ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

CVL Computer Vision Lab

3D Face Estimation from a Monocular RGB Image with Dense Landmarks

—

Master Thesis

Nicholas Simic

Advisors: Dr. Gurkirt Singh, Vasileios Choutas
Supervisor: Prof. Dr. Luc van Gool

November 21, 2022

Abstract

3D face reconstruction methods rely in some way on sparse 2D landmarks, which are not sufficient for capturing fine facial expression details, and on analysis-by-synthesis via differentiable rendering. Annotating more 2D keypoints for in-the-wild images is time-consuming and difficult for humans. Synthetic data, which provide perfect annotations, can solve the problems that plague manual annotation. However, synthesizing data that are realistic enough to overcome the domain gap from in-the-wild data is not trivial. State-of-the-art landmark prediction methods use fully convolutional networks to predict heatmaps, i. e. a discrete 2D grid of landmark location likelihood. The main disadvantages of this approach are the large memory and computation requirements, limiting the number of landmarks, thus making 3D face fitting less accurate, and heatmap resolution, reducing the precision of the detected landmarks. Analysis-by-synthesis approaches still have to use sparse landmarks to guide training and have to resort to approximations for illumination, facial appearance and rendering to make the process differentiable and the optimization tractable. In this work, we address the problem of 3D face reconstruction from single images by fitting a 3D morphable model to a dense set of predicted 2D landmarks. Instead of synthetic data, we use a pseudo ground-truth dataset, DAD-3DHeads, to predict 20x more landmarks than the standard sparse set and per landmark confidences, using a direct regression approach to avoid the issues of heatmap prediction. We then fit a 3D face model to the detected 2D landmarks. We find that it is possible to obtain competitive results for 3D face estimation using only dense landmarks trained on noisy real data, avoiding the complexity of generating synthetic datasets or analysis-by-synthesis approaches.

Contents

1. Introduction

The 3D reconstruction of the human body is a very popular problem in computer vision with many important applications in virtual and augmented reality, biomechanics and medicine. Although there exist reliable and robust solutions for estimating the 3D body from Motion Capture (Mocap) [33, 75] markers, multi-view images [45, 120] or wearable sensors [38, 110], these are limited to a few subjects and scenarios. Estimation from monocular images is a more convenient alternative, due to the widespread use of smartphones and cameras, but significantly more challenging because of occlusions, perspective ambiguities, and appearance variation. In an attempt to simplify the problem and limit the space of solutions, smaller body parts, such as the hands and face, are usually treated separately from the rest of the body. In this work, we choose to focus on faces. Face recognition [81] tries to identify a person by looking at an image of their face. Because of differences in illumination, pose and expression, face recognition from images is challenging. 3D geometry on the other hand is invariant to pose and illumination changes making recognition more robust. 3D face estimation also simplifies face animation in the game and movie industries and virtual reality. For example, facial motion re-targeting [14, 15] aims to transfer details of one person, e. g. expression and facial wrinkles, captured from a single image or a video to another person's face. While there exist non-parametric solutions to this problem [9], most approaches [34, 56, 104, 112] employ a parametric 3D model of the face [5, 26, 32, 65, 83]. These models decompose face deformations into identity or shape, expression and articulation, with separate parameters for each deformation category. This makes it easier to transfer deformations from one subject to another since one only has to copy the relevant parameters. This is also useful for medicinal applications, e. g. showing the results of face treatments only requires the modification of the 3D face model parameters obtained from an image.

Face model parameters can be obtained from an image either through direct estimation or by fitting to the output of a face alignment model. Face alignment represents human faces in images using key 2D points, called landmarks, whose position in the image varies depending on different head poses, expressions and identities. Even though it seems natural to use as many 2D points as possible to capture faces from images, state-of-the-art face analysis methods only employ a sparse set of points, usually equal to 68. The main reason for this is the difficulty of annotating dense points on images [36, 113]. It is very difficult for human annotators to accurately place points for challenging face poses, occluded parts or poorly illuminated areas. Because of this, the majority of existing datasets are labeled with only 2D/3D sparse set of points [3, 55, 58, 95, 98, 125] and are not able to handle challenging in-the-wild conditions, such as occlusions or large pose variations. Moreover, most methods for face keypoint detection are fully convolutional networks that regress heatmaps [7, 8, 67, 80, 107, 118]. Heatmaps are 2D grids representing the likelihood of a predicted landmark in each grid location. Using a large number of landmarks, e. g. ≥ 500, is not feasible with this type of approach, because of the computational and memory cost of computing the heatmaps. The lack of labeled datasets is even more severe for 3D reconstruction, given that usually we do not have 2D-3D paired

information. Datasets acquired with Kinect or multi-view cameras [10, 16, 19, 117] are accurate, but at the same time are either small or not very diverse, having only a few subjects. An attractive alternative is the creation of synthetic datasets[113, 114, 125], generated from 3D morphable models parameters. Wood et al. [113, 114] generate a synthetic dataset and train a simple regression network that predicts up to 703 landmarks and then fit a 3D model to the detected landmarks, obtaining very competitive results. However, generating data that are realistic enough to overcome the domain gap w. r. t . real data is very difficult, requiring complex rendering pipelines. Diversity is another important issue that needs to be addressed, as the available 3D assets might not cover the full range of human appearance variation. Another alternative is in-the-wild image annotation. The DAD-3DHeads [77] dataset is an example of this approach. It contains image to 3D correspondences for the whole head for $\sim$ 44k examples. To annotate the data, a user first selects correspondences between the image and a 3D face mesh, and then, an optimization-based approach estimates face model parameters that minimize the deviation between the projected 3D mesh locations and 2D image positions. Finally, analysis-by-synthesis is another approach, both for face parameter estimation and for generating pseudo ground-truth 3D data, which avoids the need of having a human annotator manually select correspondences. These methods predict illumination and appearance parameters and employ a differentiable renderer [20, 22, 27, 97, 105, 106] to render an estimated face image and compare it with the input image. The drawbacks of this type of approach are the assumptions needed to make rendering differentiable and computationally feasible, e. g. the use of spherical harmonics [87] for illumination. Note that almost all face reconstruction methods that use a photometric term still use 2D sparse landmarks in the training process [20, 27, 103, 106].

