dense surface. However, by performing Principal Component Analysis (PCA), it can be represented using at most n basis vectors. Thus the shape model, commonly referred to as Point Distribution Model (PDM), is defined as:

$$\boldsymbol{s}\left(\boldsymbol{w}^s\right) = \bar{\boldsymbol{s}} + \boldsymbol{U}\boldsymbol{w}^s = \bar{\boldsymbol{s}} + \sum_{i=1}^{n} \boldsymbol{u}_i \sigma_i w_i^s, \tag{2.3}$$

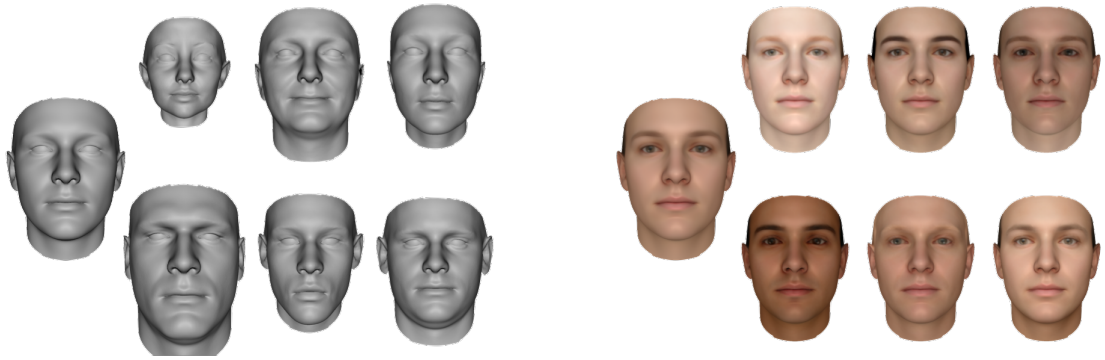
where the pair $\left(\boldsymbol{u}_i, \sigma_i^2\right)$ is the i-th eigenvector and eigenvalue of the covariance matrix $\boldsymbol{S}$ and w_i^s is a shape coefficient following a normal distribution $\mathcal{N}(0,1)$. The eigenvectors, usually referred to as *modes*, represent the directions in which the variance in the training data is maximum. Figure 2.1a shows the mean shape together with the first three modes of variations of the Basel Face Model (BFM), $\boldsymbol{u}_k,\ k \in \{1,2,3\}$, exhibiting different types of face morphology. This model has been proposed by Paysan *et al.* and has become one of the most used models over the years [Paysan et al., 2009].

Equation 2.3 forms a parametric face representation, where a shape is defined as a linear combination of modes. A new shape can be generated by sampling the shape coefficient $\boldsymbol{w}^s$ from a normal distribution.

As mentioned earlier, a 3DMM is also composed of a linear model for the texture of the face in the same fashion as the independent AAM. The same process used for modeling the shape variations is repeated to learn the possible facial appearance variations. The illumination-free appearance model is therefore defined in Equation 2.4:

$$\boldsymbol{a}\left(\boldsymbol{w}^t\right) = \bar{\boldsymbol{a}} + \boldsymbol{U}^t\boldsymbol{w}^t = \bar{\boldsymbol{a}} + \sum_{i=1}^{n} \boldsymbol{u}_i \sigma_i w_i^t, \tag{2.4}$$

where w_i^t is an appearance coefficient following a normal distribution $\mathcal{N}(0,1)$. Figure 2.1b shows the mean appearance together with the first three modes of variations, $\boldsymbol{u}_k,\ k \in \{1,2,3\}$,



(a) Shape modes (b) Reflectance modes

Figure 2.1 – The mean together with the first three principle components of the shape and reflectance PCA model. Shown is the mean shape resp. reflectance plus/minus five standard deviations σ, [Paysan et al., 2009]

illustrating different skin textures.

With both shape and appearance statistical models, any face can be approximated or generated. However, there are some intrinsic limitations to the method. As shown in Figure 2.1, only morphological changes linked to the person's identity are considered (*i.e. neutral expression*). Thus no facial expressions are part of the shape model. Moreover, only the deformations present in the training samples will be learned. No distribution extrapolation is possible with PDMs. In the model proposed by Paysan *et al.*, the BFM, both statistical models are learned on a total of 200 training samples (*i.e. 100 women and 100 men*) of mostly young caucasian people, introducing a population bias [Paysan et al., 2009].

Gaussian Process Morphable Model

Starting from Equation 2.3, Lüthi *et al.* proposed a different interpretation of the PDM. It is a model of deformations $\boldsymbol{\phi} = \sum_{i=1}^{n} \boldsymbol{u}_i \sigma_i w_i^s \sim \mathcal{N}(0, \boldsymbol{S})$ added to a mean shape $\bar{\boldsymbol{s}}$. The probability distribution is on the deformation rather than the overall shape, this generalized model is used to introduce the concept of Gaussian Process Morphable Model (GPMM) [Luthi et al., 2017]. The idea is to define a probabilistic model directly on the deformation using a Gaussian Process.

Given a reference shape $\Gamma_R \subset \mathbb{R}^3$ and a domain $\Omega \subset \mathbb{R}^3$ such that $\Gamma_R \subseteq \Omega$, a Gaussian process defined as $u \in \mathcal{GP}(\mu, k)$ with the mean function $\mu : \Omega \to \mathbb{R}^3$ and the covariance function $k : \Omega \times \Omega \to \mathbb{R}^{3\times 3}$. Any deformation $\hat{u}$ can be sampled from $\mathcal{GP}(\mu, k)$ and turned into a surface by warping the reference Γ_R as defined in Equation 2.5:

$$\Gamma = \{\boldsymbol{x} + \hat{u}(\boldsymbol{x}) \mid \boldsymbol{x} \in \Gamma_R\} \tag{2.5}$$

However, such representation is not practical but fortunately, a Gaussian process $\mathcal{GP}(\mu, k)$ can be represented in terms of an orthogonal set of basis functions $\{\phi_i\}_{i=1}^{\infty}$. If the variance of function k decreases quickly enough, a low-rank approximation of the process is enough:

$$u(\boldsymbol{x}) \sim \mu(\boldsymbol{x}) + \sum_{i=1}^{r} \phi_i(\boldsymbol{x}) \lambda_i \alpha_i, \quad \alpha_i \in \mathcal{N}(0,1), \tag{2.6}$$

where (λ_i^2, ϕ_i) are eigenvalue / eigenfunction pairs and r is the rank of the process. Equation 2.6 is known as the Karhunen-Loève expansion of a Gaussian process [Berlinet and Thomas-Agnan, 2004]. The outcome is a parametric model of finite dimension similar to standard PDM. However, no assumptions have been made on the covariance function k. Thus it can be any positive definite covariance function.

The main challenge is the estimation of the eigenvalue and eigenfunction pairs $(\lambda_i^2, \phi_i)_{i=1}^{r}$ in order to have a valid low-rank approximation. The standard numerical method for approximating them is the Nyström method proposed by Rasmussen and Williams [Rasmussen and Williams, 2006]. Later Liu and Matthies proposed an alternative approach based on Pivoted Cholesky Decomposition and Cross Approximation [Liu and Matthies, 2019].

The covariance function k or *kernel* must be positive definite and can be combined to create new kernels, leading to a powerful tool to model deformations. The simplest kernel fulfilling this requirement is the Gaussian kernel defined by:

$$k(\boldsymbol{x}, \boldsymbol{y}) = s \cdot \boldsymbol{I}_{3\times 3} \exp\left(-\|\boldsymbol{x} - \boldsymbol{y}\|^2 / \sigma^2\right), \tag{2.7}$$

where σ defines the range over which the deformations are correlated, the identity matrix $\boldsymbol{I}_{3\times 3}$ indicates that the components of the deformation field are independent, and s defines the scale of the deformation.

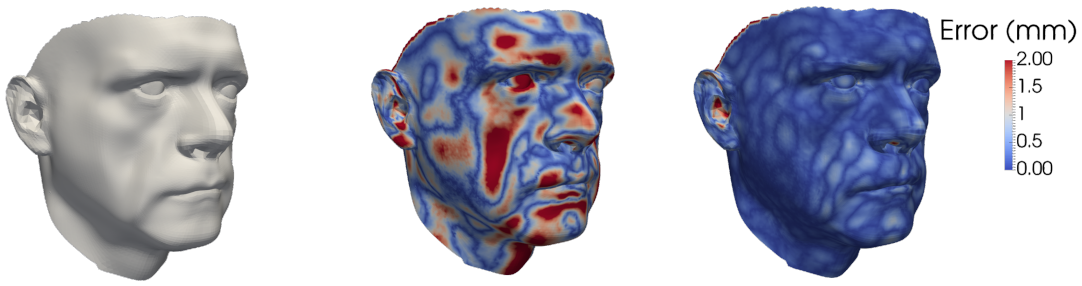
The standard PDM can also be formulated as a covariance function to include variability from experimental data. It is referred to as the *sampled covariance kernel*. It is given by Equation 2.9:

$$\mu_{PDM}(\boldsymbol{x}) = \frac{1}{n} \sum_{i=1}^{n} u_i(\boldsymbol{x}) \tag{2.8}$$

$$k_{PDM}(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{n-1} \sum_{i=1}^{n} \left(u_i(\boldsymbol{x}) - \mu_{PDM}(\boldsymbol{x})\right) \left(u_i(\boldsymbol{x}) - \mu_{PDM}(\boldsymbol{x})\right)^{\top} \tag{2.9}$$

where $u_i(\boldsymbol{x})$ is the deformation field that maps a point on the reference $\boldsymbol{x} \in \Gamma_R$ to its corresponding points on the i-th training sample.

By combining multiple kernels, Lüthi *et al.* showed how the GPMM framework can be used



(a) Target (b) Reconstruction BFM (c) Reconstruction extended BFM

Figure 2.2 – Best reconstruction of a target face (a) with the Basel Face Model (b) and the extended model (c), [Luthi et al., 2017]

to increase the flexibility of statistical models such as the BFM of [Paysan et al., 2009]. For instance, the BFM was built using data from mainly young people. Therefore it does not generalize very well to older people. This limiting factor can be attenuated by combining the deformations modeled of the BFM and a smooth Gaussian kernel with a small scale:

$$k\left(\boldsymbol{x},\boldsymbol{x}'\right) = k_{BFM}\left(\boldsymbol{x},\boldsymbol{x}'\right) + \boldsymbol{I}_{3\times 3}\exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{x}'\|^2}{30^2}\right). \tag{2.10}$$

Figure 2.2c compares reconstruction between the two models and shows that the extended model is much more expressive and generalizes better to unseen data.

In the work of Sutherland *et al.*, they proposed to build both the geometric and the appearance models of a 3DMM upon the GPMM framework. The statistical models are constructed from a single textured 3D scan and various handcrafted covariance kernels, yielding an entirely usable model. The quality of this custom model (*i.e. not data-driven*) is lower compared to the one learned with PCA but can still be used in any context where previously hand-produced 3DMMs have been required [Sutherland et al., 2020].

2.1.2 Facial Expression Model

Up to now, facial expressions have not been considered in the modeling process. However, they are a significant source of shape deformation and thus can not be neglected. Various types of models have been proposed to explicitly decouple the influence of the *identity* and the *expressions* on the shape deformation by modeling them separately. A standard method is to model both modalities in an additive manner [Egger et al., 2020].

The main idea behind additive models is to consider the offset due to facial expressions rather than the whole shape. Given two shapes of the same subject, one with expression $\boldsymbol{s}_{\text{exp}}$ and one in neutral pose $\boldsymbol{s}_{\text{ne}}$, the displacement due to the facial expression is $\boldsymbol{\Delta}_e = \boldsymbol{s}_{\text{exp}} - \boldsymbol{s}_{\text{ne}}$. The expression will effectively be transferred by adding the shape offset $\boldsymbol{\Delta}_e$ to the neutral shape

of another subject. Thus a PDM of expression offsets can be added to the identity model previously defined in Equation 2.3. The complete facial geometry model is given by

$$\boldsymbol{s}\left(\boldsymbol{w}^s, \boldsymbol{w}^e\right) = \bar{\boldsymbol{s}} + \boldsymbol{U}^s \boldsymbol{w}^s + \boldsymbol{U}^e \boldsymbol{w}^e, \tag{2.11}$$

where $\bar{\boldsymbol{s}}$ is the mean shape, $\boldsymbol{U}^s$ and $\boldsymbol{U}^e$ are the matrices of basis vectors of the identity and expression space (*i.e. eigenvectors*), and $\boldsymbol{w}^s$, $\boldsymbol{w}^e$ are identity and expression coefficients.

The most common practice in the literature is to use the BFM for modeling the identity variation and to learn the expression model on the FaceWarehouse dataset introduced by [Cao et al., 2014]. As both data do not share a common mesh topology, the method of Sumner and Popović is used to transfer the facial expressions of the FaceWarehouse to the same mesh triangulation used by the BFM [Sumner and Popović, 2004].

An alternate representation for facial expressions is the concept of *blendshapes*. This is the general approach to realistic facial animation. Initially developed in industry, it rapidly became a subject of academic research. Driven by a combination of simplicity, expressiveness, and interpretability, blendshapes became very popular [Lewis, J. P. et al., 2014].

A set of blendshapes is composed of a neutral face $\boldsymbol{b}_0$, typically in a resting position, and multiple targets $\boldsymbol{b}_i$. Targets are chosen to match pre-defined semantics of common facial expressions such as mouth-open, smile, frown or are based on the Facial Action Coding System (FACS) [Ekman and Friesen, 1976]. Figure 2.3 shows possible target expression candidates.

Using the delta blendshape formulation, a face is generated by adding a linear combination of expression specific offsets to the neutral pose as defined in:

$$\boldsymbol{f} = \boldsymbol{b}_0 + \boldsymbol{B}\boldsymbol{w} = \boldsymbol{b}_0 + \sum_{i=1}^{n} w_i \left(\boldsymbol{b}_i - \boldsymbol{b}_0\right), \tag{2.12}$$

where $\boldsymbol{b}_0$ is the neutral pose, $\boldsymbol{b}_i$ is a target expression and $w_i \in [0,1]$ is an expression weight. The weight w_i can be interpreted as a percentage indicating how much of the i-th expression is added to the mix. The target expression or *basis* $\boldsymbol{b}_i$ are not orthogonal to each other, unlike with PCA, hence information of different basis is potentially redundant.

The main limitation of the blendshape model is that the basis is person-specific, so this representation is not suitable for a generic face model. As a workaround, Bouaziz *et al.* proposed to learn a set of linear expression transfer operators $\boldsymbol{T}_i : \mathbb{R}^{3N} \rightarrow \mathbb{R}^{3N}$ that transform the neutral shape to generate personalized blendshapes: $\boldsymbol{b}_i = \boldsymbol{T}_i \boldsymbol{b}_0$ [Bouaziz et al., 2013].

Formally the parametric face model proposed by Bouaziz *et al.* is the combination of the PDM

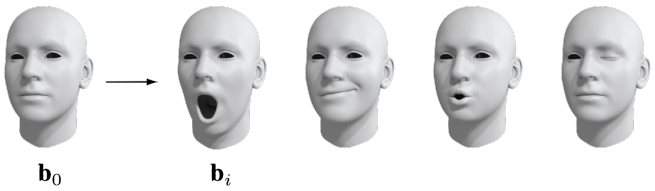


Figure 2.3 – Neutral expression $\boldsymbol{b}_0$ together with possible target blendshapes $\boldsymbol{b}_i$, [Bouaziz et al., 2013]

of Equation 2.3 and a set of d_e linear transfer operators $\boldsymbol{T}_i$, hence:

$$\boldsymbol{s}\left(\boldsymbol{w}^s, \boldsymbol{w}^e\right) = \bar{\boldsymbol{s}} + \boldsymbol{U}^s \boldsymbol{w}^s + \sum_{i=1}^{n} w_i^e \left(\boldsymbol{T}_i - \boldsymbol{I}\right)\left(\bar{\boldsymbol{s}} + \boldsymbol{U}^s \boldsymbol{w}^s\right). \tag{2.13}$$

The expressions are effectively decoupled from the identity by using expression transfer operators. This solves the main limitation of the blendshape model. Moreover, the expression model can be easily refined by extending the set of transfer operators.

Learning the expression transfer operators uses a formulation based on graph Laplacian, which depends on the mesh tessellation. If a mesh is not uniformly triangulated, the use of the graph Laplacian leads to aberrations (*i.e. discretization of Laplacian operator*). Moreover, the memory footprint of the method is substantial as the operator does not scale with high vertex density meshes (*i.e. each operator $\boldsymbol{T}_i$ is not sparse*). These two drawbacks make the blendshape model incompatible with the BFM.

2.1.3 Nonlinear Model

Most of the time, face identity and facial expression are modeled with linear subspace while assuming a Gaussian prior distribution. With recent advances in deep learning, alternate methods have been proposed to model facial deformation with nonlinear transformations or *networks*.

To extend the range of possible facial deformation, Tewari *et al.* proposed to model corrective fields $\mathcal{F}$ to add medium-scale geometry on top of existing statistical face models [Tewari et al., 2018]. The corrective model is based on nonlinear mapping $\mathcal{F} : \mathbb{R}^C \rightarrow \mathbb{R}^{3N}$ transforming the C-dimensional parameter space into per-vertex corrections. The shape model is parametrized by:

$$\boldsymbol{s}\left(\boldsymbol{w}^s, \boldsymbol{w}^e, \boldsymbol{w}^c, \Theta\right) = \boldsymbol{s}\left(\boldsymbol{w}^s, \boldsymbol{w}^e\right) + \mathcal{F}\left(\boldsymbol{w}^c, \Theta\right), \tag{2.14}$$

where $\boldsymbol{s}(\boldsymbol{w}^s, \boldsymbol{w}^e)$ is a coarse PCA-based shape model, $\boldsymbol{w}^c$ is the correction parameter and Θ is the parametrization of the corrective space learned during training and kept fixed at inference

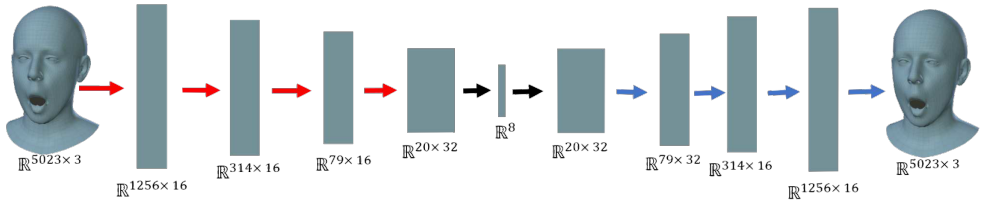


Figure 2.4 – Convolutional Mesh Autoencoder: The red and blue arrows indicate down- sampling and up-sampling layers respectively, [Ranjan et al., 2018]

time. In their work, the same principle is also applied to the appearance model.

Similarly, Tran *et al.* proposed to model the distribution of both modalities of a 3DMM by using Multilayer Perceptron (MLP) [Tran and Liu, 2018]. From a neural network's perspective, each component can be seen as a *shallow* network composed of a single fully connected layer without any activation function. Therefore the capacity of this model is quite limited. To improve the representative power of the model, a deep architecture is adopted for both shape and appearance.

Modeling face geometry with MLP can quickly lead to an overwhelmingly large number of parameters as the size of the mesh increases. Instead, Ranjan *et al.* proposed to use Graph Convolutional Network (GCN) to learn the shape distribution. They learn a nonlinear representation of the face using spectral convolution on a mesh surface directly [Ranjan et al., 2018]. The work of Defferard *et al.* on generalizing the convolution on graphs using fast Chebyshev filters is used for convolving over the face mesh [Defferrard et al., 2016]. The spectral convolutions and the sampling operations are combined in autoencoder-based network architecture to learn the shape distribution. An overview of the architecture, named Convolutional Mesh Autoencoder (*CoMA*), is given in Figure 2.4. Once the autoencoder has been trained, a new face can be generated by sampling from the latent space. Thus only the decoder is part of the face parametric model.

Alternate models have not only been proposed for shape distribution modeling but also appearance. In the work of Gecer *et al.*, a Generative Adversarial Network (GAN) is used to learn the distribution of facial appearance [Gecer et al., 2019]. GANs are very effective at preserving photo-realism, which is a crucial aspect of 3D reconstruction. However, they struggle to maintain the 3D coherency of the target distribution when trained on semi-aligned data. Thus textures from real images are converted to UV maps with per-pixel alignment to solve the issue. A total of 10'000 high-resolution UV maps are used to train a progressive GAN of [Karras et al., 2018] to model the facial appearance distribution. The main drawback with the proposed method is that residual illumination or occluders (*i.e. hairs or make-up*) will be part of the model depending on the quality of the data used during training.

Table 2.1 – Overview of some publicly available generative 3D face model together with the underlying attributes (*i.e. Id: Identity, Exp: Expression, Tex: Appearance, Pose: Head pose*)

Model	Attributes				#Vertex	Data
	Id	Exp	Tex	Pose		
Basel Face Model (BFM) 2009 [Paysan et al., 2009]	✓	✗	✓	✗	53490	100 women + 100 men
Large Scale Facial Model (LSFM) [Booth et al., 2016]	✓	✗	✗	✗	53215	9663 individuals †
Surrey Face Model [Huber et al., 2016]	✓	✓	✓‡	✗	29587	169 individuals
Faces Learned with an Articulated Model and Expressions (FLAME) [Li et al., 2017]	✓	✓	✗	✓	5023	3800 individuals for pose, 21000 frames for expression
Basel Face Model (BFM) 2017 [Gerig et al., 2018]	✓	✓	✓	✗	53149	100 women + 100 men, 160 expression scans
Convolutional Mesh Autoencoder (CoMA) [Ranjan et al., 2018]	✓	✓	✗	✗	5023	12 individuals, 12 extreme expressions, 20466 meshes in total

† At the time of the writing of this thesis, the website is offline.
‡ The texture model is only available for commercial use.

2.1.4 Publicly Available Models

Building a complete statistical model from scratch is a very demanding and complicated task. The data collection needs to be carefully planed to ensure all the human variability is gathered (*i.e. different ethnicities, facial expressions, age distribution*). The registration process is composed of a complex pipeline to guarantee all the data are correctly and precisely aligned. Lastly, the modeling step requires an efficient and faithful process to learn the underlying data distribution.

Therefore some statistical models have been made publicly available, at least for research purposes, to alleviate the whole process of face modeling. An exhaustive list of the most well-known models is given in Table 2.1. In addition, attributes covered by the model and information about the data used to build it are also provided for convenience.

2.2 Illumination

The perceived appearance of an object is determined by the interaction between the material that composes the object and the surrounding light. For the human face, the material is mostly skin. Thus the photometry of both reflectance and illumination must be explicitly modeled to simulate a realistic image formation process [Egger et al., 2020].

The Bidirectional Reflectance Distribution Function (BRDF) is often used to model how a surface reflects the light. It describes the directional dependence of local light reflection from an opaque surface. It is represented by a function $f_r(\boldsymbol{\omega}_i, \boldsymbol{\omega}_o)$ that provides the ratio of the *outgoing* reflected light in $\boldsymbol{\omega}_o$ direction to the *incoming* incident light from $\boldsymbol{\omega}_i$ direction. The irradiance $L_o(\boldsymbol{\omega}_o)$ in direction $\boldsymbol{\omega}_o$ can be expressed as a function of the light reflected from all

incident direction using a BRDF:

$$L_o(\boldsymbol{\omega}_o) = \int_{\Omega(\boldsymbol{n})} f_r(\boldsymbol{\omega}_i, \boldsymbol{\omega}_o) L_i(\boldsymbol{\omega}_i) \cos\theta_i \, \mathrm{d}\boldsymbol{\omega}_i \,, \tag{2.15}$$

where $L_i(\boldsymbol{\omega}_i)$ is the incident light from direction $\boldsymbol{\omega}_i$, $\Omega(\boldsymbol{n})$ is the half sphere around the surface normal $\boldsymbol{n}$ and θ_i is the angle between $\boldsymbol{\omega}_i$ and $\boldsymbol{n}$. A BRDF must satisfy some properties such as positivity $f_r(\boldsymbol{\omega}_i, \boldsymbol{\omega}_o) \geq 0$, the Helmholtz reciprocity $f_r(\boldsymbol{\omega}_i, \boldsymbol{\omega}_o) = f_r(\boldsymbol{\omega}_o, \boldsymbol{\omega}_i)$ and the conservation of energy:

$$\forall \boldsymbol{\omega}_i, \int_{\Omega(\boldsymbol{n})} f_r(\boldsymbol{\omega}_i, \boldsymbol{\omega}_o) L_i(\boldsymbol{\omega}_i) \cos\theta_i \, \mathrm{d}\boldsymbol{\omega}_i \leq 1, \tag{2.16}$$

in order to model a valid physical quantity. A very common and straightforward BRDF for a perfectly diffuse reflector is the Lambertian model. It assumes incident light is equally scattered in every direction, thus resulting in a constant function: $f_{\text{Lambert}}(\boldsymbol{\omega}_i, \boldsymbol{\omega}_o) = \rho_d / \pi$. The diffuse reflectivity or *albedo*, $\rho_d \in [0,1]$, is usually wavelength-dependent and can be interpreted as the color of the object itself. One well-known extension of the Lambertian formulation is the Phong model. It adds a constant ambient term and a specular component to simulate glossy reflectance. Using the BRDF representation, the Phong model is given by:

$$f_{\text{Phong}}(\boldsymbol{\omega}_i, \boldsymbol{\omega}_o) = \rho_d + \frac{\rho_a + \rho_s (\boldsymbol{r} \cdot \boldsymbol{\omega}_o)^\eta}{\boldsymbol{n} \cdot \boldsymbol{\omega}_i}, \tag{2.17}$$

where $\boldsymbol{r}$ is the reflection of $\boldsymbol{\omega}_i$ about $\boldsymbol{n}$, η is the *shininess* factor controlling the width of the specular lobe and ρ_a, ρ_s are ambient and specular albedos. It is important to note that f_{Phong} does not fulfill the physical properties established before. Hence it can not be considered a physically-valid BRDF. The computer graphics community has come up with extremely complex physically-valid BRDF that can be used in the context of 3DMM. However, these complex BRDFs are too complicated to be effectively integrated into the 3DMM fitting pipeline. Thus in most of the work, the Lambertian model or medium complexity non-physical models have been used.

The natural light of a scene is, by nature, very complex. It can be formed by multiple primary sources as well as secondary reflections from other surfaces. It is very common to assume that the illumination is distant relative to the size of the object to use a 2D environment map to approximate $L_i(\boldsymbol{\omega}_i)$. These environment maps are usually precomputed to simplify the lighting process but are specific to the scene.

The simplest light source model is a point source. The source is characterized by the intensity L_i and a unit vector $\boldsymbol{s}$ indicating the direction the light is emitted. Thus $L_i(\boldsymbol{\omega}_i)$ is a delta function. By plugging it into the previously defined BRDF (*i.e. Lambertian and Phong*), the

following shading models are obtained:

$$I_{\text{Lambert}} = L_i \rho_d \boldsymbol{n} \cdot \boldsymbol{s}, \quad I_{\text{Phong}} = L_i \left[\rho_a + \rho_d \boldsymbol{n} \cdot \boldsymbol{s} + \rho_s (\boldsymbol{r} \cdot \boldsymbol{v})^{\eta} \right], \tag{2.18}$$

where $\boldsymbol{v}$ is a unit vector indicating the viewer direction. The ambient and diffuse albedos represent the color of the surface and are often equal, $\rho_a = \rho_d$. These quantities are usually represented by a statistical model as defined in Equation 2.4. One key observation is that these models define the interaction between light and surface locally, meaning cast shadows are neglected.

These shading models depend on external information, such as source location and intensities, that might not always be available. To solve this issue, Ramamoorthi and Hanrahan proposed to use Spherical Harmonics (SH) to approximate complex natural illumination [Ramamoorthi and Hanrahan, 2001]. The SH shading model is defined as:

$$I_{SH}(\boldsymbol{n_i}) = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} l_{l,m} \mathcal{B}_{l,m}(\boldsymbol{n_i}) \tag{2.19}$$

where $\mathcal{B}_{l,m}(\boldsymbol{n_i})$ are orthogonal basis functions and $l_{l,m}$ are coefficients describing the reflectance and the illumination. The degree and the order of the SH are denoted by l and m, respectively. Hence, using SH up to order 2 (*i.e.* $l = \{0,1,2\}$) the reflected light field from a convex Lambertian object can be well approximated. For any lighting conditions, the average accuracy of a second-order approximation is at least 98% if non-negativity of the illumination is ensured [Ramamoorthi, 2006]. The use of low order SH is very common in the current state-of-the-art. However, Dib *et al.* showed that higher-order SH (*i.e. up to the 9-th order*) coupled with a simplified Cook-Torrance BRDF leads to a better approximation of the light interaction with non-Lambertian surfaces [Dib et al., 2021b,Dib et al., 2021a]. This observation opens new research directions for the future.

Specular albedo, ρ_s, has received more attention in the context of 3DMM in recent years. In their work, Smith *et al.* introduced a new image capture and processing pipeline independent of external factors such as the illumination, the camera, and the geometry to acquire diffuse and specular maps. Moreover, they released the first statistical model of facial specular albedo maps [Smith et al., 2020]. The specular prior is coupled with SH light representations to generate the final observed color leading to a more realistic model due to the effective disentanglement of the extrinsic parameters such as the reflectance and the illumination.

An alternate model capturing global illumination effects is the *ambient occlusion* model. It assumes the illumination is perfectly diffuse, in other terms, $L_i(\boldsymbol{\omega}_i)$ is constant everywhere. The shading depends only on how much the incident hemisphere is occluded. The ambient

occlusion coefficient factor $A_{\boldsymbol{v}}$ at vertex $\boldsymbol{v}$ is given by:

$$A_{\boldsymbol{v}} = \frac{1}{\pi} \int_{\Omega(\boldsymbol{n})} V(\boldsymbol{v}, \boldsymbol{\omega}) (\boldsymbol{v} \cdot \boldsymbol{\omega}) \, \mathrm{d}\boldsymbol{\omega}, \tag{2.20}$$

where $V(\boldsymbol{v}, \boldsymbol{\omega})$ is a visibility function defined as zero if the vertex $\boldsymbol{v}$ is not visible from direction $\boldsymbol{\omega}$ and one otherwise. The *bent normal* is then defined as the average unoccluded direction. Adopting SH illumination with bent normals and scaling the result by the ambient occlusion coefficient gives a rough approximation of the global illumination effect. With this approach, Aldrian and Smith learned a linear subspace of ambient occlusion and bent normals and used them in their fitting pipeline [Aldrian and Smith, 2012].