In this work, we propose a solution to the 3D face estimation problem that builds on two components, face alignment, and 3D face model fitting. We use the projected ground truth 3D vertices of DAD-3DHeads [77] as noisy 2D ground truth labels to train a dense 2D landmark detector, similar to the pipeline of Wood et al. [114]. We explore different types of losses and architectures to make our predictor more efficient and more accurate than the baseline. Specifically, we use a more powerful regression loss [64] that was shown to be able to outperform heatmap regression for human pose detection. In addition, we train our model with different backbones [35, 40, 72, 73] to build the optimal predictor. We also predict confidence values per landmark that may help the fitting of the model when there are occlusions or may serve as correcting factors for erroneous labels. To estimate model parameters, we use a new application agnostic differentiable non-linear optimization library, called Theseus [84] built on top of PyTorch [82]. Note that our pipeline is part-agnostic and could just as well be applied to other body parts, e. g. the hands. We evaluate our method on part of the validation set provided by DAD-3DHeads that we use as a test set, with DAD-3DNet [77] and DECA [27] as baselines. We find that our reconstruction is more accurate and expressive than the baselines. This demonstrates that just using many landmarks is a viable solution for 3D face reconstruction, without the need for a specialized rendering pipeline to generate a synthetic dataset.

2. Related Work

Landmarks Localization: In order to collect information from faces e. g. identity, emotions, or intention, keypoint detection is an essential step, with many applications, such as facial expression recognition [81], eye tracking [39] or 3D reconstruction [77, 113, 114], relying on them. We focus our attention on regression methods since they are closer to our approach. A more detailed analysis can be found in Wu et al. [115]. Initial approaches follow the cascade regression paradigm [12, 50, 90, 116]. This is an iterative procedure that starts from an initial guess for the landmarks and at each step learns mappings from extracted local features to landmarks position updates. The methods differ in features and regressor models used. Deep learning-based architectures [40] have become the de facto tool for keypoint estimation. There are two options for keypoint prediction, direct regression using fully-connected layers [21, 57, 124, 125], and fully convolutional networks that regress heatmaps [7, 8, 67, 80, 107, 118], i. e. a 2D array that represents the likelihood of a landmark being present at each location. Trigeorgis et al. [108] use recurrent networks to iteratively refine landmark estimates. Liu et al. [69] also use recurrent networks to process a temporal sequence of images. Other than architecture difference we can distinguish between methods that fit 3D statistical face models [5, 26, 32, 65, 83] to the images using learned features [4, 48, 49, 71, 125], region-based methods that use multiple architectures that independently analyze different part of the face [13, 24, 74, 101], and multi-task methods that jointly infer other attributes such as age, gender in addition to the landmarks [89, 122, 123]. Example datasets for this task are AFLW [55], which contains 21 landmarks annotations, LFPW [3] with 29 points and 68 with re-annotation [95], the Helen database [58] with 194 points, 300-W [95], 300-VW [98], 300-W-LP [125] all 68 landmarks.

3D Face Estimation: Estimating 3D faces from images is an extensive research area in computer vision [78]. Since the problem of reconstructing 3D from 2D is ill-posed, prior knowledge is needed to restrict the space of solutions. We can differentiate the methods based on how they add prior knowledge. Statistical model fitting methods are optimization methods that estimate parameters of statistical face models [5, 26, 32, 65, 83] by optimizing a non-linear function, in order to fit the model to the image. The 3D model gives the prior knowledge since it lies in a space learned from 3D examples. Within this category there are methods [5, 70, 85, 99, 126] that optimize parameters in order to minimize the difference between a rendered image and the original image. In general, they make use of a sparse set of landmarks to guide the fitting. Photometric methods [51, 61, 94] on the other hand use other representations for the face estimating for example surface normals assuming an illumination model that decomposes the image into albedo, lighting, and normals. If only one image is used 3D morphable models are still emplyed to add prior knowledge [11, 66, 94]. Deep learning methods learn the mapping from 2D to 3D. The main obstacle is the lack of ground truth 2D-3D labels. 3D face datasets can be differentiated by the method used to acquire data. Acquisition through Kinect e. g. FaceWarehouse [10], 4DFAB [16] or multi-view camera systems e. g. D3DFACS [19], FaceScape [117] these are very accurate datasets but are small and lack diversity since they only cover very few subjects. Another possibility is synthetic datasets. There are two main approaches, either by

fitting 3D morphable models to images and then rendering other ones e. g. 300-W-LP [125], or randomly sampling 3D morphable models and rendering the images e. g. Face Synthetics [113]. Another approach is just fitting a 3D morphable model to images e. g. DAD-3DHeads [77]. Deep Learning methods use these datasets and experiment on different network frameworks as well as training space, either 2D, 3D or model parameters. We mention methods that use a single network CNNs [15, 17, 37, 111], others that use multiple iterations such as cascade regressions [92, 96, 121, 125], encoder-decoder architectures [14, 31, 100, 119], generative adversarial networks [28, 29, 68] or geometric deep learning [60, 68]. To overcome the 2D-3D pair data problem there are methods based on self-supervision that add a differentiable rendering layer at the end of the network to produce an image and avoid the need of ground truth labels [20, 22, 27, 97, 105, 106]. For leading methods in 3D reconstruction with deep learning, we mention RingNet [96] that is trained on multiple images of the same person and predicts FLAME [65] parameters, supervision is done with image sparse landmarks. In addition, they created the NoW benchmark [96] for evaluating the quality of 3D shape reconstruction. DECA [27] also finds parameters of FLAME and uses a differentiable renderer to render an image and compare it to the input image. Sparse landmarks are also used. In addition to a coarse reconstruction, they also find a detailed and animatable expressive reconstruction. EMOCA [20] that builds on top of DECA and introduces a novel loss to obtain more expressive reconstructions. Finally, DAD-3DNet [77] that jointly predicts both landmarks and FLAME parameters. The closest approach to our own, is the recent work of Wood et al. [114]. They use a standard convolutional neural network [40] with a negative gaussian log-likelihood loss to predict 703 landmarks and variances. They optimize 3D statistical model parameters to fit to the landmarks by employing the same loss, now on the re-projected vertices and the estimated landmarks. The fundamental difference between our approaches is that they create and employ a synthetically generated dataset in order to train the predictor. This greatly increases the overall complexity and decreases the flexibility of their method compared to our own.

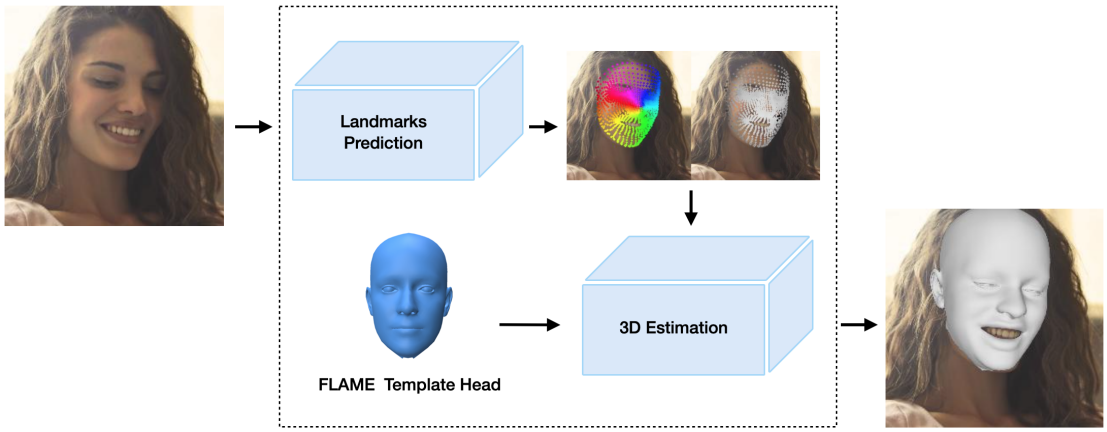

Figure 3.1: Visual representation of our proposed method. Our system receives an input image (left) and estimates a 3D head mesh (right). This is done in two steps: first, we predict dense 2D face landmarks from the input images, and then, we fit a 3D face model, FLAME, to the detected landmarks to produce the estimated head.

3. Method

Our approach consists of two main stages: a neural network for landmarks prediction, which predicts landmark positions on the image and confidences values for each point, and optimization of 3D parameters of a morphable model to fit the predicted landmarks. Figure 3.1 shows a visual overview of our pipeline, with the predicted landmarks positions and the confidences, where brighter means more confident. In Sec. 3.2 we present the dataset used. Section 3.3 describes the architecture and loss employed to train our landmark predictor. Section 3.1 contains the description of the FLAME model. Finally, Sec. 3.4 describes the pipeline that estimates a 3D face from the detected 2D landmarks.

3.1 Human Face Model

A 3D face model [5, 26, 32, 65, 83] is a mathematical representation of 3D face. The basic idea behind a 3D morphable model (3DMM) is that given a big enough set of 3D examples of faces we can learn suitable spaces to interpolate between representations of faces and therefore produce plausible new ones. The strength of this concept relies on the compact representations of faces that they provide rather than having to deal directly with the meshes.

FLAME [65] is a 3D face model parameterized by shape, pose, and expression and was implemented in order to address a gap in 3D face modeling. Available low level head models could be fitted to general data but lacked accuracy whereas high end methods were accurate and photo-realistic but required 3D scans and manual labor. We can factor the FLAME model in

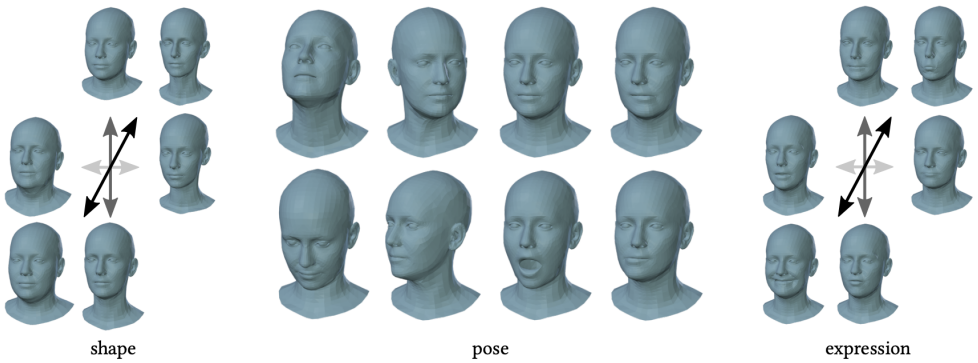

Figure 3.2: Parameterization of the model, female model shown. On the left example of shape variations, middle pose variations, and on the right expression variations. Source FLAME [65].