The concept of Precomputed Radiance Transfer (PRT) model has been proposed by Sloan *et al.* to approximate the light transport at each vertex to account for shadowing and inter-reflection. These transfer functions are precomputed offline and can be applied with any incident illumination at inference time [Sloan et al., 2002]. Building upon this framework, Schneider *et al.* have learned a linear mapping function predicting PRT transfer matrices out of 3DMM shape coefficients [Schneider et al., 2017].

2.3 Rendering

Based on the methods presented in the previous sections, it is now possible to generate a realistic human face from a relatively small set of parameters. The facial geometry, namely the identity and the facial expressions, and the facial appearance are modeled by independent PDM. The representation of the illumination that shines onto the face (i.e. spherical harmonics) and these statistical face models define a whole scene. Only a small number of parameters parameterizes the scene. The next step is to transform this 3D representation into an actual image. This process is usually referred to as the *rendering* of a scene.

In the following sections, different aspects of the rendering process in the context of 3DMM will be covered. The first component to be discussed will be the abstraction of the camera together with the different geometric transformations applied to the 3D scene, followed by the different methods of rendering used to convert 3DMM into an image.

2.3.1 Camera Model

The way how 3D points are projected to a 2D location on the image plane is described by a camera model. Generally, the camera model handles the projection and englobes every geometric transformation applied to each vertex along the way. In standard modern rendering pipelines (*i.e. OpenGL*), six coordinate systems and five transformations are considered part of the camera model, as illustrated in Figure 2.5.

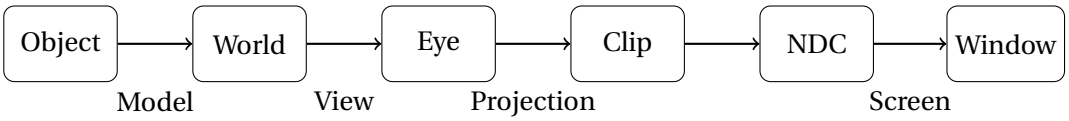


Figure 2.5 – Geometric transformations and coordinates system used in standard graphics pipeline. Rounded rectangles represent a coordinate system and arrows are transformation to move between them.

The points are represented in homogeneous coordinates to simplify the formulation of affine or projective transformation. The homogeneous coordinate system represents a point in $\mathbb{R}^n$ space by a vector in $\mathbb{R}^{n+1}$ space instead. For instance, any (x, y) point laying into the 2D Euclidean space will have $(x, y, 1)$ as homogeneous coordinates. Thus from a coordinate triplet $(\lambda x, \lambda y, \lambda)$, the original coordinates can be recovered by dividing all the components by the last element λ and get back $(x, y, 1)$. Therefore a single point in 2D space can be represented by an infinite number of homogeneous points. In other words, a single point in 2D space is mapped to every point laying on a line in 3D space, making it very useful for projective geometry.

The object space is the local coordinate system where the object or *mesh* is lying (*i.e. initial position, orientation, and scale*). This mesh can then be moved around by applying a linear transformation, usually called *model* transformation, bringing it to the *World* space, which is common for every object. The model matrix $\boldsymbol{M}_{\text{model}}$ is composed of isotropic scaling, rotation, and translation. It acts as an adaptation step to bring every object to the same reference system (*i.e. World space*).

Once objects are in a common reference system, they are brought to the camera's coordinate system called *eye* space in the literature. The transformation is defined by the *view* matrix $\boldsymbol{M}_{\text{view}}$ and is composed of rotation and translation. For convenience, the matrix is usually characterized by the camera's position in the world space and the direction it is pointing to.

With points aligned with the camera, the projection matrix $\boldsymbol{M}_{\text{proj}}$ is applied to bring them into the homogeneous *clip* space as defined in:

$$\boldsymbol{x}_{\text{clip}} = \boldsymbol{M}_{\text{proj}} \boldsymbol{M}_{\text{view}} \boldsymbol{M}_{\text{model}} \boldsymbol{x}_{\text{model}}, \tag{2.21}$$

where $\boldsymbol{x} = [x, y, z, 1]$ and $\boldsymbol{x}_{\text{clip}} = [x_{\text{clip}}, y_{\text{clip}}, z_{\text{clip}}, w_{\text{clip}}]$ are vertex position defined in homogeneous coordinates.

There are two major types of projection. The first one is the orthographic projection, in which the projection lines are orthogonal to the image plane. The location on the image plane is given by translating the 3D point parallel to the camera optical axis. Moreover, objects of the same size located at a different distance from the observer will have the same size on the image. Hence perspective information is not preserved by the model. The second is the

perspective projection, which models an ideal pinhole camera where the aperture is described as a point, and no lenses are used to focus light. Thus the perspective information will be preserved with this transformation. Both projection matrices are given in:

$$\boldsymbol{M}_{\text{proj}}^{\text{ortho}} = \begin{bmatrix} \frac{2}{w} & 0 & 0 & 0 \\ 0 & \frac{2}{h} & 0 & 0 \\ 0 & 0 & -\frac{2}{f-n} & -\frac{f+n}{f-n} \\ 0 & 0 & 0 & 1 \end{bmatrix}, \boldsymbol{M}_{\text{proj}}^{\text{persp}} = \begin{bmatrix} \frac{1}{\tan\left(\frac{fov}{2}\right)\cdot ar} & 0 & 0 & 0 \\ 0 & \frac{1}{\tan\left(\frac{fov}{2}\right)} & 0 & 0 \\ 0 & 0 & -\frac{f+n}{f-n} & -\frac{2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{bmatrix}, \quad (2.22)$$

where w and h are the width and height of the image, n and f represent the *near* and *far* planes, fov is the field of view of the camera, and ar is the aspect ratio of the camera.

The projection matrix defines a viewing volume in front of the camera, called *frustrum*, that is mapped to the Normalized Device Coordinate (NDC) space. The volume is characterized by the *near* and *far* planes (*i.e. orthographic*), the field of view, and the camera's aspect ratio (*i.e. perspective*). Points that are outside this volume will be discarded (*i.e. clipped*).

Given all points are in the clip space, a final operation called *perspective division* is performed where the x, y, and z coordinates are divided by the homogeneous w component. This division is what transforms a 4D clip space point into a 3D normalized device coordinate.

The final transformation linearly maps the resulting coordinates $\boldsymbol{x}_{\text{ndc}}$ into actual screen position as defined in:

$$\begin{pmatrix} x_{\text{screen}} \\ y_{\text{screen}} \end{pmatrix} = \begin{pmatrix} \frac{w}{2}\left(x_{\text{ndc}}+1\right) \\ \frac{h}{2}\left(y_{\text{ndc}}+1\right) \end{pmatrix}. \quad (2.23)$$

For the work carried out in this thesis, the camera is placed at the origin of the world space, $\boldsymbol{M}_{\text{view}} = \boldsymbol{I}$, without loss of generality.

2.3.2 Object Space Rendering

In the original work of Blanz and Vetter, the object space rendering technique was used [Blanz and Vetter, 1999]. This rendering process does not create a complete image but only defines pixel intensities for sub-regions on the image plane. Given an appearance and illumination model, a single color attribute is computed either for each triangle center or vertex (*i.e. per-triangle or per-vertex attributes*). The visibility of the mesh components (*i.e. triangles or vertices*) is established with the z-buffering technique. Only the visible parts are projected onto the image plane and interpolated at the pixel locations. A detailed explanation of the whole process is given in [Booth et al., 2017].

Many proposed methods have been using object space rendering. However, over recent years, reconstruction pipelines tend to use more classical rasterization pipelines. This image formation process will be discussed in the following section.

2.3.3 Differentiable Rasterization

Rasterization is the process of converting 3D models or complete scenes into an actual image. This rendering technique is the most common way to convert 3D scenes to images. The image is created by projecting the triangles or polygons that form the mesh onto the image plane, then identifying which pixels are covered by which triangles or polygons to compute the proper final color. The approach is quick and can be well parallelized through the use of Graphics Processing Units (GPUs). Over the years, the computer graphics community has developed multiple tricks and techniques to make the results appear photorealistic to humans.

OpenDR

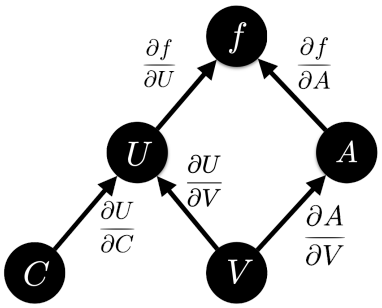
Rasterizers are designed to solve the forward process of image synthesis (*i.e. convert a 3D scene into an image*) and are not meant to be inverted. In their work, Loper and Black proposed an approximate differentiable renderer that models the relationship between changes in the scene and the image observations. They used it in the context of inverse rendering [Loper and Black, 2014].

The proposed framework named *OpenDR* is built upon an already available realistic graphics engine. The differentiable renderer is defined as a process that supplies pixels as a function of the inputs (*i.e. geometry, appearance, and camera*) and provides derivatives of the pixel values with respect to the inputs.

The rendering function $f(\Theta)$ creates the image from the set Θ of all the parameters. The considered parameters are the vertex position V, the camera parameters C, and the per-vertex brightness or appearance A, then $\Theta = \{V, C, A\}$. To make the rendering function differentiable, some assumptions about the input quantities have been made and are listed below.

The per-pixel surface *appearance* A is the product of an albedo and per-vertex brightness term. The brightness factor models the effect of reflectance and lighting and is represented as spherical harmonics. However, other direct lighting models can be used. The geometry of the 3D scene is assumed to be composed of only triangles parametrized by *vertices* V. Lastly the *camera* C is approximated using the pinhole-plus-distortion projection model. These approximations are close to those made by modern graphics rendering pipelines, with one important exception. The appearance can only be modeled as per-vertex attributes, whereas modern engines allow defining per-pixel attributes. This limitation implies that vertex density has to be high to have proper light shading.

The complete differentiation chain is shown in Figure 2.6a. The intermediate variable U is



(a)

(b)

Figure 2.6 – (a) Partial derivative structure used in *OpenDR*. (b) Left: a rotating quadrilateral. Middle: OpenDR's predicted change in pixel values with respect to in-plane rotation. Right: finite differences recorded with a change to in-plane rotation, [Loper and Black, 2014].

introduced to simplify the derivative of f with respect to geometry and the camera. It indicates the 2D coordinates on the image plane of a projected vertex. The derivatives are grouped into the effects of appearance $\left(\frac{\partial f}{\partial A}\right)$, the changes in projected coordinates $\left(\frac{\partial U}{\partial C}, \frac{\partial U}{\partial V}\right)$, and the effects of image-space deformation $\left(\frac{\partial f}{\partial U}\right)$.

The projected geometry is colored at the pixel level by interpolating the attributes of the surrounding vertices. The interpolation uses barycentric coordinates inside the triangle to define the amount of information that has to be taken from each vertex. Therefore the partial derivative $\frac{\partial f}{\partial A}$ can be quickly found. Partial derivative $\frac{\partial A}{\partial V}$ could be zero if only ambient color is used (*i.e. no illumination model*) or may be assigned to the derivative of spherical harmonics or any other illumination model.

The image values also depend on the geometry and the camera via the projected coordinates U. Therefore, the partial derivatives $\frac{\partial f}{\partial V}$ and $\frac{\partial f}{\partial C}$ are defined using the chain rule:

$$\frac{\partial f}{\partial V} = \frac{\partial f}{\partial U}\frac{\partial U}{\partial V}, \quad \frac{\partial f}{\partial C} = \frac{\partial f}{\partial U}\frac{\partial U}{\partial C}. \tag{2.24}$$

Since the projection model is well-defined, the partial derivatives $\frac{\partial U}{\partial V}, \frac{\partial U}{\partial C}$ are straightforward and well documented. The main challenge is to define the partial derivative $\frac{\partial f}{\partial U}$ properly.

The pixels on the projected surface is segmented into occlusion *boundary* pixels and *interior* pixels to estimate the partial derivative $\frac{\partial f}{\partial U}$. The change caused by interior pixels is linked to the translation of the surface patch. On the other hand, the change induced by the boundary pixels is caused by the replacement of one surface with another one. Boundary pixels are the one that lies on edges that pass the depth test and are joining triangles with opposite-facing normals (*i.e. one in the direction of the camera and one facing away*). For the classification of the pixels, three categories are considered: interior, interior/boundary, and many-boundary.

Interior label is assigned to a pixel that contains no occlusion boundaries. The intensity changes are piecewise smooth with respect to geometry changes since the appearance is the product of interpolated texture and interpolated color. Therefore the partial derivative is approximated using an image-space first-order Taylor expansion. A filtering operation performs the approximation. More importantly, the filter operation is not allowed to cross or include boundary pixels to be valid. Specifically, when pixels are not next to an occlusion boundary, the horizontal derivative is approximated using the kernel $\frac{1}{2}[-1,0,1]$. For pixels with occlusion boundaries on the left, the kernel is set to $[0,-1,1]$ and $[-1,1,0]$ when the occlusion boundaries are on the right side. In the presence of occlusion boundaries on both sides, the derivative is set to zero. The same process is applied for the vertical derivative with the transposed kernels.

Interior/boundary label is used for a pixel that is intersected by one occlusion boundary. In this case, the derivative is approximated using the kernel $\frac{1}{2}[-1,0,1]$ and its transpose. Instead of peeking behind the occluding boundary, the neighboring pixel is used as a proxy, assuming that the difference is negligible.

Many-boundary label is reserved for a pixel that is intersected by more than one occlusion boundary. Since only a few pixels will be affected for this particular case, it is treated the same way as the interior/boundary case for simplicity. Otherwise, the exact computation would require object-space analysis and be very expensive.

With partial derivatives of the image with respect to the projected geometry defined, the whole chain is complete. It is possible to invert the renderer. Figure 2.6b compares the derivatives of the differentiable renderer defined earlier with finite differencing in the case of a rotating plane. The correct finite differencing epsilon is pixel-dependent. However, using the proposed framework, the derivatives are correctly approximated.

Soft Rasterizer

A fundamental discretization step in standard graphics renderer, called rasterization, prevents the image formation process from being differentiable, as mentioned before. To tackle this issue, Liu *et al.* proposed reformulating the rendering process as an aggregation function that fuses the probabilistic contributions of all the triangles with respect to the rendered pixels [Liu et al., 2019].

Traditional rasterization can be interpreted as a binary mask determined by the relative positions between the triangles and pixels. At the same time, z-buffering merges the rasterization results in a pixel-wise one-hot manner based on the depth of the triangles. Given this interpretation, the challenge is to model the discrete binary mask and the one-hot merging operation in a soft and differentiable way. The differentiability is achieved by the use of two components, probability maps $\{\mathcal{D}_j\}$ that model the probability of each pixel to belong in a specific triangle f_j, and an aggregate function $\mathcal{A}(\cdot)$ responsible for merging per-triangle color maps based on

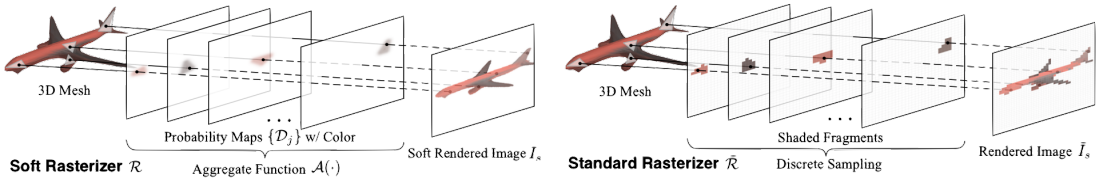


Figure 2.7 – Soft Rasterizer $\mathcal{R}$ (left), a truly differentiable renderer, which formulates rendering as a differentiable aggregating process $\mathcal{A}(\cdot)$ that fuses per-triangle contributions $\{\mathcal{D}_i\}$ in a "soft" probabilistic manner. The approach attacks the core problem of differentiating the standard rasterizer, which cannot flow gradients from pixels to geometry due to the discrete sampling operation (right), [Liu et al., 2019].

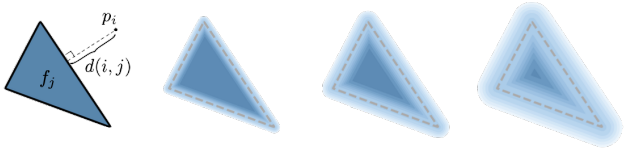
the probability maps $\{\mathcal{D}_j\}$ and the relative depths of the triangles. This probabilistic approach allows gradients to flow to the occluded and far-range vertices, which is a unique property of the framework. A comparison between a traditional rasterizer and a soft rasterizer is shown in Figure 2.7.

The influence of triangle f_j on each pixel on the image plane is modeled by the probability map $\mathcal{D}_j$. It takes into account the distance and the relative position between p_i and $\mathcal{D}_j$. The probability of pixel p_i to be overlapped by triangle f_j is defined as follow:

$$\mathcal{D}_j^i = \text{sigmoid}\left(\delta_j^i \cdot \frac{d^2(i,j)}{\sigma}\right), \tag{2.25}$$

where $\sigma > 0$ controls the sharpness of the probability distribution and δ_j^i is an indicator function defined as $\delta_j^i = \{+1, \text{if} p_i \in f_j; -1, \text{otherwise}\}$. The function $d(i,j)$ measures the distance from p_i to the closest edge of triangle f_j. Different distance metrics can be used, such as the barycentric or l_1 distance function. However, a natural choice is the Euclidean distance. The output value is normalized between $(0,1)$ using the sigmoid function, providing an accurate continuous approximation of a binary mask. Pixels inside triangle f_j are mapped to the range $(0.5,1)$ while the ones outside are in the range $(0.0,0.5)$ thanks to the indicator function δ_j^i. Figure 2.8 shows the effect the σ coefficient has on the probability map. With a smaller value of σ, the distribution becomes sharper. While with a larger value it is smoother. This behavior allows tuning the influence of the triangles on the image plane. As σ goes to zero, the probability maps will converge to the identical triangles similar to a traditional rasterizer.

Similar to the probability map $\mathcal{D}_j$, each triangle f_j has its own color map C_j. The color map is filled at each pixel p_i by interpolating per-vertex color using barycentric coordinates. The color maps $\{C_j\}$ are then merged with an aggregation function $\mathcal{A}(\cdot)$ to obtain the rendering output I based on $\{\mathcal{D}_j\}$ and the relative depths $\{z_j\}$. Similar to the softmax operator, the



(a) ground truth (b) $\sigma = 0.003$ (c) $\sigma = 0.01$ (d) $\sigma = 0.03$

Figure 2.8 – Probability maps of a triangle under Euclidean metric. (a) definition of pixel-to-triangle distance;(b)-(d) probability maps generated with different σ, [Liu et al., 2019].

aggregation function $\mathcal{A}_S$ is defined as:

$$I^i = \mathcal{A}_S\left(\{C_j\}\right) = \sum_j w_j^i C_j^i + w_b^i C_b, \tag{2.26}$$

where C_b is the background color, the weights $\{w_j\}$ satisfy $\sum_j w_j^i + w_b^i = 1$ and are defined as:

$$w_j^i = \frac{\mathcal{D}_j^i \exp\left(z_j^i/\gamma\right)}{\sum_k \mathcal{D}_k^i \exp\left(z_k^i/\gamma\right) + \exp\left(\epsilon/\gamma\right)}, \tag{2.27}$$

where z_j^i is the normalized inverse depth of the 3D points on f_i which project on p_i, ϵ is a small constant enabling the background color, and γ controls the sharpness of the aggregation function. When $\gamma \to 0$, the aggregation function only outputs color from the nearest triangle, similar to z-buffering.

Rendering the silhouette of an object is independent of its color and depth maps. Therefore an alternate aggregation function $\mathcal{A}_O$ can be defined based on the binary occupancy:

$$I_s^i = \mathcal{A}_O\left(\{\mathcal{D}_j\}\right) = 1 - \prod_j \left(1 - \mathcal{D}_j^i\right). \tag{2.28}$$

In Equation 2.28, the silhouette is modeled as the probability of having at least one triangle covering the pixel p_i.

Other forms of aggregation functions might exist. An option is to use a neural network to approximate a universal aggregation function $\mathcal{A}_N$. In their work, Liu *et al.* have shown that the performances are slightly better by learning the aggregation function $\mathcal{A}_N$ than the non-parametric function $\mathcal{A}_O$. Still, it comes with a higher computational cost.

Modular Differentiable Renderer

Recently Laine *et al.* introduced a general-purpose differentiable rendering pipeline yielding high performance compared to other methods by utilizing existing highly-optimized hardware graphics pipeline [Laine et al., 2020]. The proposed design supports all the major operations of a modern graphics pipeline, such as the rasterization of a large number of triangles, per-vertex attribute interpolation, filtered texture lookups, and shading and geometry processing.

A 3D scene can be expressed in terms of geometric shapes, materials, camera, and lighting models. Converting it into a 2D image reduces to two problems. First, determine what parts are visible from each pixel, then what color the visible parts appear to be. Then a differentiable renderer must provide gradients for all the parameters of the scene. Taking everything into account, the final color I_i of a pixel located at (x_i, y_i) in the image plane is given by:

$$I_i = \underset{x,y}{\text{filter}}\big(\text{shade}\big(M\big(P\big(x,y\big)\big),\text{lights}\big)\big)\big(x_i,y_i\big). \tag{2.29}$$

The $P\big(x,y\big)$ denotes the world point visible at continuous screen position $\big(x,y\big)$ after projection. $M\,(P)$ represents all the spatially varying factors such as texture maps and normal vectors on the surfaces defining the scene (*i.e. triangles*). The shade function represents the interaction between the surface and the light. Lastly, the antialiasing filter, essential for image quality and differentiability, is applied to the shading output in continuous $\big(x,y\big)$. The final color is sampled at the pixel center $\big(x_i,y_i\big)$.

To maximize the modularity of the rendering pipeline, some design choices have been made. The first one is to work with data already in the clip space. This way, no particular assumptions about the geometric transformations or projection have to be made. Secondly, the whole system is built upon the concept of deferred shading [Deering et al., 1988]. First, the idea is to compute the $M\big(P\big(x,y\big)\big)$ term from Equation 2.29 for each pixel and store it in an image-space regular grid (*i.e. texture*). Then shading is performed on the same regular grid and can be implemented outside the rasterizer. Lastly, the use of image-based antialiasing to convert discontinuities to smooth changes in order to provide proper gradients.

The rendering pipeline is composed of four primitive operations illustrated in Figure 2.9 that include customized gradient computation. Each component will be briefly discussed below to grasp the general idea behind the proposed framework. Note that it is highly inspired by modern graphics pipelines.

Rasterization Using triangles with their corresponding vertex position given in clip-space homogeneous coordinates $\big(x_c,y_c,z_c,w_c\big)$, the rasterizer outputs a 2D sample grid where each pixel stores a tuple composed of (Id, $u, v, z_c/w_c$). The Id value indicates the index of the triangle overlapping the sample, (u,v) are the perspective corrected barycentric coordinates indicating the position within the triangle, and z/w denotes the depth in normalized device coordinates. The 2×2 Jacobian of the barycentric with respect to the screen coordinates $\mathcal{J}_{uv} = \partial\{u,v\}/\partial\{x,y\}$

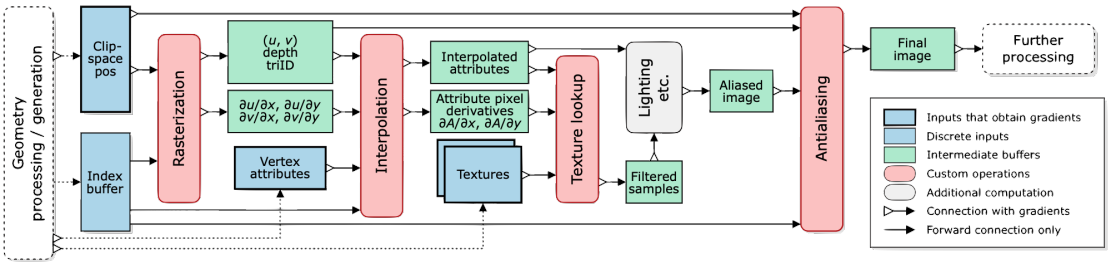


Figure 2.9 – A simple differentiable rendering pipeline with our proposed primitive operations highlighted in red. The input data for rendering (blue) may be generated by, e.g., a neural network if the pipeline is part of a larger computation graph. In simpler setups, the geometry processing might include only the model/view/perspective transformations for vertex positions with other inputs being constants or learnable parameters. All intermediate buffers (green) are in image space. Connections with gradients are denoted by a white triangle. Channel counts are fixed only for vertex positions and indices and in the intermediate buffers produced by the rasterization operation. There are no restrictions on the channel counts for vertex attributes, textures, related intermediate data, or the output image [Laine et al., 2020].

is also provided. This information is needed later on to define the footprint for the filtered texture lookups.

Interpolation This step creates a mapping between pixels and attributes defined at the vertex level. The value for a given pixel is computed as a weighted sum of per-vertex attributes. The weights are defined as perspective corrected barycentric coordinates (*i.e. provided by the rasterizer*). If needed, the screen-space derivatives $\mathcal{J}_A = \partial A / \partial \{x, y\}$ is also computed.

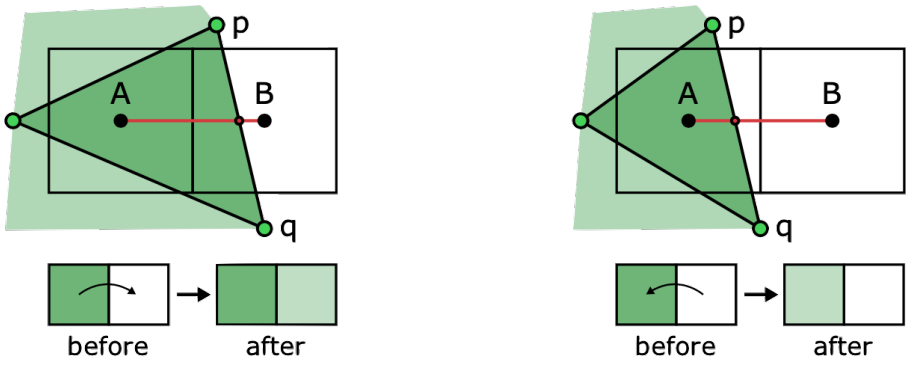
More specifically, the interpolated attribute A for a pixel located at (x, y) is defined as:

$$A = uA_{i_0} + vA_{i_1} + (1 - u - v)\, A_{i_2}, \tag{2.30}$$

where A_i is the attribute defined at the i-th vertex, $i_{0,1,2}$ are the index of the triangle visible from pixel (x, y), and u, v are the barycentric weights. The screen-space derivatives are computed using $\mathcal{J}_{uv}$ and the chain rule as $\partial A / \partial \{x, y\} = \mathcal{J}_{uv} \cdot \partial A / \partial \{u, v\}$.

Texture filtering Using the incoming screen-space derivative of the attributes used as texture coordinates, the fractional level in the mipmap pyramid is selected. The proper level selection is based on the texture-space length of the major axis of the sample footprint defined by the screen-space derivative of the texture coordinates. Once position within the mipmap pyramid is determined, trilinear interpolation from the eight surrounding texels is performed in a very similar way to attribute interpolation.

Antialiasing The shading is expected to be band-limited and without aliasing within the surfaces (*i.e. triangles*). However, aliasing happens at visibility discontinuities (*i.e. on edges*) due to point-sampled visibility. Therefore it can not produce visibility-related gradients for



(a) (b)

Figure 2.10 – Illustration of the analytic antialiasing method of [Laine et al., 2020]. A vertical silhouette edge p, q passes between centers of horizontally adjacent pixels A and B. This is detected by the pixels having a different triangle ID rasterized into them. Pixel pair A, B is processed together, and one of the following cases may occur. (a) The edge crosses the segment connecting pixel centers inside pixel B, causing color of A to blend into B. (b) The crossing happens inside pixel A, so blending is done in the opposite direction. To approximate the geometric coverage between surfaces, the blending factor is a linear function of the location of the crossing point. This antialiasing method is differentiable because the resulting pixel colors are continuous functions of positions of p and q.

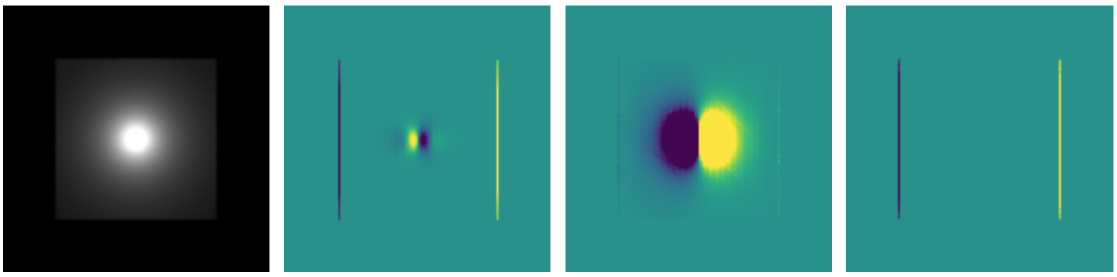
vertex positions. Antialiasing transforms these discontinuities into smooth changes making it possible to compute gradients. The antialiasing step must be performed at the very end of the rendering pipeline.

The proposed method tackles the issue with an image-based post-process antialiasing technique, as illustrated in Figure 2.10, a variant of distance-to-edge and geometric post-processing antialiasing.

This was the last component of the differentiable rendering pipeline of Laine *et al.* The design of the framework aims at handling the bare minimum in the rendering process to leave the door open for developing new parameterizations for geometry, texture, and lighting models. Therefore this is the system selected for the work carried out in this thesis.

2.3.4 Differentiable Ray Tracing

Ray tracing is a rendering technique, orthogonal to rasterization, used to generate images. The image is formed by tracing the path of the light or *ray* and simulating its interactions with objects. The resulting image is of high quality, higher than traditional rasterization approaches, due to the technique being close to the physic of the image formation process. However, the image quality comes at the cost of a large computational footprint. Therefore the method is best suited for applications where longer computation time is tolerated.



(a) Planar scene (b) OpenDR, [Loper and Black, 2014] (c) NMR, [Kato et al., 2018] (d) Ray tracing

Figure 2.11 – (a) A plane lit by a point light close to the plane. We are interested in the derivative of the image with respect to the plane moving right. Since the point light stays static, the derivatives should be zero except for the boundary. (b) (c) Previous work uses color buffer differences to approximate the derivatives, making them unable to take large variation between pixels into account and output non zero derivatives at the center. (d) Our method outputs the correct derivatives, [Li et al., 2019].

Similar to rasterization, ray tracing is not directly differentiable. However, many researchers have shown interest in tackling this issue and making ray tracing invertible over recent years. One example is the work of Li *et al.*, in which they introduce the first general-purpose differentiable ray tracer [Li et al., 2019].