two distinct parts: a skeleton, composed of a pre-defined number of joints, and skin, which is a surface mesh "attached" to the skeleton. The FLAME model mesh has $\mathtt{V} = 5023$ vertices, $\mathtt{F} = 9976$ faces and $\mathtt{J} = 4$ joints that correspond to the jaw, neck, and eyeballs, see Fig. 3.3 for more details. The connectivity of the model mesh does not depend on the deformations, whereas the location of the model's vertices and joints is determined by the process described below. Let vectors $\boldsymbol{\beta} \in \mathbb{R}^{|\boldsymbol{\beta}|}$, $\boldsymbol{\theta} \in \mathbb{R}^{|\boldsymbol{\theta}|}$, $\boldsymbol{\psi} \in \mathbb{R}^{|\boldsymbol{\psi}|}$ be the shape, expression and pose parameters. The pose vector $\boldsymbol{\theta} \in \mathbb{R}^{3(\mathtt{J}+1)+3} = [\boldsymbol{\theta}_r, \boldsymbol{\theta}_s]$ contains $\mathtt{J}+1$ rotation vectors in axis-angle format and the root translation, split into $\boldsymbol{\theta}_s \in \mathbb{R}^{3\mathtt{J}}$ that contains the per-joint rotations and $\boldsymbol{\theta}_r \in \mathbb{R}^{3+3}$ the root rotation and translation. Let $\mathcal{T}$ be a template mesh in the neutral pose. We can think of any target face to be a deformation of $\mathcal{T}$. Let $\mathcal{T}_P(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\psi})$ be the template mesh with added shape, pose, and expression vertex offsets. These offsets are the results of functions that depend on $\boldsymbol{\beta}$, $\boldsymbol{\theta}$, $\boldsymbol{\psi}$ called blendshapes.

$$\mathcal{T}_P(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\psi}) = \mathcal{T} + B_S(\boldsymbol{\beta}; \mathcal{S}) + B_P(\boldsymbol{\theta}; \mathcal{P}) + B_E(\boldsymbol{\psi}; \mathcal{E}) \tag{3.1}$$

$B_S(\boldsymbol{\beta}; \mathcal{S}) : \mathbb{R}^{|\boldsymbol{\beta}|} \rightarrow \mathbb{R}^{3\mathtt{V}}$, a shape blendshape function that explains variation in shape from the base template $\mathcal{T}$:

$$B_S(\boldsymbol{\beta}; \mathcal{S}) = \sum_{i=1}^{|\boldsymbol{\beta}|} \beta_i \boldsymbol{S_i}. \tag{3.2}$$

$\mathcal{S} \in \mathbb{R}^{3\mathtt{V} \times |\boldsymbol{\beta}|}$ is the orthonormal shape basis learned with PCA. Each element of the basis $\boldsymbol{S}_i \in \mathbb{R}^{3\mathtt{V}}$ is a vertex offset from the base template $\mathcal{T}$. $B_E(\boldsymbol{\psi}; \mathcal{E}) : \mathbb{R}^{|\boldsymbol{\psi}|} \rightarrow \mathbb{R}^{3\mathtt{V}}$, an expression blendshape function that explains variation in expression, e. g. frowning, from the base template $\mathcal{T}$:

$$B_E(\boldsymbol{\psi}; \mathcal{E}) = \sum_{i=1}^{|\boldsymbol{\psi}|} \psi_i \boldsymbol{E_i}, \tag{3.3}$$

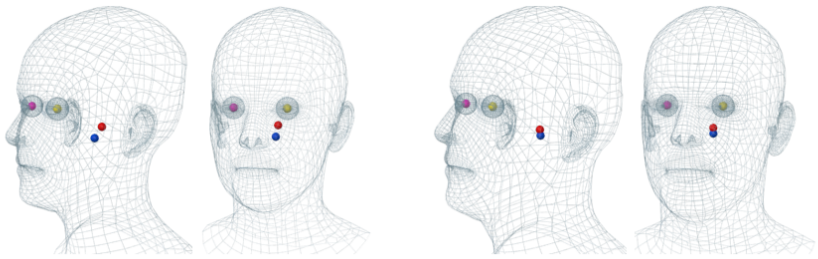

Figure 3.3: Joints location and skin. Pink/Yellow are eyes joints, red is the neck joint and blue jaw joint. Female model (left) and male model (right). Source FLAME [65].

where $\mathcal{E} \in \mathbb{R}^{3\mathtt{V}\times|\boldsymbol{\psi}|}$ is the orthonormal expression basis, where each element of the basis $\boldsymbol{E}_i \in \mathbb{R}^{3\mathtt{V}}$ is a vertex offset from the base template $\mathcal{T}$.

$B_P(\boldsymbol{\theta};\mathcal{P}) : \mathbb{R}^{|\boldsymbol{\theta}|} \rightarrow \mathbb{R}^{3\mathtt{V}}$ contains corrective pose blendshapes, that fix pose artifacts from LBS:

$$B_P(\boldsymbol{\theta};\mathcal{P}) = \sum_{i=1}^{9\mathtt{J}} (R_i(\boldsymbol{\theta}) - R_i(\boldsymbol{\theta}^\star))\boldsymbol{P_i}, \tag{3.4}$$

where $R_i(\boldsymbol{\theta}) : \mathbb{R}^{|\boldsymbol{\theta}|} \rightarrow \mathbb{R}^{9\mathtt{J}}$ is a function that takes a pose vector $\boldsymbol{\theta}$ and returns the concatenated rotation matrices. $\mathcal{P} \in \mathbb{R}^{3\mathtt{V}\times 9\mathtt{J}}$ is the pose space coefficient and $\boldsymbol{P}_i \in \mathbb{R}^{3\mathtt{V}}$ is a vertex offset from the base template $\mathcal{T}$.

Skinning is the process of controlling the deformation of the model mesh using skeleton transformations. Linear blend skinning (LBS) [63] is the most used algorithm for direct skeletal shape deformation. It is based on the concept that the vertices in the mesh (skin) move accordingly from a change in joints (bones) configuration. For example, in character animation a 3D artist would move the joints of a model and LBS is the function that moves the vertices of the mesh accordingly. It has the following formulation:

$$\mathcal{LBS} := v'_j = \left(\sum_{i}^{\mathtt{J}} \mathcal{W}_{i,j} * \mathtt{T}_i\right) * v_j, \tag{3.5}$$

where v_j is vertex j of the mesh, $\mathtt{T}_i$ are transformations for each joint from the rest configuration to the desired one, $\mathcal{W}_{i,j} \in [0,1]$ are weights that describe the amount of influence of joint i on vertex j and $\mathtt{J}$ is the number of joints.

FLAME is essentially a function that maps shape $\boldsymbol{\beta}$, expression $\boldsymbol{\psi}$ and pose $\boldsymbol{\theta}$ parameters to a 3D mesh using LBS and additive blendshape functions, defined by the following function:

$$\boldsymbol{V}(\boldsymbol{\beta},\boldsymbol{\theta},\boldsymbol{\psi}) : \mathbb{R}^{|\boldsymbol{\beta}|\times|\boldsymbol{\theta}|\times|\boldsymbol{\psi}|} \rightarrow \mathbb{R}^{3\mathtt{V}} \tag{3.6}$$

$$\boldsymbol{V}(\boldsymbol{\beta},\boldsymbol{\theta},\boldsymbol{\psi}) = \mathcal{LBS}\left(\mathcal{T}_P(\boldsymbol{\beta},\boldsymbol{\theta},\boldsymbol{\psi}), J(\boldsymbol{\beta}), \boldsymbol{\theta}, \mathcal{W}\right) \tag{3.7}$$

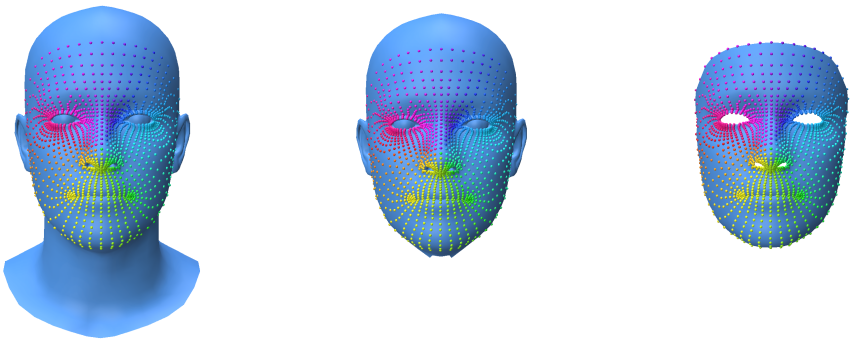

Figure 3.4: FLAME [65] full model (left), where the colored points are the $\mathtt{L} = 1609$ points selected as dense landmarks, head model with no neck (middle), and the face-only model (right).

where $\mathtt{V}$ is the number of vertices. Figure 3.2 shows the effect of varying the parameters. $J(\boldsymbol{\beta}; \mathcal{J}, \mathcal{T}, \mathcal{S}) = \mathcal{J}\left(\mathcal{T} + B_S(\boldsymbol{\beta}; \mathcal{S})\right)$ are the joint locations defined as a function of shape, since different face shapes have joints in different locations. $\mathcal{J}$ is a learnable sparse matrix that defines the mapping from mesh vertices to joint locations. $\mathcal{LBS}$ is the LBS function defined in Eq. (3.5). In order to learn $[\mathcal{S}, \mathcal{P}, \mathcal{E}, \mathcal{J}, \mathcal{W}]$, FLAME was trained on three heterogeneous datasets of 3D scans. The shape space is learned from the CAESAR [93] dataset. For pose and expression parameters, a combination from D3DFACS [19] and data made by FLAME was used.

In this work, we predict a set of $\mathtt{L} = 1609$ landmarks corresponding to the $\mathtt{V}_\mathtt{L}$ vertices of the face. For 3D evaluation of our estimated reconstructions, we use either the set of $\mathtt{V}_\mathtt{L}$ face vertices or the set of $\mathtt{V}_\mathtt{H} = 4757$ head vertices corresponding to the whole head without the neck. Throughout the work, we are going to address the landmarks subset as $\boldsymbol{L}$, the corresponding face vertex subset as $\boldsymbol{V}_L$, the FLAME vertex subset as $\boldsymbol{V}$ and the head without neck vertex subset as $\boldsymbol{V}_H$. See Figure 3.4 for a visual representation.