The rendering integral includes visibility terms that are not differentiable, making it challenging to provide gradients of an image. To tackle this issue, Li *et al.* proposed a new edge sampling strategy that directly samples the Dirac delta functions introduced by the derivatives of the discontinuous integrand. With an efficient hierarchical spatial-based importance sampling scheme, the method can produce gradients in seconds or minutes depending on the complexity of the scene and the desired precision.

Comparison of the gradients computed by the ray tracer, OpenDR, and Neural 3D Mesh Renderer (NMR) [Kato et al., 2018] for a plane moving to the right and lit by a single point light is shown in Figure 2.11. It highlights that both previous renderers output the wrong gradient. The light source being fixed, the illumination on the plane is static, and the gradients should be zero except for the pixels on the boundaries of the plane. However, since both OpenDR and NMR use the difference between pixel intensities to approximate the derivatives, the illumination change introduces wrong approximations and output non-zero derivatives. On the other hand, ray tracer correctly estimates the gradients thanks to the edge sampling strategy.

2.4 Cost Functions

In the analysis-by-synthesis framework, estimating the set of parameters $\mathcal{X}$ that most probably was used to generate a given image requires solving a nonlinear optimization system. This

optimization problem depends on a combination of different weighted cost functions tailored for this task. Over recent years, different objective functions and optimization strategies have been proposed to tackle the challenges in monocular face reconstruction.

In the context of 3DMM, these objective functions can be grouped into three categories: appearance, geometry, and regularization. The appearance-based loss function acts at pixel level and measures the distance between the observed image and the one synthesized with the generative model. The geometric-based objective function will measure distances between points either in 3D or 2D between the current instance and the ground truth. Lastly, the regularization will group all the cost functions that ensure that all the statistical models are operating in their valid range and the loss functions that ensure some properties are respected (*i.e. soft constraints*).

Each category of objective functions will be covered in the subsequent sections, and the most commonly used cost functions will be detailed.

2.4.1 Appearance

Appearance is the main driving factor in an analysis-by-synthesis framework. The distance between an instance $\bar{\boldsymbol{I}}$ of the generative model is compared to the image $\boldsymbol{I}$ under observation since the approximation should be close to the original data. The distance or *photometric* error is computed in the image space at the pixel level by using a l_p norm as defined in:

$$\mathcal{L}_{\text{ph}}\left(\bar{\boldsymbol{I}}, \mathcal{X}\right) = \frac{1}{|\mathcal{F}|} \sum_{x,y \in \mathcal{F}} \| \bar{\boldsymbol{I}}_{x,y} - \boldsymbol{I}_{x,y}\left(\mathcal{X}\right) \|_p^p, \tag{2.31}$$

where $\mathcal{F}$ defines the region covered by the projected geometry (*i.e. foreground*) [Blanz and Vetter, 1999]. Therefore only the visible vertices will contribute to this term. Usually, a l_2 norm will be used to measure the pixel distance, but in recent years l_1 norm has become quite popular for image generation tasks thanks to the advances in deep learning.

However, the objective function defined in Equation 2.31 has a flaw. When the number of pixels in $\mathcal{F}$ decreases, the objective function will most probably decrease as well. This phenomenon is referred to as the *shrinking* effect. To counteract this effect, Schönborn *et al.* have shown that using an implicit background model solves the issue [Schönborn et al., 2015]. The updated cost function is given by:

$$\mathcal{L}_{\text{ph}}\left(\bar{\boldsymbol{I}}, \mathcal{X}\right) = \frac{1}{|\mathcal{F}|} \sum_{x,y \in \mathcal{F}} \| \bar{\boldsymbol{I}}_{x,y} - \boldsymbol{I}_{x,y}\left(\mathcal{X}\right) \|_p^p + \frac{1}{|\mathcal{B}|} \sum_{x,y \in \mathcal{B}} b\left(\bar{\boldsymbol{I}}_{x,y}\right), \tag{2.32}$$

where $\mathcal{B}$ represents the background region in the image space, and $b(\cdot)$ is a background model. They have shown that even simple models like a constant, a Gaussian, or an image

histogram-based model are enough.

For most "In-the-Wild" face datasets, the images depict occluders such as glasses, hands, facial hairs, among many other things. These occlusions present a particular challenge for the reconstruction algorithm. The model tries to explain the observed appearance (*i.e. over the occluders*) by using prior knowledge that does not include such variations. This leads to the wrong estimation of the parameters of the model. However, one can define regions in the image space in which the appearance can be explained by the underlying 3DMM [Egger et al., 2018]. Then it is guaranteed that the measured error makes sense with respect to model priors. Adding it to the previous objective function, the updated metric is defined as:

$$\mathcal{L}_{\text{ph}}\left(\bar{\boldsymbol{I}},\mathcal{X}\right)=\frac{1}{\sum_{x,y\in\mathcal{F}}\delta_{x,y}}\sum_{x,y\in\mathcal{F}}\delta_{x,y}\cdot\|\bar{\boldsymbol{I}}_{x,y}-\boldsymbol{I}_{x,y}\left(\mathcal{X}\right)\|_{p}^{p}+\frac{1}{|\mathcal{B}|}\sum_{x,y\in\mathcal{B}}b\left(\bar{\boldsymbol{I}}_{x,y}\right), \tag{2.33}$$

where $\delta_{x,y}\in[0,1]$ indicates the probability of the pixel $\left(x,y\right)$ being explainable by the appearance model.

This pixel-wise loss function can be used within an L-level image Gaussian pyramid to increase the robustness of the reconstruction (*i.e. avoid bad local minima*) as proposed in [Henderson and Ferrari, 2019]. This multi-scale objective function guides the reconstruction algorithm by providing signals with various levels of complexity.

2.4.2 Geometry

Geometry-based objective functions are widespread in the context of 3DMM. The most commonly used is facial landmarks alignment with pre-detected fiducial points. Consider a set of detected landmarks $\mathbb{L}=\{\left(\boldsymbol{l}_{j},c_{j},k_{j}\right)\mid j=1,\ldots,L\}$, where $\boldsymbol{l}_{j}$ is a 2D fiducial point, $c_{j}\in[0,1]$ is a confidence score provided by the detector, and k_{j} is the index of the corresponding vertex on the 3D surface. The distance between the projected landmarks and the one from $\mathbb{L}$ must be small. The objective function measuring the distance is defined as:

$$\mathcal{L}_{\text{lms}}\left(\mathcal{X}\right)=\frac{1}{L}\sum_{j=1}^{L}c_{j}\cdot\|\Pi_{k_{j}}\left(\mathcal{S}\left(\mathcal{X}\right)\right)-\boldsymbol{l}_{j}\|_{p}^{p} \tag{2.34}$$

where $\mathcal{S}\left(\mathcal{X}\right)$ is the reconstructed surface, and $\Pi_{k_{j}}\left(\cdot\right)$ denotes the projection of the k_{j} vertex of $\mathcal{S}$. Recently, Feng *et al.* proposed to constrain the length of projected segments defined between the facial landmarks to estimate proper opening of the upper and lower eyelid [Feng et al., 2021]. Considering the set of landmark pairs $\mathbb{P}=\{\left(\boldsymbol{l}_{j_{0}},k_{j_{0}},\boldsymbol{l}_{j_{1}},k_{j_{1}}\right)\mid j=1,\ldots,P\}$, the

distance between the projected pairs is defined as in:

$$\mathcal{L}_{\text{pair}}(\mathcal{X}) = \frac{1}{P}\sum_{j=1}^{P} \left\| \left(\boldsymbol{l}_{j_1} - \boldsymbol{l}_{j_0}\right) - \left(\Pi_{k_{j_1}}(\mathcal{S}(\mathcal{X})) - \Pi_{k_{j_0}}(\mathcal{S}(\mathcal{X}))\right) \right\|_p^p, \tag{2.35}$$

where $\boldsymbol{l}_{j_0}, \boldsymbol{l}_{j_1}$ are the detected facial landmarks, k_{j_0}, k_{j_1} denote the indexes of the corresponding vertices on the 3D surfaces, and $\Pi_{k_{j_0,j_1}}(\cdot)$ is the projection of the k_{j_0} and k_{j_1} vertices.

2.4.3 Regularization

Regularization is the last category of objective functions used in monocular 3D reconstructions. These cost functions act as soft constraints to ensure some properties or prior knowledge are preserved.

Since 3DMM is based on PDM, it has a natural probabilistic prior on the distribution of its parameters. Parameters of a model built with PCA follow a normal distribution $\mathcal{N}(0,1)$. Thus the first regularization term is to ensure the estimated parameters respect this criterion. The formulation of such prior is given in:

$$\mathcal{L}_{\text{prior}}(\mathcal{X}) = \sum_{j}^{d} \frac{w_j^2}{\sigma_j^2}, \tag{2.36}$$

where σ_j^2 is the variance linked to the j-th principal component, and w is a parameter of a given statistical model (*i.e. shape or appearance*) [Blanz and Vetter, 1999].

However, when too much weight is given to the statistical regularization, the reconstruction will be biased toward the mean face $\boldsymbol{w} \to \mathbf{0}$. To avoid such an issue, Jiang *et al.* proposed to minimize the Kullback-Leibler (KL) divergence between the empirical distribution of the parameters and the normal distribution [Jiang et al., 2021]. The regularization function is defined in:

$$\mathcal{L}_{\text{kl}}(\mathcal{X}) = \frac{1}{2}\sum_{j}^{d} \bar{\boldsymbol{\sigma}}_j^2 + \bar{\boldsymbol{\mu}}_j^2 + \log\left(\bar{\boldsymbol{\sigma}}_j^2\right) - 1, \tag{2.37}$$

where $\bar{\boldsymbol{\mu}}, \bar{\boldsymbol{\sigma}}^2$ are the estimated mean and variance of the parameter distribution. This is a similar approach to Genova *et al.* where they explicitly constraint the statistics of the distribution of the parameters (i.e. $\bar{\boldsymbol{\mu}} = 0$ and $\bar{\boldsymbol{\sigma}}^2 = 1$) [Genova et al., 2018].

Regularization can make use of the intrinsic properties of the problem being solved. For instance, the human face has bilateral symmetry. This strong prior helps to disentangle facial expression and lighting from the face albedo as shown by Gao *et al.* The albedo regularization

term is given by:

$$\mathcal{L}_{\text{sym}}(\mathcal{X}) = \|\mathcal{A}(\mathcal{X}) - \text{flip}(\mathcal{A}(\mathcal{X}))\|_1, \tag{2.38}$$

where $\mathcal{A}$ denotes the face albedo (*i.e. without illumination*) and $\text{flip}(\cdot)$ is an operation that flips the face albedo horizontally [Gao et al., 2020].

Moreover, the generated albedo $\mathcal{A}$ is not guaranteed to be in a valid pixel value range (*i.e. usually between 0 and 1*). Therefore, Romdhani and Vetter proposed further constraining the appearance model to improve the separation of illumination from the albedo by using a soft constraint [Romdhani and Vetter, 2005]. This constraint function is defined as:

$$\mathcal{L}_{\text{range}}(\mathcal{X}) = \sum_j c^j_{\mathcal{A}}(\mathcal{X})^2, \quad \text{with} c^j_{\mathcal{A}}(\mathcal{X}) = \begin{cases} \mathcal{A}_j(\mathcal{X}) - l & \mathcal{A}_j(\mathcal{X}) < l \\ \mathcal{A}_j(\mathcal{X}) - u & \mathcal{A}_j(\mathcal{X}) > u \\ 0 & \text{otherwise} \end{cases}, \tag{2.39}$$

where l, u are the lower and upper bounds that the appearance $\mathcal{A}$ has to lie within.

Regularization can also be applied to keep properties between the target image and the reconstruction. In the work of Deng *et al.*, they proposed to ensure the subject's identity is preserved by the reconstruction algorithm [Deng et al., 2019]. The regularization term is defined in terms of similarity as:

$$\mathcal{L}_{\text{id}}\left(\bar{\boldsymbol{I}}, \boldsymbol{I}, \mathcal{X}\right) = 1 - \frac{\left\langle \mathcal{G}\left(\bar{\boldsymbol{I}}\right), \mathcal{G}(\boldsymbol{I}) \right\rangle}{\|\mathcal{G}\left(\bar{\boldsymbol{I}}\right)\| \|\mathcal{G}(\boldsymbol{I})\|}, \tag{2.40}$$

where $\mathcal{G}(\cdot)$ is an identity embedding from a pre-trained face recognition network such as FaceNet [Schroff et al., 2015], $\bar{\boldsymbol{I}}$ is the generated image, $\boldsymbol{I}$ is the target image and $\langle \cdot, \cdot \rangle$ is the inner vector product.

Following this idea, Gecer *et al.* proposed to use intermediate feature representations from the face recognition network to increase the quality of the reconstruction [Gecer et al., 2019]. To this end, the suggested content loss is given in:

$$\mathcal{L}_{\text{con}}\left(\bar{\boldsymbol{I}}, \boldsymbol{I}\right) = \sum_j \frac{\|\mathcal{G}^j\left(\bar{\boldsymbol{I}}\right) - \mathcal{G}^j(\boldsymbol{I})\|_2}{H_j \cdot W_j \cdot C_j}, \tag{2.41}$$

where $\mathcal{G}^j(\cdot)$ denotes the j-th activation map from the face recognition network and H_j, W_j, C_j are the dimensions of the j-th feature map.

Other types of regularization have been proposed over the recent years, especially when the

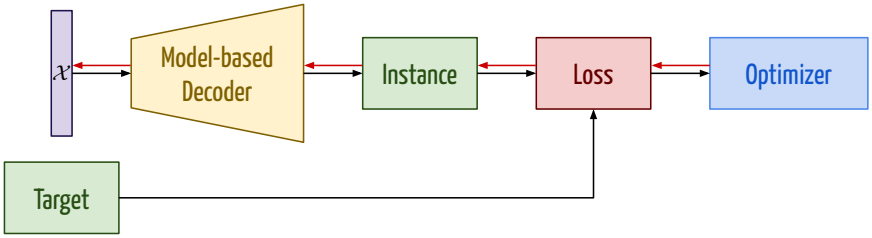


Figure 2.12 – Analysis-by-synthesis fitting

face albedo and geometry are learned simultaneously as the reconstruction process. This type of problem becomes highly ill-posed and requires more regularization. Usually, this regularization takes the form of smoothness prior on the geometry or the albedo. For instance, the l_2 norm of the mesh Laplacian has been used to regularize the shape deformation [Tewari and Kim, 2015].

2.5 Fitting Framework

Cost functions provide a way to measure the similarity of an instance of the model with respect to a target image (Section 2.4). Given a generative model and a sum of cost functions, the fitting strategy defines a method to estimate the correct set of parameters that best explains an image.

In this section, different types of fitting strategies used for 3D face reconstruction will be discussed. The three strategies covered are the classical *analysis-by-synthesis* for parametric models, the common *end-to-end* training for deep learning frameworks, and the *probabilistic* fitting orthogonal to the other methods.

2.5.1 Analysis-by-synthesis

The analysis-by-synthesis framework is the classical parameters estimation technique based on a differentiable image formation process. An instance of the 3D face model is generated with the current estimation of the parameters. This synthetic image is compared against the target image using a sum of objective functions. The residual error between them is back-propagated, and the parameters are updated and refined through a gradient descent step. This process of synthesis and analysis is repeated several times until convergence. In the end, the estimated parameters are the ones that were most likely used to synthesize the target image. Figure 2.12 illustrates the concept of an analysis-by-synthesis fitting strategy.

Because of the iterative nature of the process, it is slow. Thus once images and their corresponding parameters pairs $(\boldsymbol{I}_i, \mathcal{X}_i)$ have been established for a whole dataset, one can learn a regression function that maps the image space to the parameter space. This, in turn, will speed up the inference time.

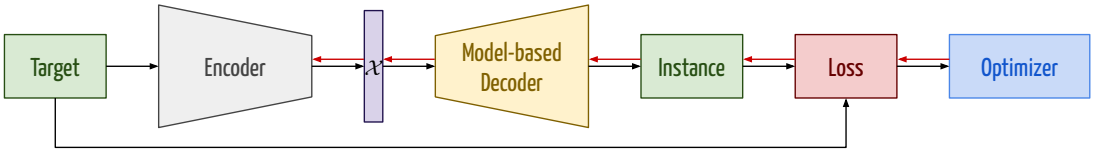


Figure 2.13 – End-to-end fitting

2.5.2 End-to-end

Over the recent years, deep learning-based methods have shown incredible capability in various domains such as classification or image generation. In the work of Tewari *et al.*, the fitting of 3DMM and the learning of a regression function mapping the image to the parameters are combined in a single optimization task [Tewari et al., 2017]. The proposed approach relies on an autoencoder-like architecture where the decoder is replaced by a parametric face model combined with a differentiable renderer. The encoder, responsible for predicting the parameters for a given image, is trained end-to-end thanks to gradient back-propagation. The whole process is illustrated in Figure 2.13, where the encoder parameters are updated by back-propagating the gradient of the residual error through the model-based decoder. The synthesized instance must be close to the input of the network. This method benefits from the speed and the robustness of deep learning-based models.

This formulation of 3DMM combined with network-based models opens up new possibilities that were not possible before. In the original method, only the encoder was trained. However, in recent work, the underlying face model in the decoder is allowed to be learned from the data [Tewari et al., 2019, Tran and Liu, 2018]. On the other hand, this flexibility introduces new challenges such as ensuring the model is learning only the attributes of the face and not external factors (*i.e. illumination, occluders*).

2.5.3 Probabilistic

The last fitting strategy is orthogonal to the gradient descent-based minimization methods presented earlier. The sampling-based fitting algorithm proposed by Schönborn *et al.* is a probabilistic approach based on Metropolis–Hastings (MH) algorithm [Schönborn et al., 2017, Egger et al., 2017]. The estimation of the parameters is conducted by performing a random walk in the parameter space. Due to the stochastic nature of the process, it is less susceptible to getting stuck in local minima, thus achieving high-quality reconstructions.

The fitting strategy is composed of two steps. The first one is the *proposal* step, where a new candidate is generated, followed by a *verification* step accepting the proposed candidate if it improves the quality of the reconstruction or rejecting it if not. More specifically, the proposed candidate is a parameter update over the current state $\mathcal{X} \rightarrow \mathcal{X}'$. The quality of this candidate is verified by evaluating its likelihood $\ell\left(\mathcal{X}' \mid \boldsymbol{I}\right)$ and determines if it is kept or rejected. This process of *proposal* - *verification* is repeated multiple times until convergence.

An attractive property of this minimization strategy is that no gradient is required to find the optimal set of parameters that explain a given image. Therefore the forward process synthesizing the image does not need to be differentiable. On the other hand, since the minimization is performed by randomly walking in the parameter space, it takes quite some time to reach the optimum.

2.6 Discussion

In the different sections of this chapter, each component of the standard 3D face reconstruction pipeline and fitting strategies have been introduced and discussed. Various technical solutions have been reviewed to solve the issues and limitations encountered in each stage of the reconstruction process.

However, not every research direction has been investigated in the scope of this thesis. The research presented in this thesis focuses on end-to-end deep learning-based reconstruction methods with partially fixed model-based decoders. In these conditions, ways to increase the robustness and the quality of the reconstructed face have been explored and will be presented and discussed in this thesis. The limiting factors of the methodology have been first identified, then ways that limit their impact or solve them entirely have been proposed. In this context, three limiting factors have been identified and explored.

The first aspect treated is the consistency of shape and appearance latent codes across head pose. When training a reconstruction network, samples are treated independently of each other. This approach does not provide any guarantee that two images of the same subject will produce the same face parameterization. The objective functions are applied at the image level and not across the samples of a batch. One approach to tackle this would be to carefully select the training data, ensuring multiple images per subject are available along with the information about the subject's identity. Then by meticulously selecting images to build the training batches, one can impose similarity constraints on the latent code that is shared across images for the same subject (*i.e. identity and albedo*). This technique has multiple flaws. First, the information about the identity of each subject on the images might not be available to all the data points in the training corpus. Second, it does not scale well well with the size of the training set. The selection of images of the same subject without introducing biases becomes difficult. Therefore we proposed an alternative way to enforce latent code consistency across head poses by using the generative face model of the decoder and modified network architecture. The general idea is to use the face model to generate random samples with the current face parameterization under different poses, then re-encode the synthesized image and impose similarity between the two predictions. The method will be discussed in more detail in Chapter 4.

The second axis of research concerns the impact of image resolution on reconstruction quality. As of today, the 3D face reconstruction is always done with high-resolution images. The face region will be downsampled to match the fixed input dimensions imposed by the network.

However, when the face region is smaller, it is first upsampled before feeding it to the neural network. This upsampling step can dramatically impact the 3D reconstruction quality, as neural networks are sensitive to image degradations [Dodge and Karam, 2016]. When performing reconstruction on low-resolution images, Mortazavian *et al.* proposed integrating the downsampling operation directly into the Point Spread Function (PSF) of the camera model [Mortazavian et al., 2012]. However, this technique is not applicable with differentiable renderers, which are a centerpiece in the reconstruction pipeline. Thus we propose building upon the recent advances in self-supervised learning to tackle the challenge of 3D reconstruction from low-resolution images. The idea is to consider the low-resolution images as an augmented version of the original one. Then constraints are applied to the feature space and the latent code to ensure that the network outputs the same representation independently of the size of the input image. The 3D reconstruction in a multi-resolutions setting will be covered in Chapter 5.

The last part investigates how one can recover the details of the human face, such as the wrinkles. The deformation of the face geometry is most of the time represented by a PDM. Because of the way classical PCA-based statistical models are built, all the tiny shape deformations (*i.e. wrinkles*) are lost in the process. Only the strongest axis of deformations are kept. The coarse geometrical information is the only part of the model. Therefore these details will not be recovered at reconstruction time. The general approach to this issue is to add the missing geometrical information in the appearance model. Explaining geometric displacements with appearance instead introduces aberrations in the rendering process and breaks the disentanglement of the facial attributes and the external factors such as the illumination. Inspired by computer graphics, we propose embedding facial details into displacement maps [Mikkelsen, 2008] and show how we can automatically recover these maps from the data without the need to handcraft them. This formulation will be discussed in detail in Chapter 6.

The next chapter will cover the experimental setup used across this thesis and establish a baseline reconstruction system used as a reference to assess the different contributions previously mentioned quantitatively. It will also include details on the data used during training, provide information about the training strategy and hyper-parameters, and discuss the evaluation protocols used to assess the quality of the reconstructions with respect to the face geometry and the facial appearance.

3 Experimental Setup

This chapter will describe and discuss the details of the experimental setup used in the scope of this thesis. The various aspects of the reconstruction pipeline will be detailed, as well as the evaluation protocols used to assess the quality of the reconstructions. This setup will serve as a baseline will allow us to evaluate the impact of our contributions quantitatively.

The subsequent sections will discuss, the architecture selected for the reconstruction network, the generative model used for the whole scene (*i.e. face, illumination, and pose*), the different loss functions used during the training, and explain the different protocols used to evaluate the quality of the reconstruction network.

3.1 Reconstruction Network Architecture

The architecture of the reconstruction network will be discussed in this section. The design follows the general idea proposed by Tewari *et al.* and uses an autoencoder-like approach [Tewari et al., 2017]. The whole system can be split into three components, an encoder extracting features from the image, multiple regression heads to predict the parameters of the generative model, and a decoder creating an instance of the model and converting it to an image. The fact that only the encoder and the regression heads are trained is an essential difference with respect to the standard formulation of the autoencoder. It allows injecting prior knowledge about the human face and external factors directly into the decoder, as discussed in Chapter 2. An overview of the whole system is given in Figure 3.1.

Encoder The encoder is composed of two blocks. First, a deep neural network or backbone projects the image into a feature space to extract relevant high-level semantic information to build upon. Two types of backbone have been used for the experiments carried out during this thesis, the ResNet18[1] [He et al., 2016a] and the B2 variant of the EfficientNetV2[2] [Tan and Le, 2021]. Both networks are based on a residual design that has shown good performance over

[1] Code/weights available at https://github.com/qubvel/classification_models
[2] Code/weights available at https://github.com/google/automl/tree/master/efficientnetv2

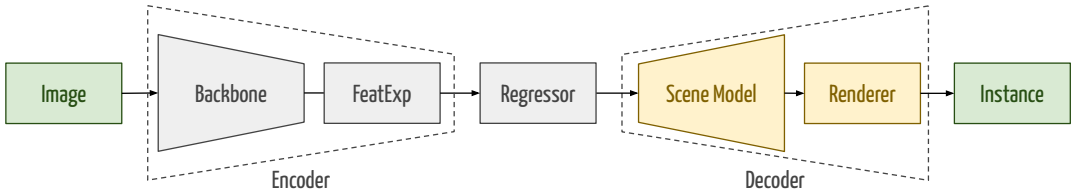


Figure 3.1 – Autoencoder-like reconstruction network architecture

different computer vision tasks (*i.e. classification, detection, and regression*). The networks have a similar number of parameters, around 11M variables. Three aspects have been taken into account to select an appropriate backbone: the model's capacity, the memory footprint, and the speed at training time. Both of the chosen backbones are a reasonable trade-off with respect to these criteria.

Next to the backbone is the Feature Expansion module. The idea is taken from the EfficientNet architecture, where the feature space is expanded before applying spatial pooling to increase the capacity of the feature space [Tan and Le, 2019]. The module comprises three layers: a 1×1 convolution layer, a batch normalization layer, and a global average pooling layer.

Regressors The regression module predicts a set of parameters $\mathcal{X}$ used by the generative scene model out of the features extracted by the encoder module. It is composed of multiple regression heads, where each one of them is responsible for predicting a single parameter of the model. Thus, the model learns a total of six parallel heads rather than a single regressor predicting parameters at once. This choice of design has shown better training stability during our experiments.

The architecture of each regression head is formed by a two layers MLP with a non-linear ReLU activation function for the first layer. The last layer has no activation function and directly regresses the parameter. The choice of two layers MLP instead of a single fully connected layer is motivated by experimental evidence. Experiments have shown that the capacity of the single-layer regressor was not enough. The complete configuration of the regression module is given in Table 3.1.

Decoder The decoder is responsible for creating the geometry and the color of a human face as a textured mesh out of the set of parameters $\mathcal{X}$ predicted by the regression module. The details of the creation of a 3D textured face will be given in Section 3.2. The 3D scene is then converted to an actual image through the differentiable rendering framework[3] of Laine *et al.* presented in Section 2.3. The decoding process defines only per-vertex facial attributes and not continuous value over the entire surface. Therefore, the rendering step only requires the *rasterization*, *interpolation*, and *antialiasing* modules. The *texture interpolation* module does not apply in this context.

[3] Code available at https://github.com/NVlabs/nvdiffrast

Table 3.1 – List of configurations for each regression head

Attributes	Symbol	Configuration
Identity	$\boldsymbol{w}^s$	Dense 256, ReLU, Dense 80
Expression	$\boldsymbol{w}^e$	Dense 256, ReLU, Dense 64
Appearance	$\boldsymbol{w}^t$	Dense 256, ReLU, Dense 80
Illumination	$\boldsymbol{w}^i$	Dense 256, ReLU, Dense 75
Rotation	$\boldsymbol{q}$	Dense 256, ReLU, Dense 3
Translation	$\boldsymbol{t}$	Dense 256, ReLU, Dense 3

3.2 Generative Scene Model

The geometric deformation of the face is represented by two PDMs, one for the identity and one for the expression, as in Equation 2.11. Therefore any surface $\mathcal{S}$ is generated by:

$$\mathcal{S}\left(\boldsymbol{w}^s, \boldsymbol{w}^e\right) = \bar{\boldsymbol{s}} + \boldsymbol{U}^s \boldsymbol{w}^s + \boldsymbol{U}^e \boldsymbol{w}^e,$$

where $\bar{\boldsymbol{s}} \in \mathbb{R}^{3N}$ is the mean shape composed of N vertices, $\boldsymbol{U}^s \in \mathbb{R}^{3N \times N_s}$ is the identity basis from BFM 2009[4] with $N_s = 80$ components [Paysan et al., 2009], $\boldsymbol{U}^e \in \mathbb{R}^{3N \times N_e}$ is the expressions basis[5] from [Guo et al., 2019] with $N_e = 64$ components and $\boldsymbol{w}^s, \boldsymbol{w}^e$ are the geometric parameters.

A PDM also parameterizes the appearance or albedo of the face as in Equation 2.4. Thus the face appearance $\mathcal{A}$ is given by:

$$\mathcal{A}\left(\boldsymbol{w}^t\right) = \bar{\boldsymbol{a}} + \boldsymbol{U}^t \boldsymbol{w}^t,$$

where $\bar{\boldsymbol{a}} \in \mathbb{R}^{3N}$ is the mean appearance, $\boldsymbol{U}^t \in \mathbb{R}^{3N \times N_t}$ are the albedo basis from BFM 2009 with $N_t = 80$ components, and $\boldsymbol{w}^t$ are the appearance parameters.

The final observed color depends on a combination of the face albedo and the lighting of the scene. Following the discussion in Section 2.2, the external illumination is modeled using SH while assuming a pure Lambertian reflectance, similar to previous work. However, following the argumentation of [Dib et al., 2021b], more SH bands are used compared to the classical methods. Moreover, the light is not assumed to be monochromatic. Thus each channel has its own illumination parameters. The final color $\mathcal{C}$ of the j-th vertex is defined as:

$$\mathcal{C}_j\left(\mathcal{A}_j, \boldsymbol{n}_j, \boldsymbol{w}^i\right) = \mathcal{A}_j \odot {\boldsymbol{w}^i}^\top \boldsymbol{\phi}\left(\boldsymbol{n}_j\right), \tag{3.1}$$

where $\odot$ denotes the element-wise product between two vectors, $\mathcal{A}_j$ and $\boldsymbol{n}_j$ are the albedo and

[4] Available at https://faces.dmi.unibas.ch/bfm
[5] Available at https://github.com/Juyong/3DFace

the normal of the surface of the j-th vertex, $\boldsymbol{w}^i = \left[\boldsymbol{w}_r^i, \boldsymbol{w}_g^i, \boldsymbol{w}_b^i\right] \in \mathbb{R}^{N_i \times 3}$ are the illumination coefficients for each color channel with $N_i = 25$ components and $\boldsymbol{\phi} : \mathbb{R}^3 \rightarrow \mathbb{R}^{N_i}$ is a vector of spherical harmonics computed from the normal of the surface.