3.2 Dataset

Collecting a dataset for face analysis or 3D estimation is not trivial. The main problems are the annotation of the landmarks for 2D and finding paired data for 3D. Most available datasets have sparse annotations for only the visible parts or are not accurate or descriptive enough for in-the-wild analysis. Overcoming this issues requires the use of synthetic data, application-specific methods or assumptions. DAD-3DHeads [77], addresses the dense annotation problem and in-the-wild generalization with a human-in-the-loop annotation process. Data acquisition is done by fitting the FLAME model [65] to the images through an interactive interface where the annotators can pair vertices of the model with pixels in the image. After selecting a new correspondence, an optimization process refines models parameters to minimize the re-projection error of the paired points. During this process annotators can visualize the resulting mesh with the texture from the image to see if the intermediate results are plausible. Using this pipeline,

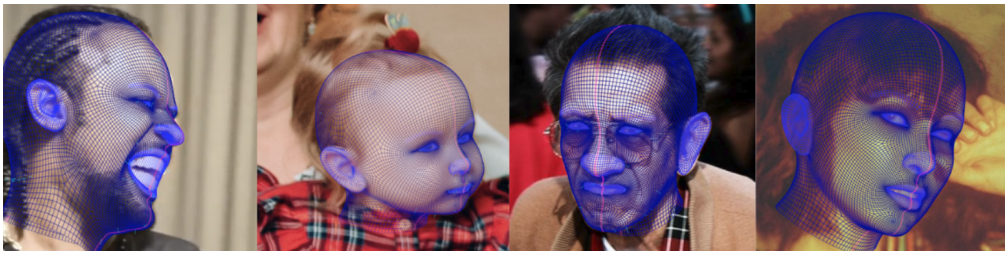

Figure 3.5: Dataset presentation with vertices fitted to the faces. Source DAD-3DHeads [77].

Martyniuk et al. [77] obtain pseudo ground truths of the full $\mathtt{V} = 5023$ vertices of the FLAME model. We refer to this as pseudo ground-truths, since the selection of correspondences from annotators is ambiguous and the accuracy of the annotation process is not reported. Nonetheless this procedure can be applied to any image irrespective of illuminations, drastic poses or occlusions for the faces. DAD-3DHeads contains 44898 images of in the wild conditions, with 37840 images in the training set, 4312 in validation and 2746 in the test set.

Despite having the whole model vertices available, we choose to predict only $\mathtt{L} = 1609$ landmarks corresponding to the face vertices. This is done to minimize the possible inaccuracies on the back of the head, since there are no salient points that annotators can reliably select. Moreover, we predict confidences per landmarks to correct possible imprecisions, both in labels and in our predictions.

3.3 Landmark Prediction

3.3.1 Backbone

The first component of our network takes the input image $\mathrm{I} \in \mathbb{N}^{H \times W \times 3}$ and extracts relevant features $\mathcal{F} \in \mathbb{R}^{C}$. It learns the following mapping:

$$\mathcal{F}(\mathrm{I}) : \mathbb{N}^{H \times W \times 3} \rightarrow \mathbb{R}^{C} \tag{3.8}$$

Convolutional neural networks (CNNs) [59] are very powerful models, used to extract features from images for classification, detection and semantic segmentation. The difference from general artificial neural networks is that they use convolution instead of general matrix multiplication in their layers. This allows to learn translation invariant representations, which is a property especially important for image analysis. In a CNN the first layer receives the input image and convolves it with certain number of filters resulting in a volume of feature images. After this a non-linear function is applied. A pooling operation can be used over the features to further reduce their shape and at the same time compress the information from close pixels. The result passes to the next layer that does the same operations now on the features. The parameters of the filters are then learned with gradient descent from the loss. Since deep neural networks are more difficult to train but more powerful, variants of CNNs were implemented.

One of these variants is a CNN called ResNet [40] that introduces residual learning. It was introduced to address the problem of degradation of accuracy in training of deep networks due

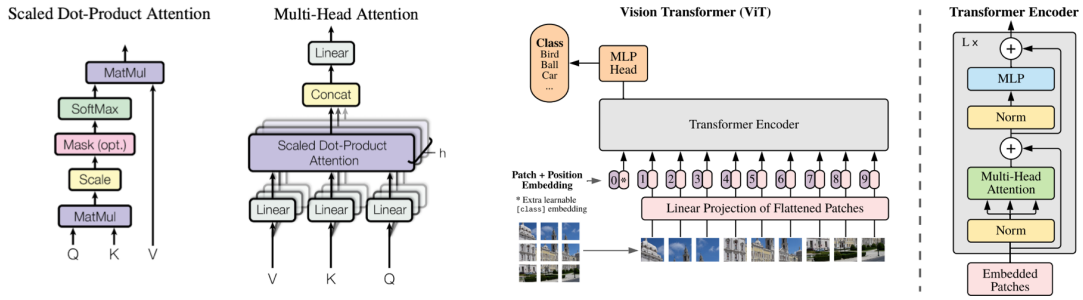

Figure 3.6: Scaled dot product attention layer, and the multi-head attention layer (left). Source: Vaswani et al. [109]. Vit Vision Transformer (right). Source Dosovitskiy et al. [25].

to vanishing gradients. If the underlying map that a block of layers should learn is $H(x)$, instead of approximating it directly by $F(x) = H(x)$ ResNet for certain layers approximates a residual map $F(x) = H(x) - x$. The motivation is that in the extreme of an identity mapping $H(x) = x$, it would be easier for a network to push the residual to zero than to approximate an identity mapping through a stack of non-linear layers. A ResNet block can be seen in Fig. 3.7.

Another type of architecture that recently came into the spotlight is Vision Transformers (ViT) [25]. Transformers [109] are a popular class of models for Natural Language Processing (NLP) problems, designed to process sequential input data, such as words in a sentence, called tokens. Different from their predecessors, Transformers are encoder-decoder architectures that process the sequential input as a whole, allowing parallel computations. The most important concept behind Transformers is a method called self-attention that computes new feature tokens by weighting the original ones. In particular, for every token three vectors are computed, the Key, the Value and the Query by multiplying the token with learned matrices. For each Query the dot product with every Key is computed. This gives for every token a set of weights corresponding to the relevance that the other tokens have on it. The new token at position i will be the sum of the Values of all tokens weighted by the weights found with Query i. This corresponds to a scaled dot product attention layer. Stacking multiple layers in parallel creates a multi-head attention layer as shown in Fig. 3.6. Vision Transformers operate in the same way as Transformers but on images, with the word tokens in NLP being replaced by image patches, as in Fig. 3.6 for ViT. Even though ViT had great success for classification, it is the Swin Transformer [72] that made vision transformers competitive for general computer vision problems such as object detection or semantic segmentation. The Swin Transformer adopts classical convolutional network concepts and builds a hierarchical feature map by merging image patches in deeper layers. It increases efficiency by computing self attentions only within a local window and not globally.

The success of the shifted window approach for local self-attention computation of the Swin Transformer brought researchers to reconsider the power of the convolutional structure and attempt ways to make self attentions closer to convolutions. Liu et al. [73] did the opposite and introduced a new version of a CNN, called ConvNeXt, that adopts insights from Transformers training and structure to fully convolutional architectures. In particular, it replaces the ReLU activation function [79] with the smoother GELU [41]. ConvNeXt uses fewer activa-

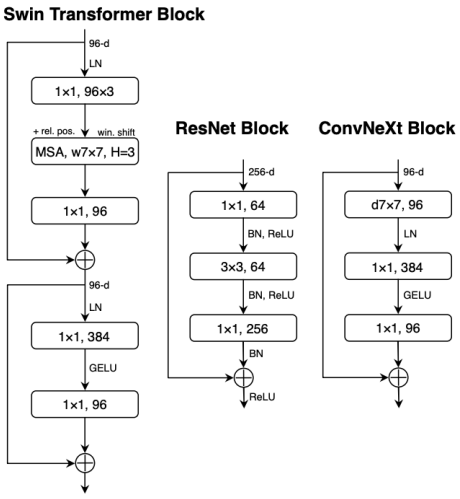

Figure 3.7: Comparison of Swin Transformer, ResNet, and ConvNeXt block structure. For simplicity linear layers in the feed forward blocks of the Transformers are labeled as 1x1 convolutions. Source Liu et al. [73].

tions, fewer normalization layers and replaces BatchNorm [46, 47] with Layer Normalization [2], in accordance with common practice in Transformer literature. A schematic view of the block architecture difference between ResNet, Swin Transformer, and, ConvNeXt can be seen in Fig. 3.7.

Finally, Mobilenet V3 [42] is a network architecture that is meant to be efficient enough to be used in mobile applications. It uses combinations of layer concepts from previous mobile network architectures, such as depth-wise separable convolutions, introduced in MobileNet V1 [44], inverted residual structures from MobileNet V2 [43], lightweight attention modules from MnasNet [102] and modified swish non-linearities [86] with hard sigmoid functions [1].

3.3.2 Predictor

The second component of our network takes the extracted features $\mathcal{F} \in \mathbb{R}^C$ and estimates a vector $\boldsymbol{q} \in \mathbb{R}^{4\mathrm{L}}$. The first $2\mathrm{L}$ entries correspond to the predicted landmarks $\hat{\boldsymbol{L}}$, the remaining entries are later mapped to $[0,1]^{2\mathrm{L}}$ to produce the predicted confidences $\hat{\boldsymbol{C}}$. It learns the following mapping:

$$\boldsymbol{q}(\mathcal{F}) : \mathbb{R}^C \rightarrow \mathbb{R}^{4\mathrm{L}} \tag{3.9}$$

The most simple predictor is a linear layer that learns a linear map from the features. Another possibility is a multi layer perceptron (MLP) that introduces non-linearities. In this work we explore on new geometric deep learning architectures. Graph Neural Networks (GNNs) [6] are generalizations of neural network architectures for processing data that can be represented by graphs. For example, CNNs can be seen as specialized graph networks for grids, and Transformers for NLP can be seen as graph networks on graphs where the nodes are the words in the

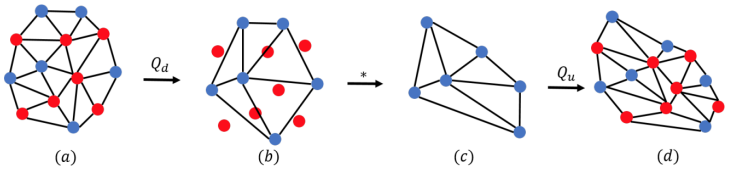

Figure 3.8: In red the contracted vertices, blu the vertices that remains. Q_d,Q_u are the Downsampling and Upsampling matrices respectively.$*$ is a message passing operation applied on the vertex features. Source COMA [88].

sentence. In the same way that we are able to extract relevant features from images and words, in 3D geometry, we want to extract features from manifolds such as meshes. Since GNNs work on graphs, their layers must implement some methods to modify the input features attached to the nodes of the graphs, and output new representations of them in potential new graphs. These layers are called message-passing layers and are the building blocks of a GNN, in the same way that convolutional layers are building blocks for CNNs and self-attensions are for Transformers. Before describing the network and its message passing layers let us first discuss the inputs and the motivation for the choice of the network in our case. The face mesh connectivity in 3D defines a structure on the position of the landmarks in 2D. Therefore, we investigate whether the explicit use of their inherent geometric structure, enforced with a GNN, will improve the accuracy of the predicted landmarks. For an in-depth overview of the general operations on graph neural networks we direct the interested reader to Bronstein et al. [6] as we are only going to explain the versions used in our work.