Following the formalism of standard rendering pipelines introduced in Section 2.3, the camera model is defined with two transformation matrices, $\boldsymbol{M}_{\text{proj}}$ and $\boldsymbol{M}_{\text{model}}$, for the projection from 3D to 2D and the rigid transformation applied to the face. The camera is assumed to be placed at the origin of the world coordinate system, and only the object is moved around it. It avoids the ambiguity of defining what part has moved, the object or the camera. The projection matrix $\boldsymbol{M}_{\text{proj}}$ uses the perspective model to model realistic projection (*i.e. pinhole camera model*) with a field of view of 30°, an aspect ratio of 1, the near plane is at 1, and the far plane is at 1000. The model transform is a composition of rotation and translation and is given by:

$$\boldsymbol{M}_{\text{model}}\left(\boldsymbol{q}, \boldsymbol{t}\right) = \begin{bmatrix} \boldsymbol{R}\left(\boldsymbol{q}\right) & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{bmatrix}, \tag{3.2}$$

where $\boldsymbol{q}$ denotes the parameterization of the rotation in axis-angle format and $\boldsymbol{t}$ is the 3D translation defining the position of the face in the world space.

Combining everything, the per-vertex position in the eye space and color attributes of the j-th vertex are computed with the scene generative model $\mathcal{H}$ given in:

$$\mathcal{H}_j\left(\mathcal{X}\right) = \left(\mathcal{C}_j\left(\mathcal{A}_j, \boldsymbol{n}_j, \boldsymbol{w}^i\right), \boldsymbol{M}_{\text{model}}\left(\boldsymbol{q}, \boldsymbol{t}\right)\left[\mathcal{S}_j\left(\boldsymbol{w}^s, \boldsymbol{w}^e\right), 1\right]\right), \tag{3.3}$$

where $\mathcal{X} = \{\boldsymbol{w}^s, \boldsymbol{w}^e, \boldsymbol{w}^t, \boldsymbol{w}^i, \boldsymbol{q}, \boldsymbol{t}\}$ is the full set of parameters. The regression modules predict these parameters out of the visual representation of the image extracted with the encoder. For our experiments, the dimensions of the each parameter is set to $\boldsymbol{w}^s \in \mathbb{R}^{80}$, $\boldsymbol{w}^e \in \mathbb{R}^{64}$, $\boldsymbol{w}^t \in \mathbb{R}^{80}$, $\boldsymbol{w}^i \in \mathbb{R}^{3\cdot 25}$, $\boldsymbol{q}^s \in \mathbb{R}^3$, and $\boldsymbol{t}^s \in \mathbb{R}^3$.

3.3 Loss functions

The loss functions used to constraint the optimization problem is a composition of the objective functions presented earlier in Section 2.4. It is composed of multiple data terms as well as regularization terms to stay in the range of valid solutions for the underlying statistical models. The complete objective function to be minimized during the training stage is given in a compact form in:

$$\mathcal{L}^c\left(\bar{\boldsymbol{I}}, \mathcal{X}\right) = \lambda_{\text{ph}}\ell_{\text{ph}} + \lambda_{\text{silh}}\ell_{\text{silh}} + \lambda_{\text{lms}}\ell_{\text{lms}} + \ell_{\text{reg}}. \tag{3.4}$$

The most influential part is the photometric error ℓ_{ph} forcing a dense photometric alignment

between the reconstruction and the image. The distance is directly computed on the image plane in *RGB* colorspace. It is based on the work of Egger *et al.* to avoid the shrinking effect that affects 3D reconstruction. It is formulated as a likelihood combined with an image pyramid as suggested by Henderson and Ferrari [Egger et al., 2018, Henderson and Ferrari, 2019]. The error between the synthesized image $\boldsymbol{I}(\mathcal{X})$ and the target image $\bar{\boldsymbol{I}}$ is given by:

$$\ell_{\text{ph}}\left(\bar{\boldsymbol{I}}, \mathcal{X}\right) = \sum_{l}^{L} \frac{1}{\left|\mathcal{F}^{l} \cap \mathcal{M}^{l}\right|} \sum_{p \in \mathcal{F}^{l} \cap \mathcal{M}^{l}} \frac{2^{l}}{\sigma_{\text{ph}}} \| \boldsymbol{I}_{p}^{l}(\mathcal{X}) - \bar{\boldsymbol{I}}_{p}^{l} \|_{1}^{1} - \log\left(h_{\mathcal{B}}\left(\boldsymbol{I}_{p}^{l}(\mathcal{X})\right)\right), \tag{3.5}$$

where $l = \{0, \ldots, L-1\}$ indicates the level in the pyramid, $\mathcal{F}^{l}$ defines the region covered by the projected geometry at level l (*i.e. foreground*), $\mathcal{M}^{l}$ denotes the segmentation mask indicating the probability of a pixel being explainable by the face model, σ_{ph}^{2} is the approximated variance of the residual photometric error and $h_{\mathcal{B}}$ is an image-based histogram modeling the likelihood of a pixel being part of the background.

To help with the initial coarse alignment of the generated surface with the target image, a silhouette loss ℓ_{silh} based on the Intersection over Union (IoU) distance is used similar to [Chen et al., 2019b]. The cost function measures the misalignment between the foreground region $\mathcal{F}$ defined by the generative model (*i.e. projected geometry*) and the segmentation mask $\mathcal{M}$ and is computed as:

$$\ell_{\text{silh}}(\mathcal{M}, \mathcal{F}) = 1 - \frac{|\mathcal{M} \cap \mathcal{F}|}{|\mathcal{M}| + |\mathcal{F}| - |\mathcal{M} \cap \mathcal{F}|}, \tag{3.6}$$

where $|\cdot|$ denotes the cardinality of the mask. To further help with geometric alignment, the distance between the projected facial landmarks and detected ones must be small. This constraint is enforced by the landmark loss ℓ_{lms} defined earlier in Equation 2.34. For clarity purposes, it is redefined in:

$$\ell_{\text{lms}}(\mathcal{X}) = \frac{1}{F} \sum_{j=1}^{F} c_{j} \cdot \| \Pi_{k_{j}}(\mathcal{S}(\mathcal{X})) - \boldsymbol{l}_{j} \|_{1}^{1},$$

where $\boldsymbol{l}_{j}$ is a detected landmark, $c_{j} \in [0,1]$ indicates the quality of the detection, $\Pi_{k_{j}}$ is a function projecting the k_{j}-th vertex to the image plane, and $\mathcal{S}(\mathcal{X})$ is the reconstructed surface (*i.e. geometry in camera space*).

These three objective functions are the data terms used in the experiments unless specified. Regularization terms are added to them to produce valid reconstruction and stay in the span of the correct solution. The details of the regularization term are defined in:

$$\ell_{\text{reg}}(\mathcal{X}) = \lambda_{\text{stats}}\left(\lambda_{\text{id}}\ell_{\text{kl}}^{\text{id}} + \lambda_{\text{exp}}\ell_{\text{kl}}^{\text{exp}} + \lambda_{\text{tex}}\ell_{\text{kl}}^{\text{tex}}\right) + \lambda_{\text{range}}\ell_{\text{range}} + \lambda_{\text{sym}}\ell_{\text{sym}}. \tag{3.7}$$

The first part enforces that the parameters of the statistical models stay within a valid range, meaning drawn from a normal distribution $\mathcal{N}(0,1)$ as expected by PCA-based models. The KL-Divergence loss from Equation 2.37 ensures the validity of the parameters. It is applied for each coefficient of the 3DMM, the identity, the expression, and the appearance. The loss is defined as:

$$\ell_{\text{kl}}^{m}(\mathcal{X}) = \frac{1}{2}\sum_{j}^{d} \bar{\boldsymbol{\sigma}}_j^2 + \bar{\boldsymbol{\mu}}_j^2 + \log\left(\bar{\boldsymbol{\sigma}}_j^2\right) - 1$$

where $m = \{\text{id}, \text{exp}, \text{tex}\}$ denotes an attribute of the face model, $\bar{\boldsymbol{\mu}}^2, \bar{\boldsymbol{\sigma}}^2$ are the empirical mean and variance of the corresponding parameter distribution.

The last two regularization terms are present to help to estimate accurate illumination from the observed color (*i.e. combination of albedo and light*). The first term, the albedo range loss ℓ_{range}, imposes that the values of the generated appearance are in the correct range for an image. It acts as a soft constraint and reuses the formulation given in Equation 2.39:

$$\ell_{\text{range}}(\mathcal{X}) = \sum_{j}^{N} c_{\mathcal{A}}^{j}(\mathcal{X})^2, \quad \text{with} \quad c_{\mathcal{A}}^{j}(\mathcal{X}) = \begin{cases} \mathcal{A}_j(\mathcal{X}) - l & \mathcal{A}_j(\mathcal{X}) < l \\ \mathcal{A}_j(\mathcal{X}) - u & \mathcal{A}_j(\mathcal{X}) > u \\ 0 & \text{otherwise} \end{cases},$$

where $c_{\mathcal{A}}^{j}$ is a thresholding operator acting on the generated appearance $\mathcal{A}$ for the j-th vertex, and l, u are the lower and upper limits of the values the appearance can take. The last regularization enforces the bilateral symmetry of the facial appearance is preserved through the reconstruction process. The symmetry loss ℓ_{sym} from Equation 2.38 is used for this purpose:

$$\ell_{\text{sym}}(\mathcal{X}) = \|\mathcal{A}(\mathcal{X}) - \text{flip}(\mathcal{A}(\mathcal{X}))\|_1,$$

where $\text{flip}(\cdot)$ is a function flips the generated appearance $\mathcal{A}$ horizontally.

To summarize, the loss function $\mathcal{L}^c\left(\bar{\boldsymbol{I}}, \mathcal{X}\right)$ is composed of one appearance-based loss ℓ_{ph}, two geometry-based losses $\ell_{\text{silh}}, \ell_{\text{lms}}$ together with regularization losses based on prior knowledge either from statistical models or property of human anatomy.

3.3.1 Training configuration

When starting to train the reconstruction network, the backbone network is initialized using weights pre-trained on ImageNet [Russakovsky et al., 2015]. Moreover, the last layer of each regression head in the regression module is initialized such that the mean face is placed in front of the camera at the beginning. Special care is taken to ensure the scene is lit when

starting the training phase.

The reconstruction network is trained by minimizing the loss function $\mathcal{L}^c\left(\bar{\boldsymbol{I}}, \mathcal{X}\right)$ using the Adam optimizer [Kingma and Ba, 2015] with a batch size of $b = 32$ and a piecewise constant learning rate schedule with an initial value of $lr = 1e^{-4}$ for about 400K iterations. Figure 3.2 shows the details of the learning rate schedule used while training the network. At the beginning of the optimization, the learning rate is linearly increased from 1×10^{-6} to 1×10^{-4} over the first 10k steps, as suggested in [Liu et al., 2020]. The complete set of hyper-parameters is provided in Table 3.2.

Table 3.2 – Hyper-parameters

Symbol	Value	Description
λ_{ph}	0.4	Photometric contribution
λ_{silh}	5.0	Silhouette contribution
λ_{lms}	0.45	Landmarks contribution
λ_{stats}	1×10^{-3}	Statistical prior contribution
λ_{id}	1.25	Identity prior contribution
λ_{exp}	1.0	Expression prior contribution
λ_{tex}	1.0	Albedo prior contribution
λ_{range}	1000	Albedo range importance
λ_{sym}	50	Albedo symmetry importance
σ_{ph}	0.043†	Variance of the photometric residue
L	3	Number of level in pyramid
l	0.0	Appearance lower bound
u	1.0	Appearance upper bound
lr	-	Learning rate
b	32	Batch size

† Taken from [Egger et al., 2018]

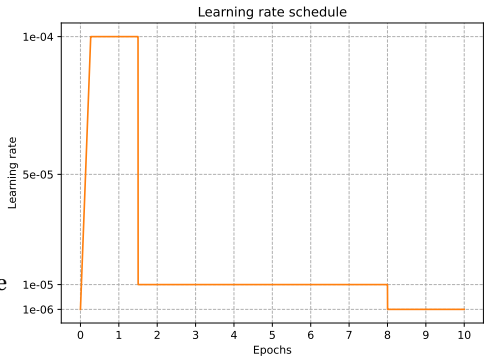


Figure 3.2 – Learning rate schedule

3.4 Datasets

The reconstruction network is trained on images gathered from multiple well-known datasets such as the CelebA[6] [Liu et al., 2015], 300W-LP[7] [Zhu et al., 2016], and VGGFace2[8] [Cao et al., 2018] as others commonly do it.

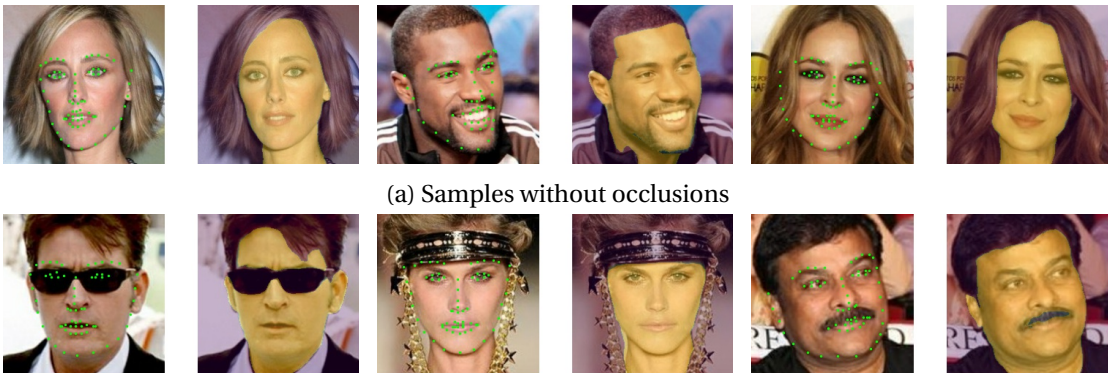
These images can not be used directly and need to be preprocessed before training the reconstruction network. The process is similar for each image of the datasets. The first step consists of finding the face bounding box using the S^{3}FD detector[9] from Section 1.1.2. An extra margin of 30% is added to the detected region to guarantee the whole face is visible and not cropped in the middle. From this extended bounding box, the image patch is then cropped and resized to 224×224 to match the dimensions imposed by the backbone network.

[6] Available at https://mmlab.ie.cuhk.edu.hk/projects/CelebA.html

[7] Available at http://www.cbsr.ia.ac.cn/users/xiangyuzhu/projects/3DDFA/main.htm

[8] Available https://github.com/ox-vgg/vgg_face2. However, at the time of the writing of this thesis, the website is offline.

[9] Code/weights available at https://github.com/sfzhang15/SFD



(a) Samples without occlusions

(b) Samples with occlusions

Figure 3.3 – Illustration of training samples where each pair shows the target image with its detected landmarks overlaid on top and the corresponding segmentation mask.

The second step is responsible for finding the facial landmarks' position using the FAN[10] method discussed in Section 1.2.2. The detected landmarks are the 2D projections in the image space of the 3D landmarks, referred to as 2.5D landmarks. Thus no anatomical aberrations will happen with the landmarks located on the chin. The alignment network is run twice with a slightly shifted bounding box, and an image with non-matching landmarks is discarded, as advised by [Feng et al., 2021].

The last step creates the segmentation mask $\mathcal{M}$ indicating which pixels are explainable by the 3DMM. The mask is produced with BiSeNet from Section 1.3.1, followed by a Pyramid Scene Parsing Network (PSPNet) trained to segment skin color applied only to the region labeled as *skin* by the semantic segmentation [Zhao et al., 2017]. This two steps segmentation allows handling the presence of facial hair carefully.

After this preprocessing step, the training set contains around 1.2M images. The validation set is composed of 50k images randomly sampled from the validation partition of the VGGFace2 dataset. Moreover, no special care has been taken to balance the sets in terms of head pose, ethnicity, or illumination conditions. Examples are shown in Figure 3.3, with their detected landmarks overlaid on top and their corresponding segmentation mask next to them. One can notice that occluders such as glasses, facial hairs, or accessories are not part of the segmentation mask, and face regions are well defined.

3.5 Evaluation Protocols

Once the reconstruction network is trained, the model is evaluated regarding each attribute, specifically the shape or *geometry* and the color or *texture* of a human face. Both of these tasks have their challenges.

[10] Code/weights available at https://github.com/1adrianb/face-alignment

Many face-related datasets do not provide 3D ground truth, mainly because of the difficulty of collecting such data (*i.e. requires complicated scanners or camera setup*). However, over recent years, some work has been done in this direction. Thus the evaluation of the shape reconstruction is conducted on the MICC[11] Florence 3D face [Bagdanov et al., 2011], and the FaceWarehouse[12] datasets [Cao et al., 2014]. Regarding the texture, to our knowledge, no dataset provides true albedo and illumination data. The main reason behind this lack of data is again the complexity of collecting them. Therefore the evaluation is done only for the combination of the albedo and the illumination, meaning the observed texture.

In the subsequent sections, the evaluation protocols for each modality will be presented and discussed. First, the geometric aspect will be covered, followed by the texture of the 3DMM.

3.5.1 Geometry

The MICC dataset contains data from 53 subjects associated with their corresponding 3D scan in neutral pose (*i.e. no expression*). Each subject has been recorded over three partitions, *cooperative*, *indoor*, and *outdoor*. For the cooperative setup, the subject is recorded with an HD camera at four levels of zooms under controlled illumination. The candidate has been instructed to move its head to generate out-of-plane rotations. The indoor setup records the subject with a lower resolution camera in an office at three zoom levels while having spontaneous behavior. In the outdoor setup, the subject is recorded outside at three levels of zoom while behaving spontaneously. Being outside increases the difficulty of the partition because of the uncontrolled light conditions and the presence of shadows. Figure 3.4 shows some samples for each partition used to evaluate the model. One can notice that the range of head motion is extensive in the first partition. Moreover, the image quality degrades quite a lot across all the partitions making the evaluation task challenging.

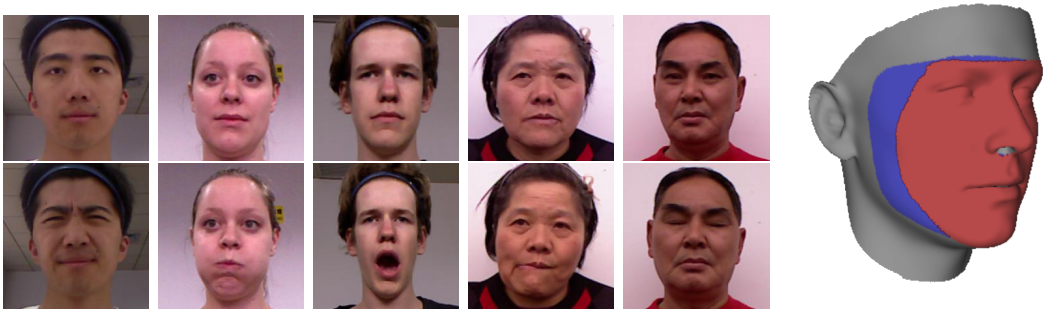
Following the protocol of Deng *et al.*, each frame of a video sequence for a given subject is first reconstructed then aggregated by averaging them together. The averaged reconstructed surface is considered the final reconstruction for the video [Deng et al., 2019]. It is important to note that the expression coefficients are set to zero to generate expressionless geometry. The ground truth scan is cropped to 95mm around the nose tip following [Genova et al., 2018] and aligned to the aggregated reconstruction using ICP with isotropic scale. The distance between the two surfaces is measured using the *point-to-plane* distance.

The evaluation on the FaceWarehouse dataset follows the protocol of [Tewari et al., 2018]. There are 180 meshes (*i.e. 9 identities, 20 expressions*) reconstructed from frontal images. Similar to the MICC dataset, the reconstructed surface is aligned to the ground truth with ICP, including isotropic scaling. The distance between the two is then measured with the point-to-point metric. Two regions of the face, illustrated in Figure 3.5b, are evaluated to assess the quality of the reconstruction. The larger region, in blue, includes more cheek area

[11] Available upon request at https://www.micc.unifi.it/resources/datasets/florence-3d-faces
[12] Available upon request at http://kunzhou.net/zjugaps/facewarehouse

(a) Cooperative (b) Indoor (c) Outdoor

Figure 3.4 – Image samples from the MICC Florence 3D face dataset used to assess the quality of the reconstructed geometry (*identity only*).



(a) Image samples with neutral face (*top*) and facial expression (*bottom*).

(b) The two regions used for evaluation.

Figure 3.5 – Image samples from the FaceWarehouse dataset used to assess the quality of the reconstructed geometry (*identity* + *expressions*).

for a more realistic comparison with the true mesh. One benefit of this evaluation set is its variety of facial expressions and ethnicity, as shown in Figure 3.5a. However, it is important to note that the number of samples is relatively small.

3.5.2 Texture

The evaluation of the texture generated by the reconstruction network through the 3DMM is quantified using the protocol proposed by Genova *et al.* The idea is to compare the two embeddings produced by a face recognition network on the target and reconstructed image and measure how similar they are [Genova et al., 2018]. Following [Parkhi et al., 2015], for every image in the evaluation dataset, the cosine similarity of the corresponding face embedding is computed between the input image and the synthesized one. If both embeddings are close, it

means the 3DMM can fool the recognition network. Thus the quality of the reconstruction is good enough. On the other hand, if the embeddings are not similar, it indicates that the network can discriminate between the true image and the rendered one. This implies the reconstruction network does not do a good job.

More specifically, the distribution of the similarity distance computed from all pairs of images in the LFW[13] dataset [Gary B. Huang et al., 2008], split into same-person and different-person, is used as an indicator of the reconstruction quality. As the face recognition network is not perfect either, an upper bound distribution is defined by measuring the similarity between real image pairs (*i.e. same-person and different-person*). The operation is then repeated for pairs composed of one real image and one synthesized image. The difference between the two distributions indicates the reconstruction quality.

3.6 Results

The evaluation results of both attributes using the protocols defined in Section 3.5 will be presented and discussed. This will serve as a baseline to compare against when evaluating our contributions. Qualitative results are shown in Figure 3.6, where images from the test partition of the VGGFace2 dataset are reconstructed using the ResNet18-based reconstruction network. The network can regress high-quality pose, shape, expression, albedo, and illumination across a broad range of conditions from a single image.

The quality of the different reconstructed components will be assessed in the next sections, Section 3.6.1 will cover the geometry, and Section 3.6.2 will examine the estimated albedo and illumination.

3.6.1 Geometry

The first reported results assess the reconstruction quality of neutral faces in various environments with the MICC dataset. The mean point-to-plane reconstruction error for each partition, measured in millimeters, and the average root mean square error (RMSE) are reported in Table 3.3. Moreover, the type of statistical shape model used to reconstruct the surface is indicated to have a fair comparison.

The proposed baseline is on par with other methods of Genova *et al.* and Deng *et al.* Both backbones produce reconstructions of similar quality with slightly better performance for the solution using the EfficientNetV2 encoder. However, it is still far from Gecer *et al.* The precise reasons behind the large gap are hard to define. However, one possible cause could be that the statistical models they used are closer to realistic distributions (*i.e. cover a more extensive range of variations*) since they are learned in a broader training set. The statistical shape model used in their experiments, namely the LSFM, is built out of 10,000 scans, and

[13] Available at http://vis-www.cs.umass.edu/lfw

(a) (b) (c) (d) (e)

Figure 3.6 – Qualitative results from the test partition of the VGGFace2 dataset generated from the ResNet18-based reconstruction network. (a) Target, (b) Reconstruction, (c) Shape, (d) Albedo, (e) Shading

Table 3.3 – Average point-to-plane reconstruction error (mm) across all subjects of the MICC dataset.

Method	Model	Cooperative		Indoor		Outdoor	
		Mean	RMSE	Mean	RMSE	Mean	RMSE
[Tran et al., 2016]†	BFM09	1.93 ± 0.27	-	2.02 ± 0.25	-	1.86 ± 0.23	-
[Genova et al., 2018]	BFM17	1.50 ± 0.13	-	1.50 ± 0.11	-	1.48 ± 0.11	
[Deng et al., 2019]	BFM09	-	$\mathbf{1.66 \pm 0.52}$	-	1.66 ± 0.46	-	1.69 ± 0.53
[Gecer et al., 2019]	LFSM	$\mathbf{0.95 \pm 0.11}^{\star}$	-	$\mathbf{0.94 \pm 0.11}^{\star}$	-	$\mathbf{0.94 \pm 0.11}^{\star}$	
ResNet18	BFM09	1.30 ± 0.31	1.73 ± 0.48	1.25 ± 0.24	1.64 ± 0.36	1.35 ± 0.25	1.79 ± 0.38
EfficientNetV2-B2	BFM09	1.27 ± 0.29	1.69 ± 0.49	1.21 ± 0.24	$\mathbf{1.59 \pm 0.36}$	1.24 ± 0.21	$\mathbf{1.64 \pm 0.33}$

† Results taken from [Genova et al., 2018]
⋆ p-value < 0.05

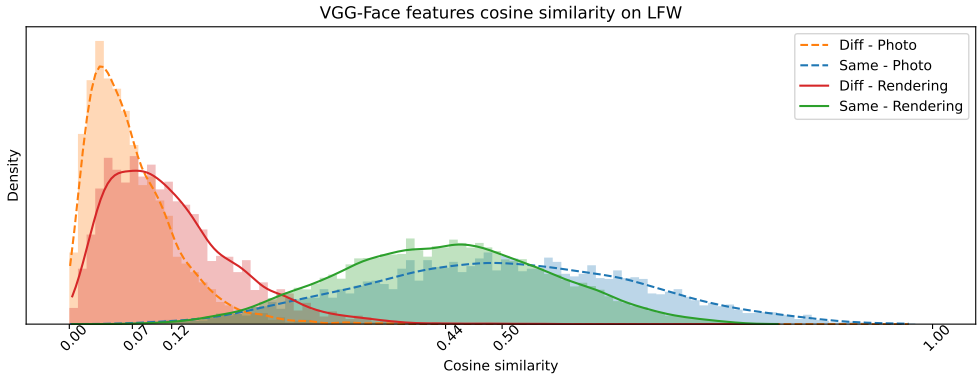
Table 3.4 – Mean point-to-point reconstruction error (mm) on 180 meshes of 9 subjects from FaceWarehouse. The suffix *-F* and *-C* denote the *fine* and *coarse* results of [Tewari et al., 2018].

Method	Model	Regions	
		Small	Large
[Tewari et al., 2018]-C	BFM09	2.03 ± 0.53	-
[Tewari et al., 2018]-F	BFM09†	1.84 ± 0.38	2.0‡
[Deng et al., 2019]	BFM09	**1.81**	**1.91**
ResNet18	BFM09	1.90 ± 0.46	2.19 ± 0.48
EfficientNetV2-B2	BFM09	1.93 ± 0.48	2.15 ± 0.50

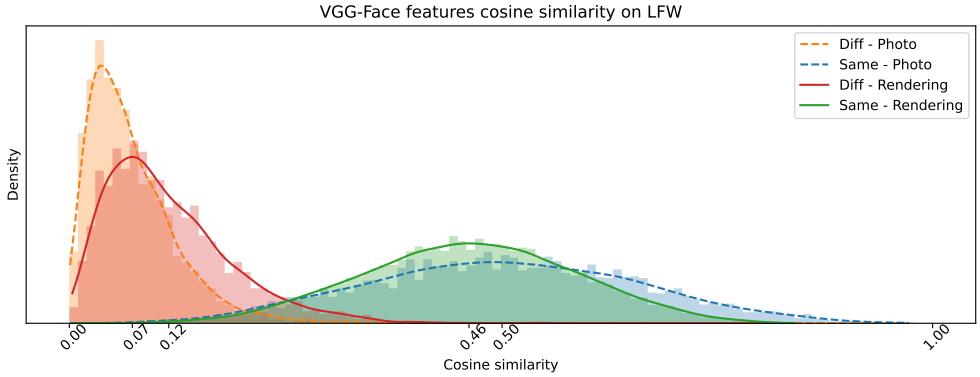
† Model is augmented with corrective basis
‡ Result taken from [Deng et al., 2019]

the gan-based appearance model is learned from 10,000 high-resolution texture maps. In contrast, the BFM is constructed only from only 200 scans. This could explain the significant gap observed while evaluating on the MICC dataset.

Subjects from the FaceWarehouse dataset are used to assess the quality of the reconstructed surface when expressions are present on the input image. The reconstructed meshes are first re-parameterized to the 60,000 vertices topology of Tewari *et al.* using nonrigid registration. The ground truth meshes from the FaceWarehouse dataset are also subdivided to increase the topology density. Once all surfaces have been updated, a 3D similarity transform is applied to align the ground truth and the reconstructions using pre-computed point-to-point correspondences from [Tewari et al., 2018]. The mean closest point-to-point error is calculated as the distance between both surfaces. Table 3.4 shows the average point-to-point error computed with the evaluation code provided by [Deng et al., 2019]. Both baselines perform similarly to the methods of Tewari *et al.* when considering only the inner part of the face (*i.e. the small region*). Compared to Deng *et al.*, the baselines are not far off for the inner part of the face. However, when considering the larger area, the error is increased significantly. It indicates that the jawline region is not well reconstructed by the proposed reference system.



(a) ResNet18



(b) EfficientNetV2-B2

Figure 3.7 – Cosine similarity distributions of VGG-Face embeddings

3.6.2 Texture

The results of the color evaluation on the LFW dataset for both backbones are given in Figure 3.7. Cosine similarity is measured between pairs of embeddings extracted with a face recognition network (*i.e. VGG-Face*). Each pair is formed either with images of the same subject or with images of different subjects. The dashed lines show distributions estimated using pairs of photo images (*i.e. photo-to-photo*. In contrast, solid lines indicate distributions computed with pairs of photo and synthesized images (*i.e. photo-to-rendering*). Ideally, the orange distribution should be close to zero as it represents the cosine distance between two different subjects, and the blue distribution should be close to one as it is from the same subject. As the face recognition network is not perfect, both distributions are flattened and shifted toward the middle. The photo-to-rendering distributions are there to assess the quality of the reconstructed texture and ideally should overlap the photo-to-photo distributions, meaning the reconstruction network is fooling the face recognition network. The plots show that both backbones performed reasonably well as the distributions overlap and have similar mean cosine distances (*i.e. shown on the horizontal axis*).

Table 3.5 – Average cosine distance for the photo-to-rendering pairs of the LFW dataset.

Method	Model	LFW	
		Same	Different
LFW - *Data*	-	0.50	0.07
[Tran et al., 2016]	BFM09	0.16†	-
[Genova et al., 2018]	BFM17	0.37	-
[Gecer et al., 2019]	GAN-based	**0.5**	-
ResNet18	BFM09	0.44	0.12
EfficientNetV2-B2	BFM09	0.46	0.12

† Results taken from [Genova et al., 2018]

The average cosine distance for the photo-to-rendering pairs is given in Table 3.5 alongside previously reported results from alternate reconstruction architectures. The first entry indicates the average cosine distance for the photo-to-photo pairs and is provided as an indicator of the target values. Both of the backbones performed similarly well and output distributions close to the photo-to-photo one. The quality of our reconstruction is close to the results of Gecer *et al.*, even with a simpler appearance model. This may be explained by using an SH-based illumination model instead of the Phong model used in their experiments and the work of Genova *et al.*

3.7 Summary

In this chapter, we have proposed in Section 3.1 a CNN-based architecture composed of an encoder and a model-based decoder to tackle the monocular face reconstruction challenge. Section 3.2 introduced the generative scene model used to synthesize 3D faces. The details of the training setup are given in Section 3.3, and the insights on the type of data used for training and validation are provided in Section 3.4. Finally, the different evaluation protocols are given in Section 3.5, and the performances of our baseline are given in Section 3.6.

The proposed baseline performs on par with previously published reconstruction methods for the identity. In the presence of facial expressions, the proposed reference system has issues correctly estimating the proper facial geometry, as shown by the evaluation on the FaceWarehouse. The reconstruction system does not correctly estimate the jawline region. Moreover, our performances are far from the reported results of Gecer *et al.* In their work, they have been using the LFSM shape model instead of the standard BFM. This highlights an important limiting factor of the model-based reconstruction approach. If the statistical model is not covering the whole deformation distribution, it will significantly impact the reconstruction quality.

Regarding the facial appearance attribute, our baseline performs similarly to other published methods and almost achieves performance close to the GAN-based texture model presented in [Gecer et al., 2019]. Acquiring the true albedo of a human face is not a trivial task and

requires a complex acquisition setup and processing pipeline. Therefore residual information from external parameters can still be part of the statistical model of the facial appearance. Such remaining information lead to photometric aberrations and can induce the wrong estimation of the facial attributes.

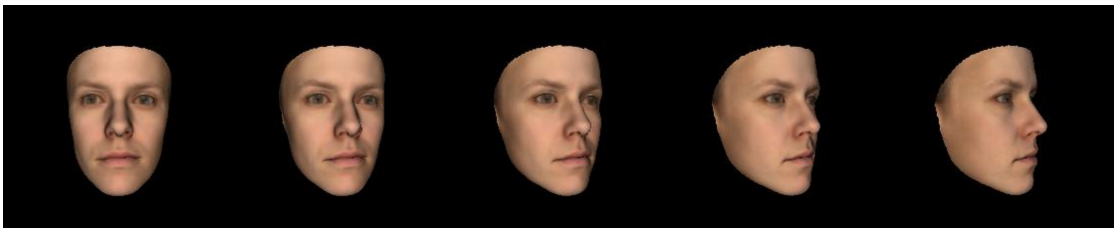
4 Cross-Pose Consistency

4.1 Introduction

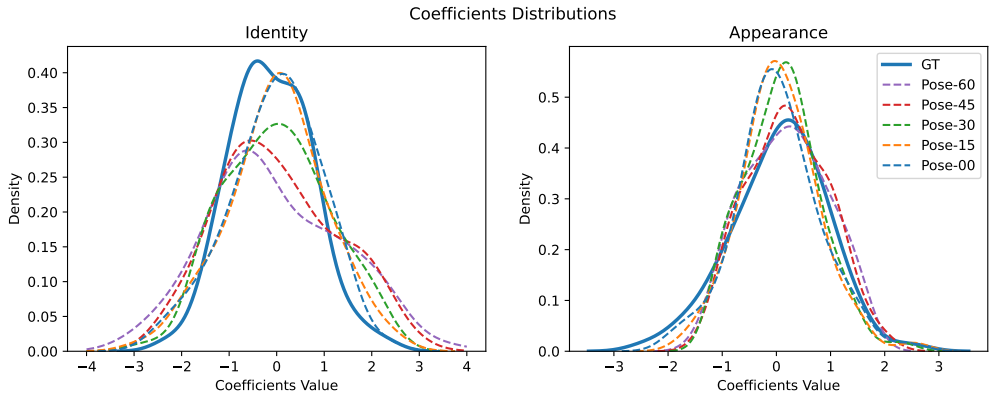
By design, the *analysis-by-synthesis* or *end-to-end* fitting strategies estimate the parameters of a model that best explains an image target. Moreover, since no ground truth is available, it is impossible to tell whether the estimated coefficients are correct or not. Therefore the reconstructed face may appear correct even though it is not the proper geometry or appearance.

When training a reconstruction network using an end-to-end strategy, each sample within a batch is treated independently and equally because most of the objective functions are applied at the sample level and not across the batch samples (*i.e. batch level*). This approach does not guarantee that the network will produce the same face parameterization for images of the same subject. This observation is a significant drawback for the reconstruction network as it should preserve some facial attributes between images of the same subject and across different poses. This phenomenon of parameter dissimilarity across head pose is depicted in Figure 4.1 with a synthetic toy example. A random instance of the BFM is rotated from left to right around the vertical axis by steps of 15°. The distributions of the identity and appearance parameters predicted by the reconstruction network (*i.e. ResNet18-based*) are shown with a kernel density estimate plot in Figure 4.1a. The true distribution is displayed with a thick blue line and serves as a reference for comparisons with the dashed line representing the distribution for each head pose. The dynamics of both modalities are quite different. The identity distribution tends to spread as the head pose increases, whereas the appearance distribution remains more or less constant. The spreading could be explained by the fact that fewer facial attributes are visible as the pose increases. Therefore it is more difficult to estimate the correct geometry. On the other hand, the facial appearance remains mostly the same as the head rotates. The observed skin appearance is less affected by the rotation in this setup.

A solution to avoid this phenomenon would be to carefully build the training dataset to ensure each subject is present on multiple images. Then during training, similarity constraints could be applied between samples of the same subjects. This approach assumes that the identity of the subject is known for each sample to pair them correctly. However, with in-the-wild



(a) Synthetic image sequence with changing head pose for 0° to 60°.



(b) Identity and appearance coefficients distributions predicted by the ResNet18-based reconstruction network.

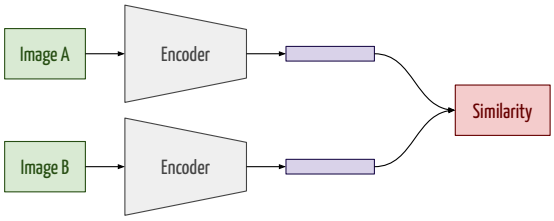
Figure 4.1 – 3DMM parameter distributions inconsistencies across head pose variations

datasets, this information might not be available. Instead, we build a solution upon [Genova et al., 2018] where they use the generative face model to create images with different head poses coupled with the recent advances in contrastive learning to impose similarity constraints without knowing any prior information about the subject on the image. The following sections will cover the description of the proposed solution, the evaluation, and the comparison against the baselines established previously in Chapter 3.

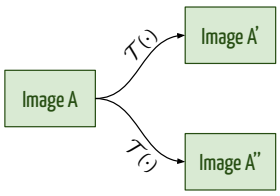
4.2 Methods

Over the recent years, many studies have presented solutions to learn visual representations of images in an unsupervised manner using methods relative to contrastive loss. While being task-independent, these methods do not require any label. The motivation behind contrastive learning is similar to human learning patterns. People do not need to remember all the details of an object to be able to recognize it. They learn an abstract representation based on some characteristics of the object.

This concept is applied to images within a contrastive learning framework where it attempts to train a model to distinguish between similar and dissimilar pairs of images. More specifically, given pairs of images, the neural network-based encoder should predict representation or embeddings that are close to each other in the feature space for *positive* pairs (*i.e. similar*



(a) Contrastive learning framework



(b) Random Image augmentation

image/object) and embeddings that are further away from each other for *negative* pairs (*i.e. dissimilar image/object*). The distance between pairs of images is measured using similarity functions. Multiple types of similarity distance have been proposed in recent studies, but it is mainly based on the cosine distance. We refer the reader to the survey of Le-Khac *et al.* for a broader and more detailed review of the different possibilities [Le-Khac et al., 2020]. An overview of the concept is given in Figure 4.2a. Up to now, the pairs of images were assumed to be provided. However, extra information or labels are needed to associate pictures of the same content together. This need for labels can be alleviated by constructing the pairs out of a single image. The pairs are automatically generated through a process $\mathcal{T}(\bar{\boldsymbol{I}})$ called random augmentation that creates pseudo-labels by applying arbitrary transformations to the original images $\bar{\boldsymbol{I}}$, as depicted in Figure 4.2b. Usually, the random augmentation process is based on a combination of color transformations, image rotation and cropping, and any other geometrical transformation.

In the context of 3DMM, contrastive learning can be used to learn a visual representation that is independent of the head pose. Following the idea of [Genova et al., 2018], the generative scene model $\mathcal{H}$ defined in Equation 3.3 can be used to generate random instances with the current estimation of the set of parameters $\mathcal{X}$ under different head pose and position. The image pairs are formed by the original images and the randomly synthesized instances of the scene model $\mathcal{H}$. The embeddings produced by the encoder in the reconstruction network should then be closed.

However, having a pose-invariant feature space is suboptimal to predict the generative scene model's position $\boldsymbol{t}$ and orientation $\boldsymbol{q}$ parameters of the generative scene model. Following Gao *et al.*'s work, we propose to update the feature expansion module to compute two subspaces out of the backbone's features [Gao et al., 2020]. One is used to predict the coefficients of models of the facial attributes (*i.e.* $\boldsymbol{w}^s, \boldsymbol{w}^e, \boldsymbol{w}^t, \boldsymbol{w}^i$), and the other is used to estimate the pose-related parameters (*i.e.* $\boldsymbol{q}, \boldsymbol{t}$). The pose-invariance constraint is then only imposed on the features used to estimate the parameters of the face models and not the pose.

The pose-invariant feature space is learned using the framework of Chen and He named SimSiam introduced in [Chen and He, 2021]. Given two augmented images x_1 and x_2, an encoder network f composed of a backbone, a feature expansion module, and a projection MLP head processes each view. The weights of the encoder f are shared between the two views. One of the views is then transformed by a prediction MLP head, denoted as h, and

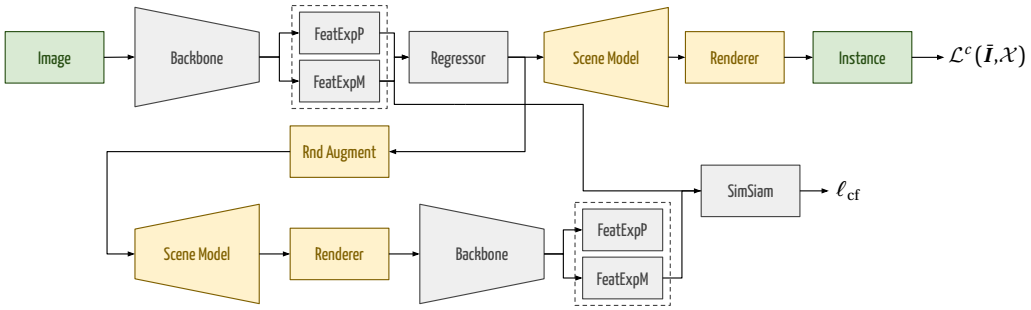


Figure 4.3 – Cross-Pose reconstruction architecture for training

matched against the other one. The matching is done by minimizing the negative cosine distance between the two representations. More formally, the output vectors are defined as $\boldsymbol{p}_1 = h\big(f(\boldsymbol{x}_1)\big)$, and $\boldsymbol{z}_2 = f(\boldsymbol{x}_2)$, the distance between them is given in:

$$\mathcal{D}\big(\boldsymbol{p}_1, \boldsymbol{z}_2\big) = -\frac{\boldsymbol{p}_1}{\|\boldsymbol{p}_1\|_2} \cdot \frac{\boldsymbol{z}_2}{\|\boldsymbol{z}_2\|_2}, \tag{4.1}$$

where $\|\cdot\|_2$ denotes the ℓ_2-norm of a vector. A critical component of the framework is the addition of a stop-gradient operation on z_2. It will be considered as a constant factor, and no gradient will be back-propagated through. The purpose is to avoid model collapsing, meaning every image is mapped to the same representation. Finally, following [Grill et al., 2020], the loss is symmetrized and defined as:

$$\ell_{cf} = \frac{1}{2}\Big(\mathcal{D}\big(\boldsymbol{p}_1, \texttt{stopgrad}(\boldsymbol{z}_2)\big) + \mathcal{D}\big(\boldsymbol{p}_2, \texttt{stopgrad}(\boldsymbol{z}_1)\big)\Big). \tag{4.2}$$

The complete loss function to minimize at the training stage is a combination of the objective function $\mathcal{L}^c\big(\bar{\boldsymbol{I}}, \mathcal{X}\big)$ defined in Equation 3.4 and the contrastive loss function ℓ_{cf} defined before, given in its compact form in:

$$\mathcal{L}\big(\bar{\boldsymbol{I}}, \mathcal{X}\big) = \mathcal{L}^c\big(\bar{\boldsymbol{I}}, \mathcal{X}\big) + \lambda_{cf}\ell_{cf}. \tag{4.3}$$

An overview of the cross-pose reconstruction architecture is shown in Figure 4.3. The reconstruction network is the same as the one presented in Section 3.1 with a modified feature expansion module. The module produces two subspaces, one for the pose-related parameters and another one for the parameters of the generative model. The random augmentation module, the scene model, and the renderer are used to generate augmented views of the input images. The two image representations are fed to the SimSiam module to impose similarity between them. Once the system has been trained, only the reconstruction network is needed for inference.

Table 4.2 – Detailed SimSiam modules configuration

Module	Symbol	Layers Configuration
Projector	f_p	Dense 1536, bias=`False` BatchNorm, m=0.9, $\epsilon = 1 \times 10^{-5}$, center=`True, scale=True` ReLU Dense 1536, bias=`False` BatchNorm, m=0.9, $\epsilon = 1 \times 10^{-5}$, center=`True, scale=True` ReLU Dense 1536, bias=`False` BatchNorm, m=0.9, $\epsilon = 1 \times 10^{-5}$, center=`False, scale=False`
Predictor	h	Dense 384, bias=`False` BatchNorm, m=0.9, $\epsilon = 1 \times 10^{-5}$, center=`True, scale=True` ReLU Dense 1536, bias=`True`

4.2.1 Implementation details

The modified reconstruction network is trained on the dataset introduced in Section 3.4. It ensures a fair comparison against the baseline presented in Section 3.1. The loss function of Equation 4.3 is minimized using an Adam optimizer and a piecewise constant learning rate with a batch size of 32 images. The learning rate is linearly increased from 1×10^{-6} to $1 \times^{-5}$ over the first 10k steps of the optimization for the ResNet18-based network and from 1×10^{-6} to 5×10^{-5} for the EfficientNetV2-based network. The complete learning rate schedules are shown in Figure 4.4.

The architecture of the projector and predictor in the SimSiam module is the same as in the original paper. A weight decay w_d is applied on the kernel of the convolution and linear layers of the encoder f and the predictor h. Moreover, the learning rate is kept fixed for all the layers in the predictor h. Fixing the learning rate of the predictor's head removes the need to add an Exponential Moving Average (EMA) on the parameters of f and h in the second branch [Tian et al., 2021]. All the hyper-parameters are given in Table 4.1, and the complete configurations of the projector head and predictor head of the SimSiam module are listed inTable 4.2.

Table 4.1 – Hyper-parameters

Symbol	Value	Description
λ_{cf}	1.0	Contrastive contribution
w_d	1×10^{-4}	Weight decay
lr	-	Learning rate
b	32	Batch size

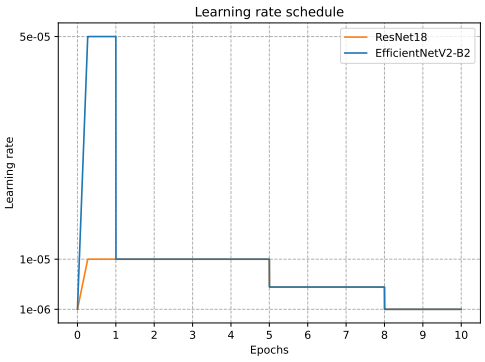
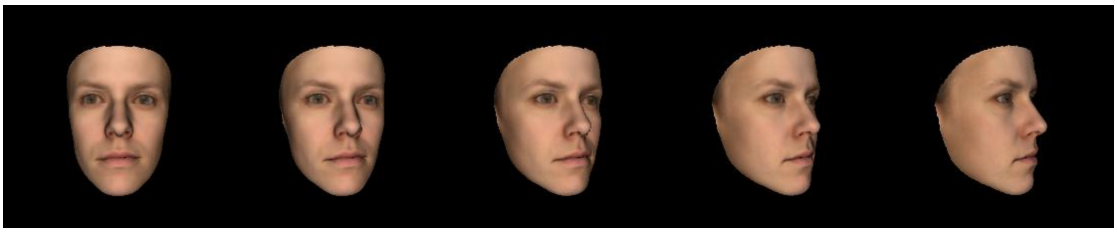
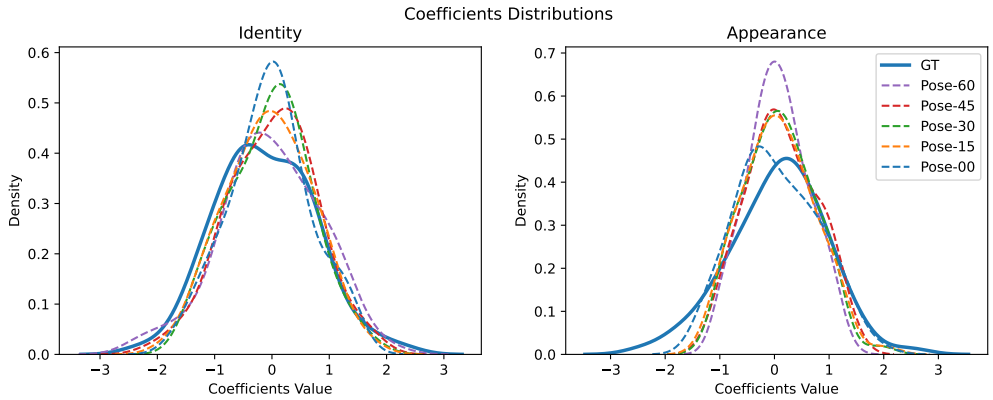


Figure 4.4 – Learning rate schedule



(a) Synthetic image sequence with changing head pose for 0° to 60°.



(b) Identity and appearance coefficients distributions predicted by the ResNet18-based pose-aware reconstruction network.

Figure 4.5 – 3DMM parameter distributions consistencies across head pose variations

4.3 Results

The synthetic example introduced in Section 4.1 is used to quantitatively assess the proposed modified training scheme's validity. The evaluation protocols established in Section 3.5 will be used to evaluate the quality of the reconstructed facial geometry and appearance. The consistency of the predicted parameters will be measured on the MICC dataset using the video sequences of each partition.

4.3.1 Synthetic

Due to the use of synthetic image sequences, the true distribution of the identity and appearance parameters is known. Figure 4.5b shows the parameter distributions predicted by the proposed pose-aware reconstruction network. One can notice that the distributions do not perfectly match the ground truth (*i.e. bold blue line*). However, the predicted parameters are more consistent across the head pose span than baseline shown in Figure 4.1b. The proposed approach effectively increases the consistency of the predicted parameters.

The deviation between the true distribution and one of the predicted parameters can be measured. To this end, the distance between them is measured with the Jensen-Shannon

Table 4.3 – Average Jensen-Shannon distance a across synthetic random sequence. The symbol ✓ indicates a pose-aware network whereas the ✗ indicates the baseline reconstruction network.

Method	Pose-Aware	Jensen-Shannon Distance	
		Identity	Albedo
ResNet18	✗	0.49 ± 0.02	0.38 ± 0.01
EfficientNetV2-B2	✗	0.51 ± 0.03	0.42 ± 0.02
ResNet18	✓	$\mathbf{0.44 \pm 0.03}^{\star}$	0.38 ± 0.02
EfficientNetV2-B2	✓	$\mathbf{0.47 \pm 0.03}^{\star}$	0.41 ± 0.03

* p-value < 0.05

distance defined as:

$$\mathcal{D}_{\mathrm{JS}}\left(\boldsymbol{p}, \boldsymbol{q}\right) = \sqrt{\frac{D_{\mathrm{KL}}\left(\boldsymbol{p} \parallel \boldsymbol{m}\right) + D_{\mathrm{KL}}\left(\boldsymbol{q} \parallel \boldsymbol{m}\right)}{2}}, \tag{4.4}$$

where $\boldsymbol{m} = \frac{1}{2}\left(\boldsymbol{p} + \boldsymbol{q}\right)$ is the point-wise mean of $\boldsymbol{p}$ and $\boldsymbol{q}$, and $D_{\mathrm{KL}}\left(\boldsymbol{x} \parallel \boldsymbol{y}\right) = \sum_i \boldsymbol{x}_i \log\left(\frac{\boldsymbol{x}_i}{\boldsymbol{y}_i}\right)$ is the KL divergence between $\boldsymbol{x}$ and $\boldsymbol{y}$.

The distances for the two baselines presented in Section 3.1 and the pose-aware reconstruction networks are given in Table 4.3. Interestingly, the average distance between the true distribution and the predicted one is smaller for the identity parameters with the proposed training scheme. However, the average distance between the distributions of the albedo parameters remains the same. This indicates that imposing feature consistency across different head poses does not improve the quality of the reconstructed facial appearance.

4.3.2 Geometry

The quality of the reconstructed geometry is assessed on the two datasets presented in Section 3.5, the MICC and the FaceWarehouse datasets. The average point-to-plane (Equation 1.15) and the point-to-point (Equation 1.14) errors are used as metrics. The videos of the MICC dataset are also used to assess the consistency of the predicted parameters across the sequences of images of each video. As no ground truth is available to compare against, the consistency metric is defined as the sum of the per-channel variance of the parameters. If the predicted parameters are constant across the sequence, the per-channel variance will be low, close to zero. On the other hand, the variance will be high if the predictions are different across all the frames. More formally, the consistency metric for a sequence $\boldsymbol{W} = \{\boldsymbol{w}_j \mid j = 1, \ldots, K\}$ is defined as:

$$\mathcal{D}_{\mathrm{con}} = \sum_j \mathrm{Var}_j\left(\boldsymbol{W}\right), \tag{4.5}$$

Table 4.4 – Average point-to-plane reconstruction error (mm) and consistency metric across all subjects of the MICC dataset. The suffix *-B* denotes the baseline models and *-PA* indicates the pose-aware models.

Method	Cooperative		Indoor		Outdoor	
	Error	$\mathcal{D}_{\text{con}}$	Error	$\mathcal{D}_{\text{con}}$	Error	$\mathcal{D}_{\text{con}}$
ResNet18-*B*	1.30 ± 0.31	87.67 ± 25.84	1.25 ± 0.24	58.35 ± 16.81	1.35 ± 0.25	74.88 ± 25.63
EfficientNetV2-B2-*B*	1.27 ± 0.29	93.85 ± 29.93	1.21 ± 0.24	59.36 ± 17.36	1.24 ± 0.21	77.66 ± 30.03
ResNet18-*PA*	1.30 ± 0.31	$\mathbf{47.59 \pm 12.59}^{\star}$	1.28 ± 0.25	$\mathbf{35.99 \pm 10.71}^{\star}$	1.29 ± 0.24	$\mathbf{45.02 \pm 15.19}^{\star}$
EfficientNetV2-B2-*PA*	1.26 ± 0.28	$\mathbf{51.12 \pm 14.92}^{\star}$	1.24 ± 0.24	$\mathbf{34.14 \pm 10.89}^{\star}$	1.31 ± 0.25	$\mathbf{48.19 \pm 17.04}^{\star}$

* p-value < 0.05

Table 4.5 – Mean point-to-point reconstruction error (mm) on 180 meshes of 9 subjects from FaceWarehouse. The suffix *-B* denotes the baseline models and *-PA* indicates the pose-aware models.

Method	Model	Regions	
		Small	Large
ResNet18-*B*	BFM09	1.90 ± 0.46	2.19 ± 0.48
EfficientNetV2-B2-*B*	BFM09	1.93 ± 0.48	2.15 ± 0.50
ResNet18-*PA*	BFM09	1.92 ± 0.45	2.18 ± 0.45
EfficientNetV2-B2-*PA*	BFM09	1.87 ± 0.45	2.10 ± 0.44

where $\boldsymbol{W}$ is a matrix stacking on each row the predicted parameters for a given facial attribute, K is the length of the sequence, and Var_j is the variance of the j-th component of the parameters.

The results of the neutral expression reconstruction evaluated on the MICC dataset are given in Table 4.4. The previous results of the two baselines are also provided as an element of comparison. The consistency metric is computed and reported for the identity parameter $\boldsymbol{w}^s$ for all the methods. The proposed pose-aware reconstruction network performs similarly to the previously established baseline in terms of distance between the two surfaces. However, the identity parameters are predicted more consistently across the sequences of images, as indicated by a lower variance. This suggests that the model has difficulties learning the correct facial geometry even when constraints are applied to the feature space to be robust against head pose variations.

The quality of the reconstructed facial geometry is assessed on a subset of images with facial expressions from the FaceWarehouse dataset. Since only a single image is available for each facial expression, it does not make sense to compute the parameters consistency metric in this case. The average point-to-point error of the pose-aware reconstruction network is given in Table 4.5. For both regions, the average errors are similar to the baselines. This is the expected behavior as the reconstructions are done on frontal images. Thus having a pose-invariant feature space will not improve the fidelity of the reconstructed surface and does not degrade the performance either. Both of the pose-aware reconstruction networks have difficulties correctly reconstructing the jawline region, similar to the baselines.

Table 4.6 – Average cosine distance for the photo-to-rendering pairs of the LFW dataset. The suffix *-B* denotes the baseline models and *-PA* indicates the pose-aware models.

Method	**Model**	**LFW**	
		Same	Different
LFW - *Data*	-	0.50	0.07
[Genova et al., 2018]	BFM17	0.37	-
ResNet18-*B*	BFM09	**0.44**	0.12
EfficientNetV2-B2-*B*	BFM09	**0.46**	0.12
ResNet18-*PA*	BFM09	0.37	0.15
EfficientNetV2-B2-*PA*	BFM09	0.39	0.14

4.3.3 Texture

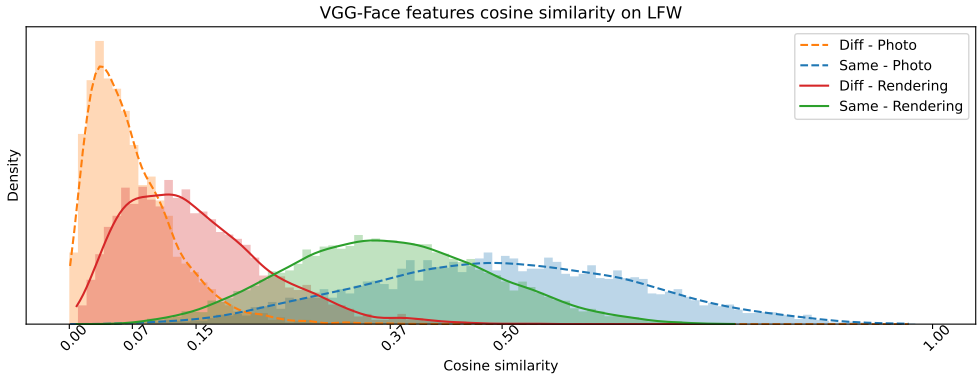
The quality of the reconstructed color is assessed using the evaluation protocol defined in Section 3.5 with the same dataset (*i.e. LFW*). The distributions of the similarity of the embeddings are shown in Figure 4.6 for both pose-aware reconstruction models. Imposing pose-invariance harms the recovery of the texture, as indicated by the shift to the left of the distribution for images of the same person (*i.e. green distribution*). This could suggest that having a single pose-invariant feature subspace is suboptimal for albedo and illumination estimation. As observed in Section 4.3.1, the albedo parameters are relatively consistent across the different head pose. Therefore the illumination parameters are most probably sensitive to head pose variations. The cosine similarity distributions for pairs of images of different subjects also shift to the right, indicating that the face recognition network has more difficulties differentiating between the subjects.

The average cosine distance similarity is given in Table 4.6. The results for the baselines are provided as elements of comparisons. The gap between baseline models and the pose-aware reconstruction networks is quite significant. The exact causes for the reconstruction errors are hard to identify due to the complexity of the problem. The performance drop put the proposed pose-aware reconstruction methods on par with the solution of [Genova et al., 2018].

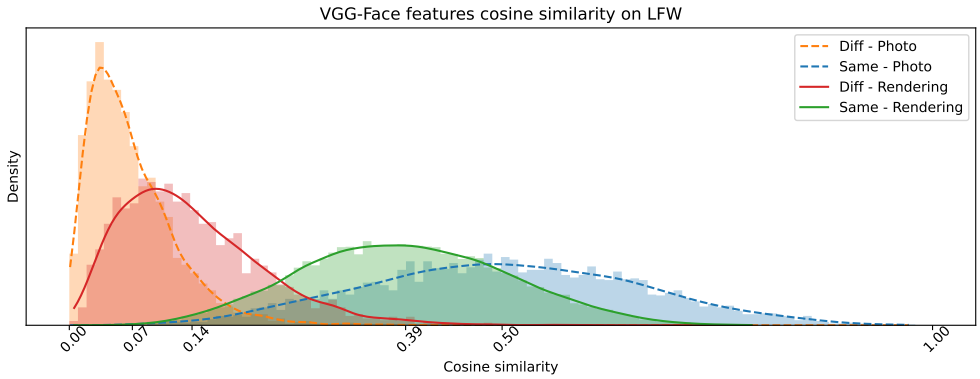
4.4 Summary

This chapter presents a solution to increase the consistency of the parameters predicted by the reconstruction network across a wide range of head pose. Our approach builds upon recent advances in self-supervised contrastive learning. We propose to use the SimSiam method of [Chen and He, 2021] to learn a pose-invariant feature subspace to predict the parameters of the generative models. Similar [Genova et al., 2018], we use the generative scene model to create on-the-fly augmented views of the input images under random head poses.

The proposed pose-aware reconstruction network is on par with the baselines established in Section 3.1 for the facial geometry. We have also shown that the parameters predicted by the



(a) ResNet18



(b) EfficientNetV2-B2

Figure 4.6 – Cosine similarity distributions of VGG-Face embeddings for pose-aware models.

pose-aware networks are more consistent for sequences of images. However, forcing similarity constraints on the feature subspace seems to significantly impact the quality of estimated facial albedo and illumination, as shown in Section 4.3.3. We hypothesized that having a pose-invariant feature subspace is suboptimal to estimate proper illumination coefficients. This could open future research directions.

We have also noticed that the proposed method is prone to over-fitting during our experiments, and hyper-parameters must be carefully selected. This could indicate that the augmentation technique used to generate the different views does not provide enough variability in the samples. Based on the comments of [Chen and He, 2021], the encoder did not collapse. The standard deviation of the average per-channel variance of the feature representation remains near the $1/\sqrt{d}$.

This concludes our first contribution to increasing the predicted face parameterization consistency across a wide range of head pose variations. The next chapter will investigate the possibilities to increase the robustness of the reconstruction networks for images of various resolutions.

5 Resolution-aware 3D Reconstruction

5.1 Introduction

In most recent works on monocular 3D face reconstruction, the assumption is that the reconstructed images are of high resolution or quality. However, when working with unconstrained or *in-the-wild* images, the pictures' quality and resolution can drastically change (*i.e. DSLR vs. security cameras*). In the context of low-resolution reconstruction, Mortazavian *et al.* proposed explicitly integrating the target image's resolution in the camera PSF by integrating downsampling operator in the virtual camera model [Mortazavian et al., 2012]. However, this formulation can not be used with neural network-based reconstruction as it is impossible to use PSF with modern differentiable rasterizers. Modern rasterizers are based on industry standards, such as *OpenGL*, which does not allow the use of a custom camera model to our knowledge.

The general trend with deep learning-based models when working with low-resolution images is to upsample them to match the required input size of the network. Furthermore, the performance of a model trained for a specific resolution but used with lower resolution images will typically decrease significantly. This issue can be solved in two ways by either applying super-resolution algorithms with the risk of introducing artifacts or by training different models for each targeted resolution.

Ideally, the backbone network should produce the exact image representation independently of the resolution of the input. To this end, we propose to follow the work of Xu *et al.* and adapt our baseline network presented in Chapter 3 to be resolution-aware to be able to do reconstruction independently of the input resolution [Xu et al., 2021]. In the subsequent sections of this chapter, the method will be introduced and discussed, the results of the different evaluation protocols will be given and compared against the baseline previously established in Chapter 3.

5.2 Methods

The selected backbones are based on the residual architecture introduced in [He et al., 2016a]. The core component of the design is called ResBlock, where several variations have been proposed over time. Without loss of generality, it is defined as:

$$\boldsymbol{z}_k = \boldsymbol{z}_{k-1} + \phi_k(\boldsymbol{z}_{k-1}), \tag{5.1}$$

where $\boldsymbol{z}_k$ is the feature maps produced by the k-th ResBlock and ϕ_k denotes a nonlinear function learning the feature residuals. This function is mainly composed of convolutions and nonlinear activation functions. However, multiple variants have been studied [He et al., 2016b]. The whole network is composed of numerous stacks of such blocks, and the final feature maps $\boldsymbol{z}_B$ is given by:

$$\boldsymbol{z}_B = \boldsymbol{z}_0 + \sum_{k=1}^{B} \phi_k(\boldsymbol{z}_{k-1}), \tag{5.2}$$

where $\boldsymbol{z}_0$ is the low-level feature maps extracted from the image $\bar{\boldsymbol{I}}$ by the stem of the network (*e.g. usually composed of multiple convolution layers with ReLU activation functions*) and B denotes the total number of residual blocks.

From the original formulation of the residual blocks given in Equation 5.2, Xu *et al.* proposed to modify it to integrate some information about the resolution at which a given block is operating. The proposed solution is based on the observation that two identical images (*i.e. of the same content*) at different resolutions are very similar. Thus most of the structure will be shared between them, but the quality will differ. Therefore, most parts of the feature extractor can be shared. Only a few parameters need to be resolution-dependent to adapt to the changes induced by the different resolutions. The proposed resolution-aware residual block is defined as:

$$\boldsymbol{z}_{i,B} = \boldsymbol{z}_{i,0} + \sum_{k=1}^{B} \alpha_{i,k}\phi_k\left(\boldsymbol{z}_{i,k-1}\right), \quad i = 1,\dots,R, \tag{5.3}$$

where $\boldsymbol{\alpha} \in \mathbb{R}^{R\times B}$ is a resolution-dependent learnable parameters that adapt the residual maps, i denotes the index of the image resolution (*i.e. $i = 1$ indicates the original resolution, $i > 1$ denotes smaller image*). R is the total number of resolutions considered, $\alpha_{i,k}$ and $\boldsymbol{z}_{i,k}$ are the adaptive weight and the feature maps of the k-th residual block for the i-th resolution.

This updated version of the residual block can easily be replaced in the original backbone adding only a small overhead of the adaptive weight $\boldsymbol{\alpha}$. However, the only downside of the modification is that the image and corresponding resolution index need to be provided as

input to the network to select the proper adaptive weights (*i.e. the correct row of* $\boldsymbol{\alpha}$). The remaining parts of the reconstruction network (i.e. the features expansion module, the multi-head predictor, and the model-based decoder) are the same as in Section 3.1.

5.2.1 Training strategy

The downside of the resolution-aware residual block is the amount of data required to learn the adaptive weight $\boldsymbol{\alpha}$. Each high-resolution image needs to be downsampled $R-1$ time to generate the appropriate data necessary to learn every row of $\boldsymbol{\alpha}$. Thus instead of considering all R resolutions, the span of R is divided into P ranges or *buckets*, covering multiple resolutions. A single set of adaptive parameters is learned, reducing the dimension of the adaptive weights $\boldsymbol{\alpha} \in \mathbb{R}^{P \times B}$ effectively. The first range is a particular case as it contains only the high-resolution image; the remaining buckets cover various image sizes. Images of lower dimensions $\bar{\boldsymbol{I}}_2, \ldots, \bar{\boldsymbol{I}}_P$ are generated from the high-resolution image $\bar{\boldsymbol{I}}_1$ at each training iteration using bicubic interpolation. The target image size is defined by sampling the bounds of the bucket of the i-th resolution level uniformly. This bucketing process effectively reduces the complexity during the training phase. One can argue that the way the low-resolution images are generated is not realistic and not representative of how real-life low-resolution photos are created. However, as mentioned by Bulat *et al.*, creating realistic low-resolution images is complex and still, an ongoing topic of research [Bulat et al., 2018].

The training strategy for the resolution-aware network is slightly different than the standard end-to-end strategy. Using all the possible image resolution ranges directly from the beginning leads to a precarious training process. This is mainly due to the model having issues handling at the same time image properties from various resolutions. To solve this problem, the resolution-aware is trained using a progressive scheme. First, the network is trained only on high-resolution images as they are easier to handle (*i.e. more semantic information is present*). Then images of lower resolutions are progressively added in the process one level at a time. This step is repeated until all resolution buckets have been added. This process of slowly increasing the reconstruction complexity makes the training process stable and allows the resolution-aware network to extract appropriate images representation effectively. The drawback of this strategy is that the training time is considerably increased.

The complete loss function to minimize during the progressive training strategy is a combination of the primary loss functions ℓ_b to constraint the 3D reconstruction, self-supervision loss ℓ_s, and contrastive feature loss ℓ_f as defined in its compact form in:

$$\mathcal{L}^m\left(\bar{\boldsymbol{I}}\right) = \ell_\mathrm{b} + \lambda_\mathrm{s}\ell_\mathrm{s} + \lambda_\mathrm{f}\ell_\mathrm{f}. \tag{5.4}$$

The primary loss function ℓ_b constrains the 3D reconstruction of every range of resolutions

and is defined as:

$$\ell_{\mathrm{b}} = \sum_{i} \mathcal{L}^{c}\left(\bar{\boldsymbol{I}}_{i}, \mathcal{X}_{i}\right), \tag{5.5}$$

where $\mathcal{L}^{c}\left(\bar{\boldsymbol{I}}_{i}, \mathcal{X}_{i}\right)$ is the loss function from Equation 3.4, $\bar{\boldsymbol{I}}_{i}$ and $\mathcal{X}_{i}$ are the image and the predicted set of parameters for the i-th resolution level. This term ensures correct reconstruction happens at each level.

Relying on photometric and landmark errors only when training with the lower resolution images (*i.e.* $i > 1$) is suboptimal as the quality of the targets will decrease with the image resolution. To help solve this issue, a self-supervised loss ℓ_{b} is added to help with the network's training. Self-supervision loss is designed to ensure that the network predicts similar feature representation for identical images under different augmentation conditions. The lower resolution images are generated from the high-resolution images and can be seen as different types of image augmentations. Consider the set of parameters $\mathcal{X}_1$ predicted from the high-resolution image $\bar{\boldsymbol{I}}_1$ by the resolution-aware reconstruction network $\mathcal{X}_1 = f_{\mathrm{RA}}\left(\bar{\boldsymbol{I}}_1\right)$. Ideally, they should be the same as the one predicted from another image resolution $\mathcal{X}_1 = \mathcal{X}_j | j \in \{2, \ldots, P\}$. The self-supervision loss minimizes the Mean Square Error (MSE) between the sets of parameters predicted by the reconstruction network and is defined as:

$$\begin{aligned} \ell_{\mathrm{s}} &= \sum_{i,j} w_{i,j} \| \bar{f_{\mathrm{RA}}}\left(\bar{\boldsymbol{I}}_{i}\right) - f_{\mathrm{RA}}\left(\bar{\boldsymbol{I}}_{j}\right) \|_{2}^{2}, \\ w_{i,j} &= \mathbb{1}\left(j - i > 0\right) \cdot \left(j - i\right), \end{aligned} \tag{5.6}$$

where $w_{i,j}$ is the importance weight for the image pair $\left(\bar{\boldsymbol{I}}_{i}, \bar{\boldsymbol{I}}_{j}\right)$. It is nonzero only when the image $\bar{\boldsymbol{I}}_{i}$ is of higher resolution than $\bar{\boldsymbol{I}}_{j}$ and provides more robust guidance when the resolution gap is significant. The $\bar{f_{\mathrm{RA}}}$ denotes the network where no gradient will be back-propagated (*i.e. directional*). This design ensures lower resolution images $\bar{\boldsymbol{I}}_{j}$ produce a similar set of parameters as those predicted from higher resolution images $\bar{\boldsymbol{I}}_{i}$ and avoid propagating errors from low-resolution images into the high-resolution parameters (*e.g. adaptive weight* $\boldsymbol{\alpha}$).

The training of the resolution-aware network can be further improved by enforcing the final image representation $\boldsymbol{\varphi}$, out of which the set of parameters $\mathcal{X}$ are regressed to be similar across different image resolutions. This feature consistency loss is defined as:

$$\ell_{\mathrm{f}} = \sum_{i,j} w_{i,j}\, g\left(\bar{\boldsymbol{\varphi}}_{i}, \boldsymbol{\varphi}_{j}\right), \tag{5.7}$$

where $w_{i,j}$ is identical to Equation 5.6, $\boldsymbol{\varphi}_{i}$ denotes the final feature representations of the image $\bar{\boldsymbol{I}}_{i}$ for the i-th resolution level, $\bar{\boldsymbol{\varphi}}$ indicates that no gradient will be back-propagated

to this feature representation and $g(\cdot,\cdot)$ is a function responsible for maximizing the mutual information between its inputs. Contrastive loss is designed to encourage feature representations of identical images (*i.e. at different resolutions*) to be close to each other but far from other images of different content. The contrastive feature loss follows the formulation of [He et al., 2020] and can be written as:

$$g\left(\bar{\boldsymbol{\varphi}}_i, \boldsymbol{\varphi}_j\right) = -\log \frac{\exp\left(\rho\left(\bar{\boldsymbol{\varphi}}_i, \boldsymbol{\varphi}_j\right)/\tau\right)}{\exp\left(\rho\left(\bar{\boldsymbol{\varphi}}_i, \boldsymbol{\varphi}_j\right)/\tau\right) + \sum_{\boldsymbol{q}\in\mathcal{Q}} \exp\left(\rho\left(\boldsymbol{q}, \boldsymbol{\varphi}_j\right)/\tau\right)}, \tag{5.8}$$

where $\rho(\cdot,\cdot)$ is a function measuring the similarity between its input usually defined as the cosine distance, τ is the temperature hyper-parameter, $\mathcal{Q}$ is a queue of previous data points built progressively at each iteration throughout the training process such that $\boldsymbol{\varphi}_i, \boldsymbol{\varphi}_j \notin \mathcal{Q}$. The contrastive feature loss is effectively a softmax-based classifier in which the resolution pair $\left(\boldsymbol{\varphi}_i, \boldsymbol{\varphi}_j\right)$ is the positive sample. In contrast, the features in the queue are negative samples. Similar to [Chen et al., 2020], the similarity function $\rho(\cdot,\cdot)$ includes a two layers MLP to project the features into a smaller subspace of 128 dimensions to measure the similarity between them.

5.2.2 Implementation Details

The resolution-aware reconstruction network is trained on the dataset introduced in Section 3.4. The image resolution span has been split into $P = 4$ ranges defined as $\{224, [128,224), [64,128), [32,64)\}$. The low-resolution images are generated by downsampling the high-resolution image $\bar{\boldsymbol{I}}_i$ to the target image size and resizing it back to 224 in order to match the size required by the network using bicubic interpolation. The loss function of Equation 5.4 is minimized using an Adam optimizer and a piecewise constant learning rate schedule with a batch size of 32 images. Similar to the baseline configuration, the learning rate is linearly increased from 1×10^{-6} to 8×10^{-5} over the first 10k steps of the optimization. Figure 5.1 shows the exact learning rate schedule used to train the network. The list of hyper-parameters for the added objective functions is given in Table 5.1, whereas the basic loss functions ℓ_{b} reuses the parameters given in Table 3.2.

Table 5.1 – Hyper-parameters

Symbol	Value	Description
λ_s	0.5	Self-supervision contribution
λ_f	0.5	Contrastive contribution
τ	0.1	Temperature
$\|\mathcal{Q}\|$	8192	Queue size
lr	-	Learning rate
b	32	Batch size

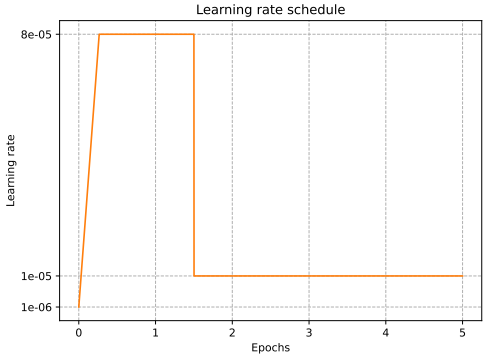


Figure 5.1 – Learning rate schedule

5.3 Results

The evaluation of both modalities of the resolution-aware reconstruction model presented earlier follows the same protocols defined previously in Section 3.5. However, applying these protocols at every image size contained in each resolution bucket would be cumbersome. Therefore only the middle image resolution of each bucket is considered, more specifically $\{224, 176, 96, 48\}$. Similar to the training phase, the images are downsampled to the target resolution and resized to the input dimensions of the network using bicubic interpolation. The transformed images and the resolution indexes are fed to the resolution-aware reconstruction network to predict the different parameters of the models and reconstruct the images.

5.3.1 Geometry

The quality of the reconstructed geometry is assessed on the two datasets presented in Section 3.5, the MICC and the FaceWarehouse datasets. The average point-to-plane and the point-to-point error metrics are given in Table 5.2 at various image resolutions. The metrics from the baseline models established previously in Section 3.6.1 are also reported for comparison.

With high-resolution images, the reconstruction error on the three partitions of the MICC dataset of the resolution-aware network is on par with the baseline established earlier. It indicates that the added resolution-dependent mechanism does not penalize the system in the first place. The performance on other image resolutions showed that the resolution-aware reconstruction network can produce facial geometry of similar quality. This is the intended behavior imposed by the ℓ_s and ℓ_f objective functions during the training stage. It further indicates that the progressive training scheme is valid.

Regarding the evaluation on the FaceWarehouse dataset, the difference between the two regions is present as well. It indicates that once again, the resolution-aware reconstruction network has troubles reconstructing the jawline region. This is expected as the network archi-

Table 5.2 – Average point-to-plane reconstruction error (mm) across all subjects of the MICC dataset and average point-to-point reconstruction error (mm) on the FaceWarehouse dataset for different image resolutions. The suffix *-B* denotes the baseline model and *-RA* indicates the resolution-aware model.

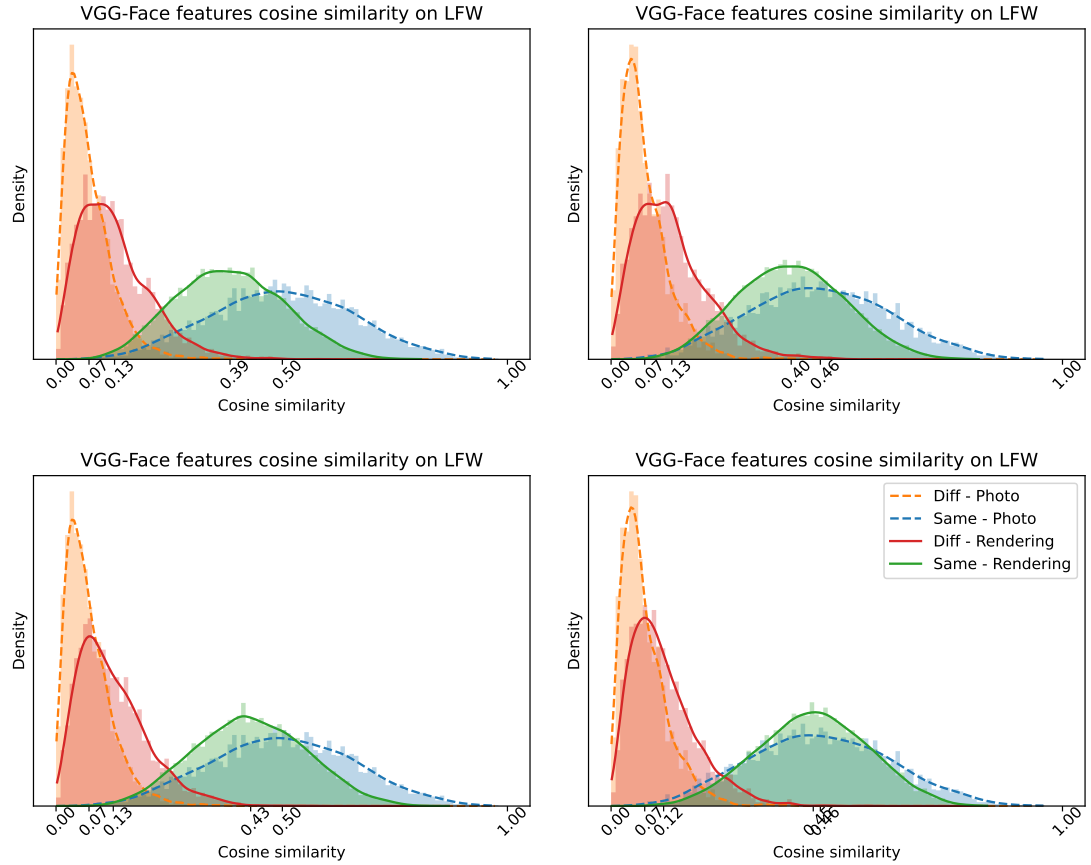
Method	**Resolutions**	**MICC**			**FaceWarehouse**	
		Cooperative	Indoor	Outdoor	Small	Large
ResNet18-*B*	224	1.30 ± 0.31	1.25 ± 0.24	1.35 ± 0.25	1.90 ± 0.46	2.19 ± 0.48
EfficientNetV2-B2-*B*	224	1.27 ± 0.29	1.21 ± 0.24	1.24 ± 0.21	1.93 ± 0.48	2.15 ± 0.50
ResNet18-*RA*	224	1.28 ± 0.31	1.27 ± 0.26	1.32 ± 0.28	1.97 ± 0.45	2.19 ± 0.46
	176	1.28 ± 0.31	1.27 ± 0.26	1.32 ± 0.28	1.97 ± 0.45	2.19 ± 0.46
	96	1.28 ± 0.31	1.27 ± 0.26	1.33 ± 0.28	1.96 ± 0.45	2.19 ± 0.45
	48	1.28 ± 0.30	1.27 ± 0.26	1.33 ± 0.28	1.96 ± 0.45	2.18 ± 0.45
EfficientNetV2-B2-*RA*	224	1.30 ± 0.29	1.23 ± 0.24	1.28 ± 0.28	1.99 ± 0.43	2.27 ± 0.46
	176	1.30 ± 0.30	1.22 ± 0.24	1.27 ± 0.23	2.02 ± 0.45	2.27 ± 0.46
	96	1.30 ± 0.29	1.23 ± 0.24	1.28 ± 0.24	1.99 ± 0.43	2.26 ± 0.46
	48	1.29 ± 0.30	1.23 ± 0.24	1.27 ± 0.24	2.06 ± 0.47	2.28 ± 0.46

tecture has not fundamentally changed compared to the baseline network. However, there is a slight error increase for the inner part of the face. One can observe that the performances across the different image resolutions are similar and consistent.

5.3.2 Texture

The quality of the reconstructed color is assessed using the same evaluation protocol as defined for the baseline model; the evaluation is carried out on the same dataset as well (*i.e. LFW*). The distributions of the similarity of the embeddings are shown in Figure 5.2 for both resolution-aware models. Only the extremums are displayed, the highest resolution is depicted in Figure 5.2a, and the lowest resolution is shown in Figure 5.2b. One can notice that both photo-to-rendering similarity distributions (*i.e. same person and different person*) are slightly shifted, indicating lower performances, even at the highest resolution. However, the shift is constant across the whole range of image resolution, meaning that the resolution-aware model is not sensitive to image size. It is worth mentioning that the face embedding extractor is sensitive to the image resolution, as shown in Figure 5.2b. The photo-to-photo similarity distribution slightly shifts to the left (*i.e. dashed blue curve*), indicating that the VGG-Face network is not invariant to image size. However, the displacement is not significant and does not impact much the evaluation protocols as the overall shape of the distribution remains constant across the image resolution change.

The average cosine distance for each image resolution level is given in Table 5.3 together with previously reported methods for comparisons. The performances of both of the resolution-aware models are slightly lower than the one of the two baselines established before.



(a) Image size - 224

(b) Image size- 48

Figure 5.2 – Cosine similarity distribution of VGG-Face embedding for different model, ResNet18-*RA* (*first row*), EfficientNet-V2-B2-*RA* (*second row*).

Table 5.3 – Average cosine distance of the photo-to-rendering pairs of the LFW dataset. The suffix *-B* denotes the baseline model and *-RA* indicates the resolution-aware model.

Method	**Model**	**Resolution**	**LFW**	
			Same	Different
LFW - *Data*	-	224	0.50	0.07
LFW - *Data*	-	48	0.46	0.07
[Genova et al., 2018]	BFM17	224	0.37	-
ResNet18-*B*	BFM09	224	0.44	0.12
EfficientNetV2-B2-*B*	BFM09	224	0.46	0.12
ResNet18-*RA*	BFM09	224	0.39	0.13
		176	0.39	0.13
		96	0.39	0.13
		48	0.40	0.13
EfficientNetV2-B2-*RA*	BFM09	224	0.43	0.13
		176	0.43	0.12
		96	0.43	0.13
		48	0.45	0.12

5.4 Summary

This chapter presents a method to perform monocular 3D face reconstruction across a wide range of image resolutions without significantly dropping performance. The solution capitalizes on the resolution-aware residual block introduced in [Xu et al., 2021] and the recent advances in self-supervised and contrastive learning to train a robust reconstruction network.

The proposed system has been evaluated following the protocols defined in Section 3.5. The quality of the reconstructed facial geometry and appearance is similar to the baseline networks, as indicated by the geometric metrics and photometric similarity. The evaluations of the resolution-aware models also showed that the system is performing similarly across a wide range of image resolutions.

The solution works across an extensive range of image resolutions by sharing most of the feature extractor across the different resolution streams and has only a small subset of parameters depending on the input image size. This approach does not increase the complexity of the whole system drastically at inference. However, this formulation is limited to residual-based networks and can not be directly transferred to purely convolutional architecture like VGG-based networks. The main drawback of the method is the increased complexity of the training stage. The progressive training of the reconstruction system induces a huge memory footprint and time penalty, limiting quickly and drastically the system's capacity. In practice, the input images will most of the time be preprocessed before being input into the reconstruction network. Thus the resolution index can be assumed to be known based on the dimensions of the detected bounding boxes.

The use of resolution-aware residual blocks can effectively reduce the impact of the second limiting factor identified in Section 2.6. The next chapter will introduce our last contribution, an approach to recover the mid-level facial details lost with PCA-based statistical shape models.

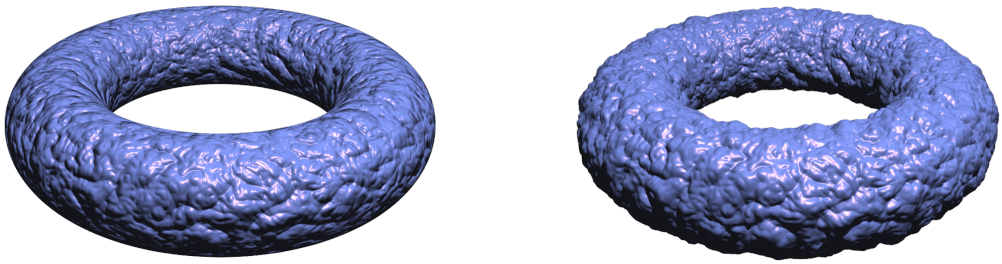
6 Fine Facial Details Recovery

6.1 Introduction

It is widespread to represent facial geometry through the literature with classical PCA-based statistical models. However, one of the main drawbacks of this approach is the lack of capability to model mid-level deformations such as facial wrinkles. This limitation is explained by how such models are built, where only the strongest axis of deformations present in the data will be kept. Therefore all the tiny shape variations will be lost in the process (*i.e. low-pass filtering of surface deformations*) and out of the geometry distribution. This lack of fine facial details is quite hurting in the presence of facial expressions where nonrigid deformations happen around some face components such as the eyebrows or the mouth corner. These depth cues play an important role in perceiving and interpreting the correct facial expressions and are needed.

The recovery of this lost information could be tackled in two ways: using a different type of statistical model that can incorporate both low-level and mid-level facial deformations or by adding a corrective model on top of the coarse PCA-based shape model. In the scope of this thesis, we have decided to follow the second option instead of building a new statistical shape model.

Researchers from the computer graphics community have spent lots of time developing techniques for rendering photo-realistic objects from various mesh complexity. One solution to increase the realism of a rendered low polygon mesh is *normal* maps [Mikkelsen, 2008]. The final color depends on the interaction of the scene's light and the object's color being rendered. This light-object interaction is modeled using the normals of the object as described in Section 2.2. Therefore to increase the realism of the rendered image, the shading process uses pre-computed normals, backed into a texture, instead of the true normals of the surfaces to compute the final colors. This principle creates the illusion that the object is textured even though the geometry has not moved. Figure 6.1a shows a rendering using a normal map for the computation of the shading on a donut-like mesh. This technique can generate photo-realistic images with a high level of detail by adding proper depth cues to the color.



(a) (b)

Figure 6.1 – Comparison between normal map (a) and displacement map (b) applied to a donut-like shape, [Mikkelsen, 2008]

However, by looking at the object's boundaries, one can see that the illusion is not correct as no bumps are present (*i.e. smooth silhouette*).

For meshes with high polygon density, there is another solution to generate photo-realistic images. The normal maps are replaced by *displacement* maps instead. A displacement map contains actual displacements or corrections to add on top of the mesh geometry [Mikkelsen, 2008]. The idea behind the displacement map concept is to add small correction δ, backed into a texture, along the normal $\boldsymbol{n}$ of the surface: $\boldsymbol{s}' = \boldsymbol{s} + \delta \cdot \boldsymbol{n}$. For this adjustment to be correct, the surface needs to be highly tessellated. Figure 6.1b shows a rendered torus mesh with a displacement map applied to it to add high-level detail.

As illustrated in Figure 6.1, the use of displacement maps is more geometrically correct than using normal maps (*i.e. on the borders*). Therefore we use displacement maps to represent the facial details, such as the wrinkles, in the experiment conducted in this thesis. Moreover, the displacement map avoids the discrepancy between the geometry of the object and the surface normal. This is not true with a normal map. The following sections will go over how the facial geometry is refined using displacement map and how the maps are recovered from target images without having any ground truth, and the evaluation protocol used to assess the quality of the proposed method. Lastly, a discussion on the benefits and limitations of the proposed method will conclude this chapter.

6.2 Methods

The refined facial geometry $\mathcal{S}^f$ is composed of the coarse surface $\mathcal{S}$ defined in Equation 2.11 and a small correction δ in the direction of the surface's normal. The corrected position of the j-th vertex is given by:

$$\mathcal{S}_j^f\left(\boldsymbol{w}^s, \boldsymbol{w}^e, \mathcal{D}\right) = \mathcal{S}_j\left(\boldsymbol{w}^s, \boldsymbol{w}^e\right) + \mathcal{D}_{\text{uv}} \cdot \boldsymbol{n}_j, \tag{6.1}$$

where $\boldsymbol{n}_j$ is the normal of the j-th vertex, $\mathcal{D}_{\text{uv}} = \textbf{interp}\left(\mathcal{D}, u_j, v_j\right)$ is the interpolated value from the displacement map $\mathcal{D} \in \mathbb{R}^{N_d \times N_d}$, and u_j, v_j are the texture coordinates of the j-th vertex. The refined generative scene model is defined in the same way as in Equation 3.3. By plugging $\mathcal{S}_j^f\left(\boldsymbol{w}^s, \boldsymbol{w}^e, \mathcal{D}\right)$ into $\mathcal{H}\left(\mathcal{X}\right)$, the refined per-vertex position and color attributes of the j-th vertex are computed with the generative model $\mathcal{H}^f\left(\mathcal{X}, \mathcal{D}\right)$ given in:

$$\mathcal{H}_j^f\left(\mathcal{X}, \mathcal{D}\right) = \left(\mathcal{C}_j\left(\mathcal{A}_j, \boldsymbol{n}_j^f, \boldsymbol{w}^i\right), \boldsymbol{M}_{\text{model}}\left(\boldsymbol{q}, \boldsymbol{t}\right)\left[\mathcal{S}_j^f\left(\boldsymbol{w}^s, \boldsymbol{w}^e, \mathcal{D}\right), 1\right]\right), \tag{6.2}$$

where $\mathcal{X} = \{\boldsymbol{w}^s, \boldsymbol{w}^e, \boldsymbol{w}^t, \boldsymbol{w}^i, \boldsymbol{q}, \boldsymbol{t}\}$ is the complete set of parameters generating the coarse reconstruction, $\mathcal{D}$ denotes the displacement map, $\mathcal{S}^f$ represents the refined geometry (*i.e. including facial details*), and $\boldsymbol{n}^f$ is the normal of the refined surface used during the shading stage.

Fitting: The parameters of the refined 3D face reconstruction are estimated by minimizing the objective function defined in:

$$\mathcal{L}^f\left(\bar{\boldsymbol{I}}, \mathcal{X}, \mathcal{D}\right) = \mathcal{L}^c\left(\bar{\boldsymbol{I}}, \mathcal{X}\right) + \lambda_{\text{ph\_f}}\ell_{\text{ph\_f}} + \lambda_{\text{silh\_f}}\ell_{\text{silh\_f}} + \lambda_{\text{lms\_f}}\ell_{\text{lms\_f}} + \lambda_{\text{reg\_f}}\ell_{\text{reg\_f}}. \tag{6.3}$$

The displacement map embeds corrections of small magnitude. Thus the reconstruction of the general shape and color of the face needs to be constrained as well. To this end, the objective function $\mathcal{L}^c\left(\bar{\boldsymbol{I}}, \mathcal{X}\right)$ defined in Equation 3.4 is used to estimate all the parameters of the generative scene model $\mathcal{H}\left(\mathcal{X}\right)$. The estimation of the small shape correction $\mathcal{D}$ is mostly constrained once again by measuring the photometric error between the refined reconstruction and the target image, similar to Equation 3.5. The photometric loss $\ell_{\text{ph\_f}}$ measures the difference between the synthesized refined image $\boldsymbol{I}\left(\mathcal{X}, \mathcal{D}\right)$ and the target image $\bar{\boldsymbol{I}}$ and is defined as:

$$\ell_{\text{ph\_f}}\left(\bar{\boldsymbol{I}}, \mathcal{X}, \mathcal{D}\right) = \sum_l^L \frac{1}{\left|\mathcal{F}_{\text{fine}}^l \cap \mathcal{M}^l\right|} \sum_{p \in \mathcal{F}_{\text{fine}}^l \cap \mathcal{M}^l} \frac{2^l}{\sigma_{\text{ph}}} \left\|\boldsymbol{I}_p^l\left(\mathcal{X}, \mathcal{D}\right) - \bar{\boldsymbol{I}}_p^l\right\|_1^1 - \log\left(h_{\mathcal{B}}\left(\boldsymbol{I}_p^l\left(\mathcal{X}, \mathcal{D}\right)\right)\right), \tag{6.4}$$

where $l = \{0, \cdots, L-1\}$ indicates the level in the image pyramid, $\mathcal{F}_{\text{fine}}^l$ denotes the region covered by the projected refined geometry at level l (*i.e. foreground*), $\mathcal{M}^l$ is the segmentation mask indicating the probability of a pixel being explainable by the face model, σ_{ph}^2 is the approximated variance of the residual photometric error, and $h_{\mathcal{B}}$ is an image-based histogram modeling the likelihood of a pixel being part of the background.

To help with the initial coarse alignment of the generated surface with the target image, the silhouette loss $\ell_{\text{silh\_f}}$ based on the IoU distance is used. The cost function measures the misalignment between the foreground region $\mathcal{F}_{\text{fine}}$ defined by the generative model (*i.e. projected geometry*) and the segmentation mask $\mathcal{M}$. The cost function is defined in Equation 3.6 but is

provided below for clarity:

$$\ell_{\text{silh\_f}}(\mathcal{M}, \mathcal{F}_{\text{fine}}) = 1 - \frac{|\mathcal{M} \cap \mathcal{F}_{\text{fine}}|}{|\mathcal{M}| + |\mathcal{F}_{\text{fine}}| - |\mathcal{M} \cap \mathcal{F}_{\text{fine}}|},$$

where $|\cdot|$ denotes the cardinality of the mask. To further help with geometric alignment, the distance between the refined projected facial landmarks and detected ones must be small. This constraint is enforced by the landmark loss $\ell_{\text{lms\_f}}$ defined earlier in Equation 2.34. For clarity purposes, it is redefined in:

$$\ell_{\text{lms\_f}}(\mathcal{X}, \mathcal{D}) = \frac{1}{F} \sum_{j=1}^{F} c_j \cdot \| \Pi_{k_j} \left(\mathcal{S}^f(\mathcal{X}, \mathcal{D}) \right) - \boldsymbol{l}_j \|_1^1,$$

where $\boldsymbol{l}_j$ is a detected landmark, $c_j \in [0,1]$ indicates the quality of the detection, Π_{k_j} is a function projecting the k_j-th vertex to the image plane, and $\mathcal{S}^f(\mathcal{X}, \mathcal{D})$ is the refined reconstructed surface (*i.e. geometry in camera space*).

The last term of the objective function is the regularization of the displacement map $\mathcal{D}$. This term ensures that the spatial variations of the corrective field stored in the displacement map are smooth. The corrections applied to the coarse geometry should be locally similar. Therefore the regularization term $\ell_{\text{reg\_f}}$ penalizes the magnitude of the Laplace operator applied to the displacement map and is defined as:

$$\ell_{\text{reg\_f}}(\mathcal{D}) = \| \Delta \mathcal{D} \|_2^2. \tag{6.5}$$

Using the analysis-by-synthesis fitting strategy presented in Section 2.5, the displacement map $\mathcal{D}$ and the set of parameters $\mathcal{X}$ of the generative scene model $\mathcal{H}^f$ are estimated by minimizing the objective function defined in Equation 6.1. In our experiment, the cost function is minimized using an Adam optimizer for about 4k iterations. The complete list of hyper-parameters is given in Table 6.1.

6.3 Results

The proposed method is evaluated on the FACES[1] dataset introduced in [Ebner et al., 2010]. This choice of evaluation data is motivated by various aspects. The dataset proposed for research on facial expression analysis is composed of images of people performing posed expressions (*i.e. acted*). Therefore it guarantees deformations directly linked to facial expressions are present in the data, and the magnitude of these deformations cover a wide range of values. Each image of the subjects is captured in a controlled environment, meaning there is

[1] Available upon request at https://faces.mpdl.mpg.de/imeji

Table 6.1 – Hyper-parameters

Symbol	Value	Description
λ_{ph}	1.0	Photometric contribution
λ_{silh}	5.0	Silhouette contribution
λ_{lms}	0.3	Landmarks contribution
λ_{stats}	1×10^{-3}	Statistical prior contribution
λ_{id}	3×10^{-4}	Identity prior contribution
λ_{exp}	2.5×10^{-4}	Expression prior contribution
λ_{tex}	1.0×10^{-4}	Albedo prior contribution
λ_{range}	1000	Albedo range importance
λ_{sym}	50	Albedo symmetry importance
σ_{ph}	0.043[†]	Variance of the photometric residue
$\lambda_{\text{ph\_f}}$	1.0	Photometric contribution
$\lambda_{\text{silh\_f}}$	5.0	Silhouette contribution
$\lambda_{\text{reg\_f}}$	5×10^{-2}	Displacement regularization
N_d	256	Displacement map dimension
L	3	Number of level in pyramid
l	0.0	Appearance lower bound
u	1.0	Appearance upper bound
lr	1×10^{-3}	Learning rate
b	5	Batch size

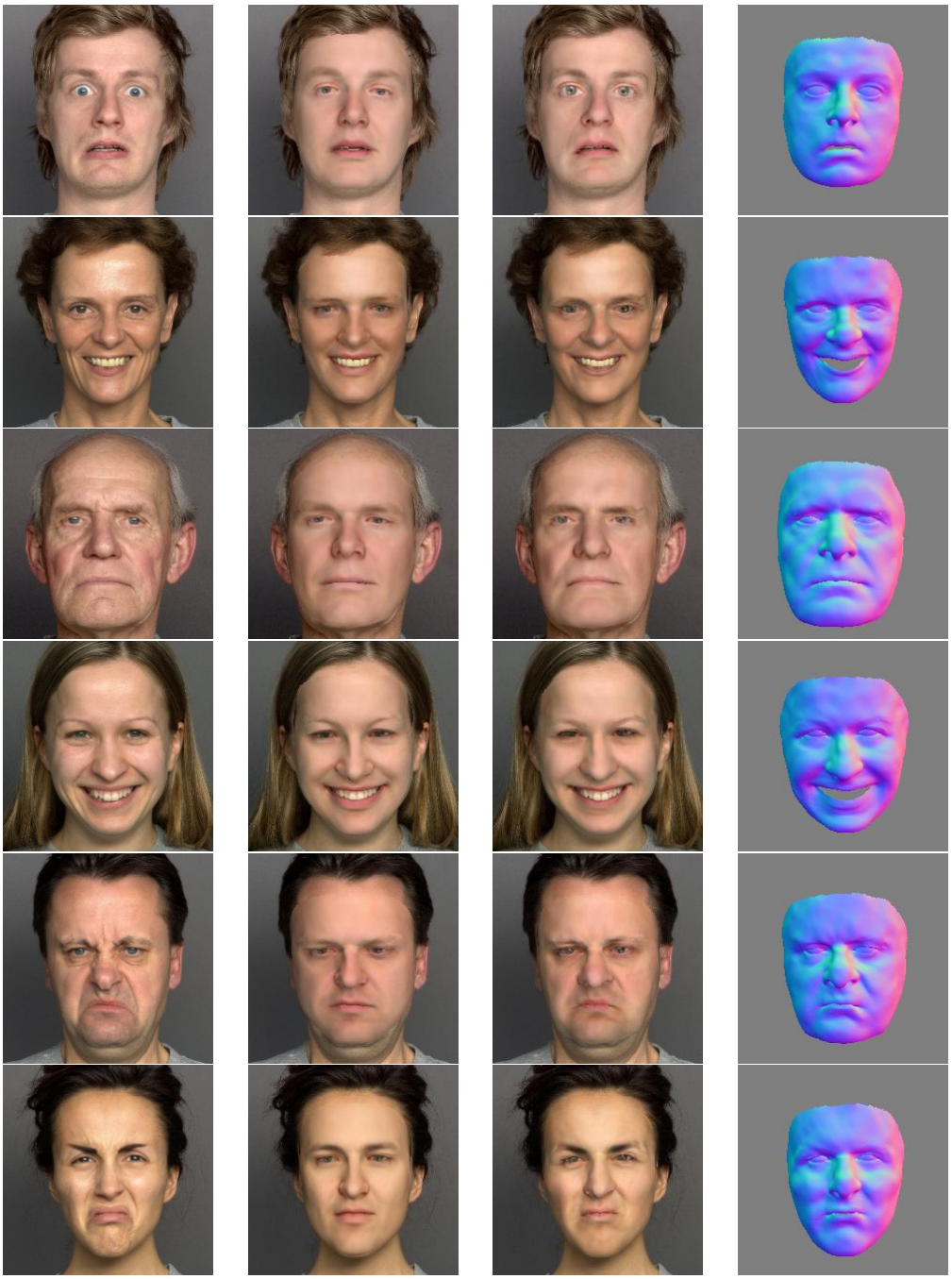
[†] Taken from [Egger et al., 2018]

no head pose perturbations (*i.e. frontal image*), and the illumination is somewhat controlled. It allows us to carefully assess the capability of the proposed method to recover mid-level facial geometry without being perturbed by external factors. Lastly, the age distribution of the subjects spans an extensive range starting from 19 years old up to 80 years old, ensuring age-based deformation is included in the evaluation data.

6.3.1 Qualitative Evaluation

For each image in the FACES dataset, the parameters $\mathcal{X}, \mathcal{D}$ are estimated using an analysis-by-synthesis fitting strategy and minimizing the objective function defined earlier in Equation 6.3 to recover mid-frequency facial geometric details embedded into a displacement map.

Figure 6.2 displays reconstructed faces with the proposed generative model $\mathcal{H}^f$ and the ResNet18-based reconstruction network defined previously in Chapter 3 for some samples of the FACES dataset. It shows that the proposed generative model with geometric corrections can recover some facial details lost with classical PCA-based models, thus indicating the use of displacement maps is appropriate within the scope of 3DMM. The added deformations are globally increasing the photorealism of synthesized images. For some cases, it restores or improves the fidelity of the facial expression, as shown in rows $1, 5, 6$. Moreover, the regions most impacted by the displacement map seem to be around the mouth, eye, and cheeks.



(a) (b) (c) (d)

Figure 6.2 – Comparison between reconstruction methods on samples from the FACES dataset. (a) Target image, (b) output of the ResNet18-based reconstruction network, (c) proposed method, and (d) normals of the refined surfaces.

6.3.2 Quantitative Evaluation

Different image quality metrics have been used to assess the images reconstructed using the proposed method formally. These metrics will only measure the quality of the appearance generated. Since no ground truth is available for the FACES dataset, no geometry assessment quality is conducted.

The first metric is the Peak Signal-to-Noise Ratio (PSNR), measuring the ratio between the signal's maximum power and the power of the corrupting noise that affects the original signal. For images, the corrupting noise is estimated by measuring the MSE between the reconstructed image $\boldsymbol{I}$ and the target image $\bar{\boldsymbol{I}}$ as defined in:

$$\mathrm{MSE}\left(\bar{\boldsymbol{I}}, \boldsymbol{I}\right) = \frac{1}{HWC} \sum_{i} \left(\bar{\boldsymbol{I}}_i - \boldsymbol{I}_i\right)^2, \tag{6.6}$$

where H, W, C are the image's dimensions, and i denotes a single element from an image channel. The PSNR is then defined as:

$$\mathrm{PSNR}\left(\bar{\boldsymbol{I}}, \boldsymbol{I}\right) = 20 \cdot \log_{10}\left(\frac{\bar{\boldsymbol{I}}_{\max}}{\sqrt{\mathrm{MSE}}}\right), \tag{6.7}$$

where $\bar{\boldsymbol{I}}_{\max}$ is the maximum possible value a pixel can have in the image, for instance, when images are coded using 8 bits per pixel, the maximum value is defined as $2^8 - 1 = 255$.

The second metric used to assess the image quality is the Structural Similarity Index Measure (SSIM) proposed by Wang *et al.*, the metric includes perceptual phenomena such as luminance and contrast masking and considers image degradation as *perceived change in structure information* [Wang et al., 2004]. The quality index between two image patches is defined as:

$$\begin{aligned}
\mathrm{SSIM}\left(\boldsymbol{y}^t, \boldsymbol{y}^p\right) &= \frac{\left(2\mu_{x^t}\mu_{y^p} + C_1\right)\left(2\sigma_{y^t y^p} + C_2\right)}{\left(\mu_{y^t}^2 + \mu_{y^p}^2 + C_1\right)\left(\sigma_{y^t}^2 + \sigma_{y^p}^2 + C_2\right)} \\
\mu_{y^{\{t,p\}}} &= \sum_i w_i y_i^{\{t,p\}} \qquad \sigma_{y^{\{t,p\}}} = \left(\sum_i w_i \left(y_i^{\{t,p\}} - \mu_{y^{\{t,p\}}}\right)^2\right)^{\frac{1}{2}} \\
\sigma_{y^t y^p} &= \sum_i w_i \left(y_i^t - \mu_{y^t}\right)\left(y_i^p - \mu_{y^p}\right) \qquad C_{\{1,2\}} = \left(K_{\{1,2\}} \cdot L\right)^2
\end{aligned} \tag{6.8}$$

where $\boldsymbol{y}^t$ is a reference patch, $\boldsymbol{y}^p$ is a reconstructed patch, $\boldsymbol{w}$ is a weighting function normalize to unit sum (*i.e.* $\sum_j w_j = 1$),and $K_{\{1,2\}} \ll 1$ are small constants to avoid instability when either $\left(\mu_{y^t}^2 + \mu_{y^p}^2\right)$ or $\left(\sigma_{y^t}^2 + \sigma_{y^p}^2\right)$ are very close to zero. In the original work, a symmetric circular Gaussian weighting function with a standard deviation of 1.5 is used to compute the index on

Table 6.2 – Average image quality metrics on the FACES dataset

Method	**MSE** ($\times 10^{-3}$)	**PSNR** (dB)	**MSSIM**	$\boldsymbol{D}_{\text{RGB}}$ ($\times 10^{-3}$)
ResNet18	3.14 ± 3.1	26.09 ± 2.68	0.78 ± 0.05	98.47 ± 30.61
EfficientNetV2-B2	3.15 ± 3.3	26.18 ± 2.28	0.78 ± 0.05	96.79 ± 31.80
Displacement maps	$\mathbf{2.67 \pm 3.6}^{\star}$	$\mathbf{27.47 \pm 3.42}^{\star}$	$\mathbf{0.81 \pm 0.05}^{\star}$	$\mathbf{78.68 \pm 32.84}^{\star}$

* p-value < 0.05

patches of dimensions 11×11, and the constants are set to $K_1 = 0.01, K_2 = 0.03$.

The overall quality measure of the entire image is then defined by averaging every patch's score together. The final mean SSIM index is given in:

$$\text{MSSIM}\left(\bar{\boldsymbol{I}}, \boldsymbol{I}\right) = \frac{1}{M} \sum_{j} \text{SSIM}\left(\bar{\boldsymbol{I}}_j, \boldsymbol{I}_j\right), \tag{6.9}$$

where $\bar{\boldsymbol{I}}$ is the reference image, $\boldsymbol{I}$ denotes the reconstructed image, $\bar{\boldsymbol{I}}_j, \boldsymbol{I}_j$ are the image content of the j-th local window, and M indicates the total number of windows in the image.

As mentioned earlier, no geometrical evaluation has been conducted. However, one can highlight the geometric discrepancies by measuring the signed distance between pixels from the target and the reconstruction in the RGB space. The signed pixel-wise distance can be computed as follow:

$$d_{\text{RGB}}\left(\boldsymbol{y}^t, \boldsymbol{y}^p\right) = \delta \|\boldsymbol{y}^t - \boldsymbol{y}^p\|, \qquad \delta = \begin{cases} +1 & \text{if} \left(\boldsymbol{y}^p - \boldsymbol{y}^t\right) \cdot \boldsymbol{y}^t \geq 0.0, \\ -1 & \text{otherwise,} \end{cases} \tag{6.10}$$

where $\boldsymbol{y}^t$ is the true pixel value, $\boldsymbol{y}^p$ is the predicted pixel value, and δ is an indicator function defining the sign of the error based on whether the predicted value is further away or not from the true value. The distance for an entire image is then computed by averaging the absolute values of d_{RGB} of every pixel in the foreground region $\mathcal{F}$:

$$D_{\text{RGB}}\left(\bar{\boldsymbol{I}}, \boldsymbol{I}\right) = \frac{1}{|\mathcal{F}|} \sum_{j \in \mathcal{F}} \left| d_{\text{RGB}}\left(\bar{\boldsymbol{I}}_j, \boldsymbol{I}_j\right) \right|. \tag{6.11}$$

All the metrics listed before are evaluated on the FACES datasets; their values are grouped and reported in Table 6.2. The displacement maps-based model is compared with the two baselines established in Chapter 3. The results confirm that the proposed method performs better, providing the lowest reconstruction errors and the highest image quality indexes.

Moreover, the false-color signed distance maps shown in Figure 6.3 indicate that the reconstructed facial geometry is richer and includes mid-frequency geometry details compared to

the one generated by the ResNet18-based approach. However, our solution still lacks high-frequency details such as tiny wrinkles (*i.e. Row 2 and 4 around the mouth*), indicating that vertex-wise corrections are insufficient. The whole displacement map should be transferred to the model through interpolation. The level of details that can be recovered with the proposed method is directly linked to the vertex density in the mesh and the length of the edges between each vertex. The finesse of the details represented by two neighboring vertices will be much higher if the distance between them is small. For instance, consider that the vertices on the forehead are evenly spaced on a regular grid with a distance of 10mm between each one of them. With this configuration, it is impossible to accurately represent wrinkles linked to facial expressions (*i.e. frowning*) because the spatial resolution of the mesh is far too large to pick such tiny displacements (*i.e. few millimeters magnitude*).

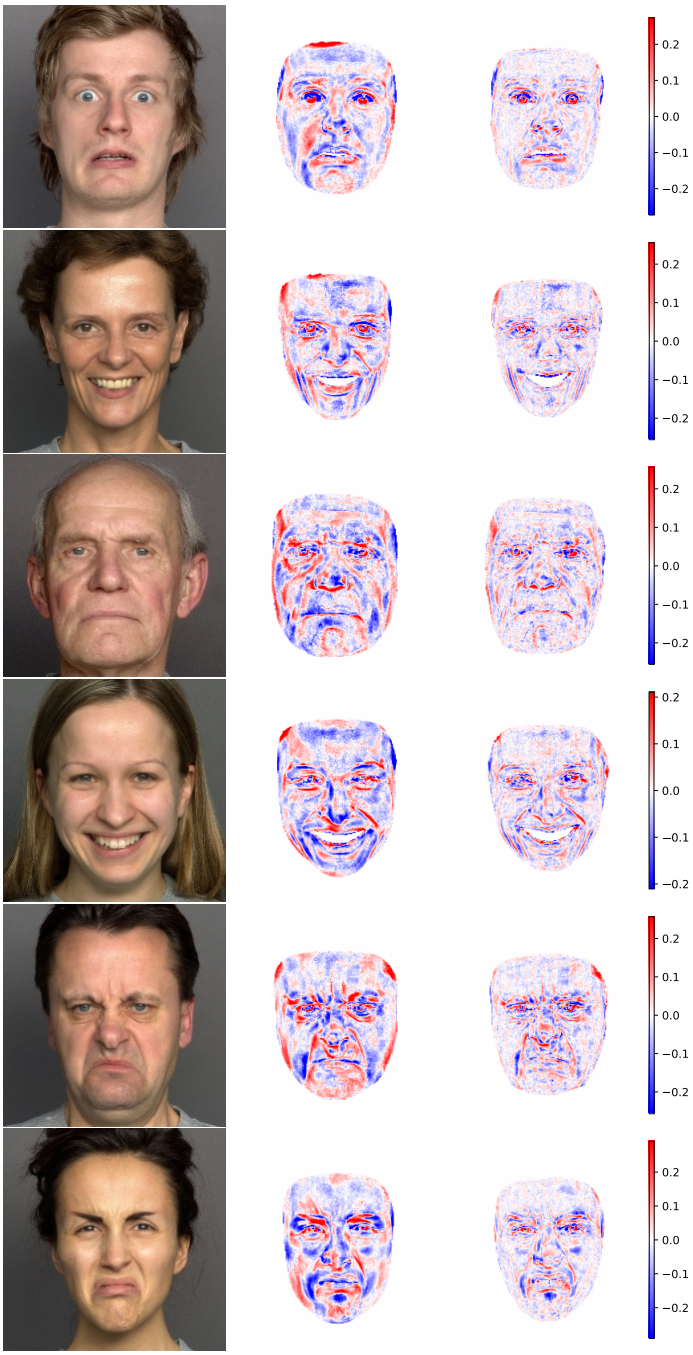
6.4 Summary

In this chapter, we have presented a method to improve the quality of the facial geometry during the face reconstruction process by adding a displacement map on top of an existing 3DMM. The displacement map is estimated at the same time as the coarse geometry during the fitting step. To this end, extra constraints are added to the objective function minimized by gradient descent within an analysis-by-synthesis framework.

The proposed solution of adding correction only at the vertex-level works because the original mesh, the BFM in our experiment, is densely tessellated. Adding displacement maps on a dense mesh effectively increases the fidelity of the reconstructed surfaces and adds more realism overall. Some of the depth cues linked to facial expressions are effectively restored with our approach. However, as mentioned earlier, one limitation of the method is that vertex-wise corrections are not enough to recover every facial geometric detail. The whole displacement maps should be transferred to the 3D model through interpolation to increase the spatial resolutions of the corrections similar to [Feng et al., 2021].

The segmentation mask provides regions where the photometric error can be evaluated and be meaningful to the underlying appearance model and solves the problem with occlusions. However, regarding the geometry aspect, the segmentation mask effectively blocks regions from moving. Thus geometrical aberrations appear at the boundaries of these regions. The proposed method is, by design, sensitive to occluders.

The solution presented in this chapter can effectively recover the missing facial details of a PCA-based model and shows promising results. However, it is sensitive to majors limiting factors, opening new research directions for future work.



(a) (b)

Figure 6.3 – False-color signed distance maps (red/blue = brighter/darker than reference) highlight the geometric discrepancies between the ResNet18-based reconstruction network ((b)-left) and the proposed method ((b)-right).

7 Conclusion

This final chapter of this thesis will summarize the contributions and discuss the benefits and limitations of the proposed ideas and concepts to tackle the limiting factors identified in Section 2.6, namely the consistency of the parameters across head pose variations, the robustness of the reconstruction network against image resolutions, and the loss of the mid-level facial details in statistical models. The discussion continues on the perspectives and possibly new research directions for future work.

Chapter 1, introduces multiple methods linked to the monocular 3D face reconstruction task but not directly involved in the reconstruction process. These algorithms are either used to prepare the training data, build statistical models, extract the information required by the training process, or during the evaluation step. We presented two components commonly used in standard facial image analysis frameworks, specifically face detection and face landmarks localization algorithms. Then we introduced a semantic segmentation network, namely the BiSeNet, used to create probability maps defining regions where the photometric can reliably be computed to be relevant to the 3DMM. The last part discussed the problem of aligning surfaces in a rigid and a nonrigid way and reviewed each element of the ICP algorithm. Over the years, its simplicity and yet effectiveness make it the default goto alignment method.

In Chapter 2, we laid down the groundwork for monocular 3D face reconstruction. For each component in the reconstruction pipeline, we presented a comprehensive literature review presenting the multiple options available to model a human face, its interaction with the light, and the ways 3D scenes can be turned into images. We also talked about the different objective functions and fitting strategies presented over the last decades and discussed their benefits. We conclude this chapter with a discussion on the limiting factors identified in the current methodology from the literature. The first observation is the incapability of reconstruction networks to predict consistent face parameterization across large head pose variations. Second, reconstruction networks are sensitive to image quality and, more specifically, to image resolutions. Furthermore, the last finding is the loss of mid-level facial details due to how PDMs are constructed. Our contributions are built upon these three observations.

To fairly assess the added value of the proposed methods, we implemented in Chapter 3 a simple reconstruction network based on two commonly used architectures. The details are provided with respect to the type of statistical face models, how lighting is represented, the selected architecture, the objective functions, and the data used to train these reconstruction networks. To assess the quality of the reconstructed faces, we have presented three evaluation protocols used in the literature and established baseline results that are compared against our contributions.

Our first contribution is presented in Chapter 4, where we investigate the issue of face representation consistency across head pose variations. With the current training mechanisms, there is no way preventing the reconstruction network from predicting different parameterizations of the same faces when they are seen under different viewpoints. We propose to impose representation similarity between different viewpoints through the use of a contrastive learning framework. The image pairs required to learn contrastively are generated on the fly using the generative face model and randomly sampling the pose space. This mechanism helps to learn a pose-invariant feature subspace out of which the parameters of the statistical face models are regressed. We have shown through experiments that the predicted face representation is more consistent across head pose variations. However, the performances of the pose-aware reconstruction network were slightly lower than the baseline, indicating that the added mechanism penalizes the reconstruction task. This reduction in performance could be an indicator of the augmentation method not being effective enough.

Chapter 5 presents our second contribution, where the impact of the image resolution is studied. We proposed to use the resolution-aware residual block of [Xu et al., 2021] in the backbone network to extract image representations that are similar across a wide range of image resolutions. We implemented and trained a resolution-aware reconstruction network to work with image sizes in the range of $[32, 224]$ pixels. While the modification to the original residual block is straightforward, the training strategy and cost are drastically increased compared to the baseline network. Moreover, the quality of the reconstructed facial geometry and appearance is slightly below our baseline. On the other hand, the performances of the resolution-aware system are pretty consistent across the different image resolutions.

Our last contribution is introduced in Chapter 6, where we tried to recover the mid-level facial details, such as the wrinkles, that are missing from standard statistical face models. The way how statistical shape models are built, mainly using PCA-based techniques, the learned distributions contains only low-frequency deformations. Thus tiny details are missing from smooth generated surfaces. We propose representing mid-level facial details using displacement maps on top of a classical statistical shape model to overcome this limitation. The person-specific displacement maps are directly learned from the data using an analysis-by-synthesis fitting strategy. The effectiveness of the proposed facial details representation has been demonstrated qualitatively and quantitatively on a dataset of facial expressions images. Our experiments have shown that the displacement maps are capable of representing the missing details. However, the experiments also showed that vertex-wise refinement does not

have a high enough spatial resolution, even with densely tessellated mesh. Therefore, very tiny details are still missing.

Future Perspectives

Over the recent years, deep learning has shown a tremendous capability to learn visual representations. In the context of end-to-end 3D face reconstruction, the major limiting factor resides in the statistical face models used in the decoder to represent the identity, the expression, and the appearance of human faces. The publicly available models presented in Section 2.1.4 have intrinsic biases by design (*i.e. ethnicity, age*). These biases are present due to the type of data used to learn them. Most of the time, only a small number of people have been used. Therefore only a tiny portion of the actual distribution is represented. However, building a bias-free statistical model might be impracticable due to the complexity of the data collection and acquisition (*i.e. true albedo is very complicated to measure*). An alternative could be to learn the model directly from images that are simpler to acquire but requires ways to ensure that the learned distribution represents the object and not external factors such as occluders. Creating such models would have a significant impact on the computer vision community.

Most of the 3D face reconstruction pipelines assume that the facial albedo is purely Lambertian. However, this strong assumption does not hold, as shown by recent works [Smith et al., 2020, Dib et al., 2021b]. The interaction between the light and the human skin is, in fact, much more complex. It would be interesting to investigate how human skin is rendered traditionally within computer graphics applications and see if their techniques can be applied in the context of 3D face reconstruction. There is existing work in which they integrate the subsurface scattering property of the human skin into BRDF function [d'Eon,Eugene et al., 2007]. Investigating if such formulation can be applied to the current reconstruction pipelines would be of great interest. Having more realistic models of the interaction between the human face and the light would significantly improve the quality and fidelity of the reconstructions.

Bibliography

[Aldrian and Smith, 2012] Aldrian, O. and Smith, W. A. P. (2012). Model-Based Ambient Occlusion for Inverse Rendering. In Hutchison, D., Kanade, T., Kittler, J., Kleinberg, J. M., Mattern, F., Mitchell, J. C., Naor, M., Nierstrasz, O., Pandu Rangan, C., Steffen, B., Sudan, M., Terzopoulos, D., Tygar, D., Vardi, M. Y., Weikum, G., Campilho, A., and Kamel, M., editors, *Image Analysis and Recognition*, volume 7324, pages 338–347. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Amberg et al., 2007] Amberg, B., Romdhani, S., and Vetter, T. (2007). Optimal Step Nonrigid ICP Algorithms for Surface Registration. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8.

[Arun et al., 1987] Arun, K. S., Huang, T. S., and Blostein, S. D. (1987). Least-Squares Fitting of Two 3-D Point Sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9(5):698–700.

[Bagdanov et al., 2011] Bagdanov, A. D., Del Bimbo, A., and Masi, I. (2011). The florence 2D/3D hybrid face dataset. In *Proceedings of the 2011 Joint ACM Workshop on Human Gesture and Behavior Understanding - J-HGBU '11*, page 79, Scottsdale, Arizona, USA. ACM Press.

[Bazarevsky et al., 2019] Bazarevsky, V., Kartynnik, Y., Vakunov, A., Raveendran, K., and Grundmann, M. (2019). BlazeFace: Sub-millisecond Neural Face Detection on Mobile GPUs. *arXiv:1907.05047 [cs]*.

[Bentley, 1975] Bentley, J. L. (1975). Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517.

[Berlinet and Thomas-Agnan, 2004] Berlinet, A. and Thomas-Agnan, C. (2004). *Reproducing Kernel Hilbert Spaces in Probability and Statistics*. Springer US, Boston, MA.

[Besl and McKay, 1992] Besl, P. J. and McKay, N. D. (1992). A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14:239–256.

[Blanz and Vetter, 1999] Blanz, V. and Vetter, T. (1999). A Morphable Model for the Synthesis of 3D Faces. In *Proceedings of the 26th Annual Conference on Computer Graphics and Inter-*

active Techniques, SIGGRAPH '99, pages 187–194, New York, NY, USA. ACM Press/Addison-Wesley Publishing Co.

[Booth et al., 2017] Booth, J., Antonakos, E., Ploumpis, S., Trigeorgis, G., Panagakis, Y., and Zafeiriou, S. (2017). 3D Face Morphable Models" In-the-Wild". In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Booth et al., 2016] Booth, J., Roussos, A., Zafeiriou, S., Ponniah, A., and Dunaway, D. (2016). A 3D Morphable Model Learnt from 10,000 Faces. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5543–5552, Las Vegas, NV. IEEE.

[Bouaziz et al., 2013] Bouaziz, S., Wang, Y., and Pauly, M. (2013). Online modeling for realtime facial animation. *ACM Transactions on Graphics (TOG)*, 32(4):40.

[Bulat and Tzimiropoulos, 2017a] Bulat, A. and Tzimiropoulos, G. (2017a). Binarized Convolutional Landmark Localizers for Human Pose Estimation and Face Alignment with Limited Resources. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 3726–3734.

[Bulat and Tzimiropoulos, 2017b] Bulat, A. and Tzimiropoulos, G. (2017b). How Far are We from Solving the 2D & 3D Face Alignment Problem? (and a Dataset of 230,000 3D Facial Landmarks). In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 1021–1030, Venice. IEEE.

[Bulat et al., 2018] Bulat, A., Yang, J., and Tzimiropoulos, G. (2018). To Learn Image Super-Resolution, Use a GAN to Learn How to Do Image Degradation First. In Ferrari, V., Hebert, M., Sminchisescu, C., and Weiss, Y., editors, *Computer Vision – ECCV 2018*, volume 11210, pages 187–202. Springer International Publishing, Cham.

[Cao et al., 2014] Cao, C., Weng, Y., Zhou, S., Tong, Y., and Zhou, K. (2014). Facewarehouse: A 3d facial expression database for visual computing. *Visualization and Computer Graphics, IEEE Transactions on*, 20(3):413–425.

[Cao et al., 2018] Cao, Q., Shen, L., Xie, W., Parkhi, O. M., and Zisserman, A. (2018). VGGFace2: A Dataset for Recognising Faces across Pose and Age. In *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*, pages 67–74, Xi'an. IEEE.

[Chen et al., 2019a] Chen, K., Ming, L., and Modersitzki, J. (2019a). Chapter 15 - Image and surface registration. In *Processing, Analyzing and Learning of Images, Shapes, and Forms: Part 2*, volume 20 of *Handbook of Numerical Analysis*, pages 579–611. Elsevier.

[Chen et al., 2019b] Chen, W., Gao, J., Ling, H., Smith, E. J., Lehtinen, J., Jacobson, A., and Fidler, S. (2019b). Learning to Predict 3D Objects with an Interpolation-based Differentiable Renderer. In *Advances In Neural Information Processing Systems*, page 12.

[Chen et al., 2020] Chen, X., Fan, H., Girshick, R., and He, K. (2020). Improved Baselines with Momentum Contrastive Learning. *arXiv:2003.04297 [cs]*.

[Chen and He, 2021] Chen, X. and He, K. (2021). Exploring Simple Siamese Representation Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15750–15758.

[Chen and Medioni, 1991] Chen, Y. and Medioni (1991). Object modeling by registration of multiple range images. In *Proceedings. 1991 IEEE International Conference on Robotics and Automation*, volume 3, pages 2724–2729.

[Chetverikov et al., 2002] Chetverikov, D., Svirko, D., Stepanov, D., and Krsek, P. (2002). The Trimmed Iterative Closest Point algorithm. In *Object Recognition Supported by User Interaction for Service Robots*, volume 3, pages 545–548, Quebec City, Que., Canada. IEEE Comput. Soc.

[Chollet, 2017] Chollet, F. (2017). Xception: Deep Learning With Depthwise Separable Convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, page 8.

[Deering et al., 1988] Deering, M., Winner, S., Schediwy, B., Duffy, C., and Hunt, N. (1988). The triangle processor and normal vector shader: A VLSI system for high performance graphics. *ACM SIGGRAPH Computer Graphics*, 22(4):21–30.

[Defferrard et al., 2016] Defferrard, M., Bresson, X., and Vandergheynst, P. (2016). Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, pages 3844–3852.

[Deng et al., 2020] Deng, J., Guo, J., Zhou, Y., Yu, J., Kotsia, I., and Zafeiriou, S. (2020). RetinaFace: Single-stage Dense Face Localisation in the Wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5203–5212.

[Deng et al., 2019] Deng, Y., Yang, J., Xu, S., Chen, D., Jia, Y., and Tong, X. (2019). Accurate 3D Face Reconstruction with Weakly-Supervised Learning: From Single Image to Image Set. *arXiv:1903.08527 [cs]*.

[d'Eon,Eugene et al., 2007] d'Eon,Eugene, Luebke, David, and Enderton, Eric (2007). Efficient Rendering of Human Skin. In *Proceedings of the 18th Eurographics Conference on Rendering Techniques*, pages 147–157.

[Dib et al., 2021a] Dib, A., Bharaj, G., Ahn, J., Thébault, C., Gosselin, P., Romeo, M., and Chevallier, L. (2021a). Practical Face Reconstruction via Differentiable Ray Tracing. *Computer Graphics Forum*, 40(2):153–164.

[Dib et al., 2021b] Dib, A., Thebault, C., Ahn, J., Gosselin, P.-H., Theobalt, C., and Chevallier, L. (2021b). Towards High Fidelity Monocular Face Reconstruction with Rich Reflectance using Self-supervised Learning and Ray Tracing. *arXiv:2103.15432 [cs]*.

[Dodge and Karam, 2016] Dodge, S. and Karam, L. (2016). Understanding how image quality affects deep neural networks. In *2016 Eighth International Conference on Quality of Multimedia Experience (QoMEX)*, pages 1–6, Lisbon, Portugal. IEEE.

[Dorai et al., 1998] Dorai, C., Wang, G., Jain, A. K., and Mercer, C. (1998). Registration and integration of multiple object views for 3D model construction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(1):83–89.

[Ebner et al., 2010] Ebner, N. C., Riediger, M., and Lindenberger, U. (2010). FACES—A database of facial expressions in young, middle-aged, and older women and men: Development and validation. *Behavior Research Methods*, 42(1):351–362.

[Egger et al., 2017] Egger, B., Schönborn, S., Blumer, C., and Vetter, T. (2017). Probabilistic Morphable Models. In *Statistical Shape and Deformation Analysis*, pages 115–135. Elsevier.

[Egger et al., 2018] Egger, B., Schönborn, S., Schneider, A., Kortylewski, A., Morel-Forster, A., Blumer, C., and Vetter, T. (2018). Occlusion-Aware 3D Morphable Models and an Illumination Prior for Face Image Analysis. *International Journal of Computer Vision*.

[Egger et al., 2020] Egger, B., Smith, W. A. P., Tewari, A., Wuhrer, S., Zollhoefer, M., Beeler, T., Bernard, F., Bolkart, T., Kortylewski, A., Romdhani, S., Theobalt, C., Blanz, V., and Vetter, T. (2020). 3D Morphable Face Models—Past, Present, and Future. *ACM Transactions on Graphics*, 39(5):1–38.

[Ekman and Friesen, 1976] Ekman, P. and Friesen, W. V. (1976). Measuring facial movement. *Environmental Psychology and Nonverbal Behavior*, 1(1):56–75.

[Feng et al., 2021] Feng, Y., Feng, H., Black, M. J., and Bolkart, T. (2021). Learning an animatable detailed 3D face model from in-the-wild images. *ACM Transactions on Graphics*, 40(4):1–13.

[Freund and Schapire, 1997] Freund, Y. and Schapire, R. E. (1997). A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of Computer and System Sciences*, 55(1):119–139.

[Gao et al., 2020] Gao, Z., Zhang, J., Guo, Y., Ma, C., Zhai, G., and Yang, X. (2020). Semi-supervised 3D Face Representation Learning from Unconstrained Photo Collections. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1426–1435, Seattle, WA, USA. IEEE.

[Gary B. Huang et al., 2008] Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller (2008). Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments. In *Workshop on Faces in 'Real-Life' Images: Detection, Alignment, and Recognition*.

[Gecer et al., 2019] Gecer, B., Ploumpis, S., Kotsia, I., and Zafeiriou, S. (2019). GANFIT: Generative Adversarial Network Fitting for High Fidelity 3D Face Reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Gelfand et al., 2003] Gelfand, N., Ikemoto, L., Rusinkiewicz, S., and Levoy, M. (2003). Geometrically stable sampling for the ICP algorithm. In *Fourth International Conference on 3-D Digital Imaging and Modeling, 2003. 3DIM 2003. Proceedings.*, pages 260–267.

[Genova et al., 2018] Genova, K., Cole, F., Maschinot, A., Sarna, A., Vlasic, D., and Freeman, W. T. (2018). Unsupervised Training for 3D Morphable Model Regression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Gerig et al., 2018] Gerig, T., Morel-Forster, A., Blumer, C., Egger, B., Luthi, M., Schonborn, S., and Vetter, T. (2018). Morphable Face Models - An Open Framework. In *2018 13th IEEE International Conference on Automatic Face Gesture Recognition (FG 2018*, pages 75–82.

[Godin et al., 1994] Godin, G., Rioux, M., and Baribeau, R. (1994). Three-dimensional registration using range and intensity information. In El-Hakim, S. F., editor, *Videometrics III*, pages 279–290, Boston, MA.

[Gogić et al., 2021] Gogić, I., Ahlberg, J., and Pandžić, I. S. (2021). Regression-based methods for face alignment: A survey. *Signal Processing*, 178:107755.

[Grill et al., 2020] Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P. H., Buchatskaya, E., Doersch, C., Pires, B. A., Guo, Z. D., Azar, M. G., Piot, B., Kavukcuoglu, K., Munos, R., and Valko, M. (2020). Bootstrap your own latent: A new approach to self-supervised Learning. *arXiv:2006.07733 [cs, stat]*.

[Gross et al., 2010] Gross, R., Matthews, I., Cohn, J., Kanade, T., and Baker, S. (2010). Multi-PIE. *Image and Vision Computing*, 28(5):807–813.

[Guo et al., 2019] Guo, Y., Zhang, J., Cai, J., Jiang, B., and Zheng, J. (2019). CNN-Based Real-Time Dense Face Reconstruction with Inverse-Rendered Photo-Realistic Face Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(6):1294–1307.

[He et al., 2020] He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. (2020). Momentum Contrast for Unsupervised Visual Representation Learning. *arXiv:1911.05722 [cs]*.

[He et al., 2016a] He, K., Zhang, X., Ren, S., and Sun, J. (2016a). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778.

[He et al., 2016b] He, K., Zhang, X., Ren, S., and Sun, J. (2016b). Identity Mappings in Deep Residual Networks. In Leibe, B., Matas, J., Sebe, N., and Welling, M., editors, *Computer Vision – ECCV 2016*, volume 9908, pages 630–645. Springer International Publishing, Cham.

[Henderson and Ferrari, 2019] Henderson, P. and Ferrari, V. (2019). Learning Single-Image 3D Reconstruction by Generative Modelling of Shape, Pose and Shading. *International Journal of Computer Vision*.

[Huang et al., 2017] Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A., Fathi, A., Fischer, I., Wojna, Z., Song, Y., Guadarrama, S., and Murphy, K. (2017). Speed/Accuracy Trade-Offs for Modern Convolutional Object Detectors. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3296–3297, Honolulu, HI. IEEE.

[Huber et al., 2016] Huber, P., Hu, G., Tena, R., Mortazavian, P., Koppen, W. P., Christmas, W. J., Rätsch, M., and Kittler, J. (2016). A Multiresolution 3D Morphable Face Model and Fitting Framework:. In *Proceedings of the 11th Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, pages 79–86, Rome, Italy. SCITEPRESS - Science and Technology Publications.

[Jiang et al., 2021] Jiang, D., Jin, Y., Deng, R., Tong, R., Zhang, F., Yai, Y., and Tang, M. (2021). Reconstructing Recognizable 3D Face Shapes based on 3D Morphable Models. *arXiv:2104.03515 [cs]*.

[Karras et al., 2018] Karras, T., Aila, T., Laine, S., and Lehtinen, J. (2018). Progressive Growing of GANs for Improved Quality, Stability, and Variation. In *International Conference on Learning Representations*.

[Kato et al., 2018] Kato, H., Ushiku, Y., and Harada, T. (2018). Neural 3D Mesh Renderer. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3907–3916, Salt Lake City, UT. IEEE.

[Kingma and Ba, 2015] Kingma, D. P. and Ba, J. (2015). Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, {ICLR} 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

[Laine et al., 2020] Laine, S., Hellsten, J., Karras, T., Seol, Y., Lehtinen, J., and Aila, T. (2020). Modular Primitives for High-Performance Differentiable Rendering. *Association for Computing Machinery*, 39(6).

[Le-Khac et al., 2020] Le-Khac, P. H., Healy, G., and Smeaton, A. F. (2020). Contrastive Representation Learning: A Framework and Review. *IEEE Access*, 8:193907–193934.

[Lee et al., 2020] Lee, C.-H., Liu, Z., Wu, L., and Luo, P. (2020). MaskGAN: Towards Diverse and Interactive Facial Image Manipulation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Lewis, J. P. et al., 2014] Lewis, J. P., Anjyo, Ken, Rhee, Taehyun, Zhang, Mengjie, Pighin, Fred, and Deng, Zhigang (2014). Practice and Theory of Blendshape Facial Models. The Eurographics Association.

[Li et al., 2017] Li, T., Bolkart, T., Black, M. J., Li, H., and Romero, J. (2017). Learning a model of facial shape and expression from 4D scans. *ACM Transactions on Graphics*, 36(6):1–17.

[Li et al., 2019] Li, T.-M., Aittala, M., Durand, F., and Lehtinen, J. (2019). Differentiable Monte Carlo ray tracing through edge sampling. *ACM Transactions on Graphics*, 37(6):1–11.

[Liu and Matthies, 2019] Liu, D. and Matthies, H. G. (2019). Pivoted Cholesky decomposition by Cross Approximation for efficient solution of kernel systems. *arXiv:1505.06195 [cs, math]*.

[Liu et al., 2020] Liu, L., Jiang, H., He, P., Chen, W., Liu, X., Gao, J., and Han, J. (2020). On the Variance of the Adaptive Learning Rate and Beyond. *arXiv:1908.03265 [cs, stat]*.

[Liu et al., 2019] Liu, S., Li, T., Chen, W., and Li, H. (2019). Soft Rasterizer: A Differentiable Renderer for Image-based 3D Reasoning. *arXiv:1904.01786 [cs]*.

[Liu et al., 2016] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. (2016). SSD: Single Shot MultiBox Detector. In *ECCV*.

[Liu et al., 2015] Liu, Z., Luo, P., Wang, X., and Tang, X. (2015). Deep Learning Face Attributes in the Wild. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 3730–3738, Santiago, Chile. IEEE.

[Loper and Black, 2014] Loper, M. M. and Black, M. J. (2014). OpenDR: An Approximate Differentiable Renderer. In Fleet, D., Pajdla, T., Schiele, B., and Tuytelaars, T., editors, *Computer Vision – ECCV 2014*, volume 8695, pages 154–169. Springer International Publishing, Cham.

[Low, 2004] Low, K.-L. (2004). Linear Least-Squares Optimization for Point-to-Plane ICP Surface Registration. Technical report.

[Lowe, 2004] Lowe, D. G. (2004). Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60:91–110.

[Luo et al., 2016] Luo, W., Li, Y., Urtasun, R., and Zemel, R. (2016). Understanding the Effective Receptive Field in Deep Convolutional Neural Networks. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 4905–4913.

[Luthi et al., 2017] Luthi, M., Gerig, T., Jud, C., and Vetter, T. (2017). Gaussian Process Morphable Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1.

[Masuda et al., 1996] Masuda, T., Sakaue, K., and Yokoya, N. (1996). Registration and integration of multiple range images for 3-D model construction. In *Proceedings of 13th International Conference on Pattern Recognition*, pages 879–883.

[Matthews and Baker, 2004] Matthews, I. and Baker, S. (2004). Active Appearance Models Revisited. *International Journal of Computer Vision*, 60(2):135–164.

[Meagher, 1982] Meagher, D. (1982). Geometric modeling using octree encoding. *Computer Graphics and Image Processing*, 19(2):129–147.

[Messer et al., 2009] Messer, K., Matas, J., Kittler, J., Luettin, J., and Maitre, G. (2009). XM2VTSDB: The Extended M2VTS Database. In *Proc. Second International Conference on Audio and Video-Based Biometric Person Authentication (AVBPA'99)*, page 6.

[Mikkelsen, 2008] Mikkelsen, M. (2008). Simulation of Wrinkled Surfaces Revisited.

[Minaee et al., 2021] Minaee, S., Luo, P., Lin, Z., and Bowyer, K. (2021). Going Deeper Into Face Detection: A Survey. *arXiv:2103.14983 [cs]*.

[Mortazavian et al., 2012] Mortazavian, P., Kittler, J., and Christmas, W. (2012). 3D morphable model fitting for low-resolution facial images. In *Biometrics (ICB), 2012 5th IAPR International Conference On*, pages 132–138. IEEE.

[Newell et al., 2016] Newell, A., Yang, K., and Deng, J. (2016). Stacked Hourglass Networks for Human Pose Estimation. In *Computer Vision – ECCV 2016*, pages 483–499. Springer International Publishing.

[Parkhi et al., 2015] Parkhi, O. M., Vedaldi, A., and Zisserman, A. (2015). Deep Face Recognition. In *Procedings of the British Machine Vision Conference 2015*, pages 41.1–41.12, Swansea. British Machine Vision Association.

[Paysan et al., 2009] Paysan, P., Knothe, R., Amberg, B., Romdhani, S., and Vetter, T. (2009). A 3D Face Model for Pose and Illumination Invariant Face Recognition. pages 296–301. IEEE.

[Pomerleau et al., 2013] Pomerleau, F., Colas, F., Siegwart, R., and Magnenat, S. (2013). Comparing ICP variants on real-world data sets. *Autonomous Robots*, pages 133–148.

[Qu et al., 2015] Qu, C., Gao, H., Monari, E., Beyerer, J., and Thiran, J.-P. (2015). Towards robust cascaded regression for face alignment in the wild. In *2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1–9.

[Ramamoorthi, 2006] Ramamoorthi, R. (2006). Modeling illumination variation with spherical harmonics. *Face Processing: Advanced Modeling Methods*, pages 385–424.

[Ramamoorthi and Hanrahan, 2001] Ramamoorthi, R. and Hanrahan, P. (2001). An efficient representation for irradiance environment maps. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, pages 497–500. ACM.

[Ranjan et al., 2018] Ranjan, A., Bolkart, T., Sanyal, S., and Black, M. J. (2018). Generating 3D Faces Using Convolutional Mesh Autoencoders. In Ferrari, V., Hebert, M., Sminchisescu, C., and Weiss, Y., editors, *Proceedings of the European Conference on Computer Vision (ECCV)*.

[Rasmussen and Williams, 2006] Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian Processes for Machine Learning*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, Mass.

[Ren et al., 2015] Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

[Romdhani and Vetter, 2005] Romdhani, S. and Vetter, T. (2005). Estimating 3D shape and texture using pixel intensity, edges, specular highlights, texture constraints and a prior. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference On*, volume 2, pages 986–993. IEEE.

[Rusinkiewicz and Levoy, 2001] Rusinkiewicz, S. and Levoy, M. (2001). Efficient Variants of the ICP Algorithm. In *Proceedings Third International Conference on 3-D Digital Imaging and Modeling*, pages 145–152.

[Russakovsky et al., 2015] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3):211–252.

[Sagonas, 2016] Sagonas, C. (2016). 300 Faces In-The-Wild Challenge: Database and results. *Image and Vision Computing*, page 16.

[Schneider et al., 2017] Schneider, A., Schonborn, S., Egger, B., Frobeen, L., and Vetter, T. (2017). Efficient Global Illumination for Morphable Models. In *International Conference on Computer Vision (ICCV)*, pages 3885–3893. IEEE.

[Schönborn et al., 2015] Schönborn, S., Egger, B., Forster, A., and Vetter, T. (2015). Background modeling for generative image models. *Computer Vision and Image Understanding*, 136:117–127.

[Schönborn et al., 2017] Schönborn, S., Egger, B., Morel-Forster, A., and Vetter, T. (2017). Markov Chain Monte Carlo for Automated Face Image Analysis. *International Journal of Computer Vision*, 123(2):160–183.

[Schroff et al., 2015] Schroff, F., Kalenichenko, D., and Philbin, J. (2015). FaceNet: A Unified Embedding for Face Recognition and Clustering. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 815–823.

[Shen et al., 2015] Shen, J., Zafeiriou, S., Chrysos, G. G., Kossaifi, J., Tzimiropoulos, G., and Pantic, M. (2015). The First Facial Landmark Tracking in-the-Wild Challenge: Benchmark and Results. In *2015 IEEE International Conference on Computer Vision Workshop (ICCVW)*, pages 1003–1011.

[Shijie Hao et al., 2020] Shijie Hao, Zhou, Yuan, and Guo, Yanrong (2020). A Brief Survey on Semantic Segmentation with Deep Learning. *Neurocomputing*, 406:302–321.

[Sloan et al., 2002] Sloan, P.-P., Kautz, J., and Snyder, J. (2002). Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. *ACM Transactions on Graphics*, 21(3):527–536.

[Smith et al., 2020] Smith, W. A. P., Seck, A., Dee, H., Tiddeman, B., Tenenbaum, J. B., and Egger, B. (2020). A Morphable Face Albedo Model. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5010–5019, Seattle, WA, USA. IEEE.

[Stein and Medioni, 1992] Stein, F. and Medioni, G. (1992). Structural indexing: Efficient 3-D object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14:125–145.

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

[Sutherland et al., 2020] Sutherland, S., Egger, B., and Tenenbaum, J. (2020). Building 3D Morphable Models from a Single Scan. *arXiv:2011.12440 [cs]*.

[Tan and Le, 2019] Tan, M. and Le, Q. V. (2019). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pages 6105–6114. PMLR.

[Tan and Le, 2021] Tan, M. and Le, Q. V. (2021). EfficientNetV2: Smaller Models and Faster Training. *arXiv:2104.00298 [cs]*.

[Tewari et al., 2019] Tewari, A., Bernard, F., Garrido, P., Bharaj, G., Elgharib, M., Seidel, H.-P., Perez, P., Zollhofer, M., and Theobalt, C. (2019). FML: Face Model Learning from Videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10812–10822.

[Tewari and Kim, 2015] Tewari, A. and Kim, H. (2015). High-Fidelity Monocular Face Reconstruction based on an Unsupervised Model-based Face Autoencoder. 14(8):14.

[Tewari et al., 2018] Tewari, A., Zollhofer, M., Garrido, P., Bernard, F., Kim, H., Perez, P., and Theobalt, C. (2018). Self-supervised Multi-level Face Model Learning for Monocular Reconstruction at over 250 Hz. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, page 16.

[Tewari et al., 2017] Tewari, A., Zollhöfer, M., Kim, H., Garrido, P., Bernard, F., Pérez, P., and Theobalt, C. (2017). MoFA: Model-based Deep Convolutional Face Autoencoder for Unsupervised Monocular Reconstruction. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops*.

[Tian et al., 2021] Tian, Y., Chen, X., and Ganguli, S. (2021). Understanding self-supervised Learning Dynamics without Contrastive Pairs. *arXiv:2102.06810 [cs]*.

[Tran et al., 2016] Tran, A. T., Hassner, T., Masi, I., and Medioni, G. (2016). Regressing robust and discriminative 3D morphable models with a very deep neural network. *arXiv preprint arXiv:1612.04904*.

[Tran and Liu, 2018] Tran, L. and Liu, X. (2018). Nonlinear 3D Face Morphable Model. *arXiv:1804.03786 [cs]*.

[Turk and Levoy, 1994] Turk, G. and Levoy, M. (1994). Zippered Polygon Meshes from Range Images. In *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '94, pages 311–318.

[Viola and Jones, 2001] Viola, P. and Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, pages I–511–I–518, Kauai, HI, USA. IEEE Comput. Soc.

[Viola and Jones, 2004] Viola, P. and Jones, M. J. (2004). Robust Real-Time Face Detection. International Journal of Computer Vision(57):137–154.

[Wang et al., 2004] Wang, Z., Bovik, A., Sheikh, H., and Simoncelli, E. (2004). Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Transactions on Image Processing*, 13(4):600–612.

[Xiong and De la Torre, 2013] Xiong, X. and De la Torre, F. (2013). Supervised descent method and its applications to face alignment. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference On*, pages 532–539. IEEE.

[Xu et al., 2021] Xu, X., Chen, H., Moreno-Noguer, F., Jeni, L. A., and De la Torre, F. (2021). 3D Human Pose, Shape and Texture from Low-Resolution Images and Videos. *arXiv:2103.06498 [cs]*.

[Yu et al., 2018] Yu, C., Wang, J., Peng, C., Gao, C., Yu, G., and Sang, N. (2018). BiSeNet: Bilateral Segmentation Network for Real-time Semantic Segmentation. In *Computer Vision - ECCV 2018*, pages 334–349. Springer International Publishing.

[Zafeiriou, 2015] Zafeiriou, S. (2015). A survey on face detection in the wild: Past, present and future. *Computer Vision and Image Understanding*, page 24.

[Zafeiriou et al., 2017] Zafeiriou, S., Trigeorgis, G., Chrysos, G., Deng, J., and Shen, J. (2017). The Menpo Facial Landmark Localisation Challenge: A Step Towards the Solution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.

[Zhang et al., 2017] Zhang, S., Zhu, X., Lei, Z., Shi, H., Wang, X., and Li, S. Z. (2017). S^3FD: Single Shot Scale-Invariant Face Detector. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 192–201, Venice. IEEE.

[Zhao et al., 2017] Zhao, H., Shi, J., Qi, X., Wang, X., and Jia, J. (2017). Pyramid Scene Parsing Network. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6230–6239, Honolulu, HI. IEEE.

[Zhu et al., 2016] Zhu, X., Lei, Z., Liu, X., Shi, H., and Li, S. Z. (2016). Face Alignment Across Large Poses: A 3D Solution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 146–155.

Christophe **Ecabert**

PhD Student · Computer Vision & Machine learning

Avenue d'Echallens 41, 1004, Lausanne, Switzerland

(+41) 79 603 51 19 | christophe.ecabert@gmail.com | cecabert

Education

Ecole Polytechnique Fédérale de Lausanne (EPFL) — *Lausanne, Switzerland*

Ph.D. candidate in Electrical Engineering — *2017 - Present*

- Thesis Topic: *Model-based 3D Face Reconstruction*
- Adviser: Prof. Jean-Philippe Thiran

Ecole Polytechnique Fédérale de Lausanne (EPFL) — *Lausanne, Switzerland*

M.S. in Electrical Engineering — *2011 - 2014*

- Thesis Topic: *Model-based Virtual View Synthesis of Faces from Multiple Cameras*
- Area of Study: Major in **information technologies**

HES-SO He-Arc Ingénerie — *St-Imier, Switzerland*

B.S. in Electrical Engineering — *2007 - 2010*

- Area of Study: Embedded Systems

Lycée technique Baptiste-Savoie — *St-Imier, Switzerland*

CFC in Electrical Engineering & professional maturity — *2004 - 2007*

- Area of Study: Electronics

Professional Experience

Ecole Polytechnique Fédérale de Lausanne (EPFL) — *Lausanne, Switzerland*

Doctoral Assistant, Signal Processing Lab (LTS5) — *2017 - PRESENT*

- Doctoral assistant in the field of monocular 3D face reconstruction
- Designed & developped a complete reconstruction pipeline that synthesis 3D face out of a single RGB image using computer vision & machine learning deep learning techniques
- Supervision of students for various projects related to 3D face reconstruction and facial image analysis

Ecole Polytechnique Fédérale de Lausanne (EPFL) — *Lausanne, Switzerland*

Scientific Assistant, Signal Processing Lab (LTS5) — *2014 - 2017*

- Research and development of a pipeline for the generation of synthetic frontal view of the human face using statistical 3D face model and multi-camera setup
- Face alignment, facial expressions and action unit detection algorithm implementation and integration
- Development and maintenance of a colaborative and multi-platform C++ library for facial image processing and analysis
- Conception of an Android application for medical domain (User interface, C++ framework integration)

nViso SA — *Lausanne, Switzerland*

Software Engineer Intern — *2013 - 2014*

- Software engineer intern for the development of an Android mobile application for pain detection based on facial expression analysis

Skills

Programming	Python, C++, CUDA
Libraries	STL, OpenMP / TBB, OpenCV, OpenGL, Numpy, Scipy, Scikit-learn, Tensorflow
Tools	CMake, Git, CI/CD, Docker, Bash scripting
Languages	French (native) English (TOEIC - 92pts, tested in 2011), German (Basic)

Projects & Publications

Conference Publication

- D. Engin, **C. Ecabert**, H. K. Ekenel, J.-P. Thiran. Face Frontalization for Cross-Pose Facial Expression Recognition. *26th European Signal Processing Conference (EUSIPCO)*, 2018
doi:10.23919/EUSIPCO.2018.8553087

Projects *Lausanne*

- Model-based Virtual View Synthesis of Faces from Multiple Cameras.
  - Master thesis addressing the synthesis of a 3D frontal virtual view of the face using statistical model in a multi-camera environment
  - Professor: J.-P. Thiran Supervisor : Dr H. Gao
- Kinect : Motor and functional assessment of upper limbs using Microsoft Kinect
  - Kinect data logger implementation, validation of the measurements during Fugl- Meyer test on healthy subjects
  - Professor: K. Aminian Supervisor : F. Massé

124