In order to capture global and local context, as well as for efficiency, we build a decoder that works on downsampled versions of the mesh. COMA [88] introduces a mesh sampling operation, see Fig. 3.8 for a schematic description. It can be viewed as a pooling operation on meshes. The mesh resolution is iteratively reduced by contracting vertex pairs so as to minimize surface error using quadratic metrics [30]. Parallel to the downsampling operation that gives downsamplig matrices Q_d, this process returns upsampling matrices Q_u used to recover the higher resolution mesh. The contracted vertices are computed using the barycentric coordinates of the closest triangle in the decimated mesh.

Let $\boldsymbol{V}_L$ be the face vertices of the FLAME [65] model. We use the operations described above to compute m downsample Q_d and upsample Q_u matrices. We iteratively downsample our mesh m times to obtain a new set of vertices $\boldsymbol{V}^m$ with size $D = |\boldsymbol{V}^m| < \mathtt{L}$. Instead of the 3D vertex positions we generalize our mesh and use a linear map to attach to each vertex the features $\mathcal{F} \in \mathbb{R}^C$ computed by the backbone. We denote this new set by $\boldsymbol{X}^m \in \mathbb{R}^{D \times C}$ which is going to be the input mesh feature of the GNN decoder. Note that each element $\boldsymbol{x}^m \in \boldsymbol{X}^m$ is a vertex feature. At every layer k, the current mesh feature $\boldsymbol{X}^{k-1}$ undergoes upsampling:

$$\boldsymbol{U}^{k-1} = Q_u^{k-1} \boldsymbol{X}^{k-1} \tag{3.10}$$

In SpiralNet++ [35] Gong et al. define their parallel to the convolution operation called Spiral Convolution. The vertices aggregate neighbor features along a pre-computed spiral around them

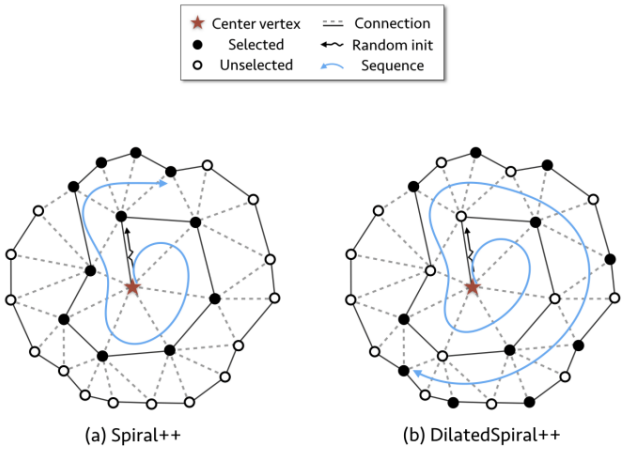

Figure 3.9: Example of spiral path for the computation of the features of a central node (left). Dilated version that allows to keep the length but skip vertex features (right). Source SpiralNet++ [35].

and a linear map follows to encode the newly aggregated vertex features. Figure 3.9 shows an example of a spiral around a central vertex. Let $\boldsymbol{u}^{k-1} \in \boldsymbol{U}^{k-1}$ be a vertex feature after upsampling. We can define the Spiral Convolution for vertex i at layer k:

$$\mathcal{C}\text{onv}_i(\boldsymbol{U}^{k-1}) = A^{k-1} \, ||_{j \in S_{i,l}} \, \boldsymbol{u}_j^{k-1} \tag{3.11}$$

where A^{k-1} is a linear map, $||$ is the concatenation operator, $S_{i,l}$ is the index set of the pre-computed spiral around vertex i of length l. Finally by applying a non linear function ELU [18] after the convolution, the spiral message passing layer of the GNN decoder can be defined:

$$x_i^k = \text{ELU}[\mathcal{C}\text{onv}_i(\boldsymbol{U}^{k-1})] \tag{3.12}$$

In summary at every layer the mesh undergoes an un-pooling operation, i. e. it is multiplied with the corresponding upsampling matrix. Then the features attached to the vertices are concatenated following a spiral around each vertex and linearly mapped to fit a suited dimension, finally a non-linear activation function ELU is applied.

3.3.3 Loss

Regression methods are more efficient than heatmap methods. They get rid of the high resolution 2D grids and therefore have smaller computational and storage complexity. Moreover, they have a continuous output and do not suffer from quantization issues. However, their performance lags behind heatmap based methods. Li et al. [64] introduce an efficient regression method that remedies this issue, outperforming the state-of-the-arts heatmap methods in Human Pose detection.

Instead of letting the model learn landmark positions, the model learns a distribution over landmark positions, conditioned on the input images. This formulation enables modeling the uncertainty of the prediction. Therefore, we represent our landmarks as i.i.d 2D random variables following an unknown distribution that we aim to estimate. Formally, in the context of Maximum Likelihood Estimation (MLE) we want to minimize the following loss:

$$\mathcal{L} = -\log P_\theta(\boldsymbol{x}|\mathrm{I})|_{\boldsymbol{x}=\boldsymbol{\mu}_{gt}}, \tag{3.13}$$

where $P_\theta(\boldsymbol{x}|I)$ is the underlying distribution of landmarks given an image I, θ are the parameters of the model, and $\boldsymbol{\mu}_{gt}$ are the ground truth landmarks. In other words taking into consideration the ambiguities in the labels we want to find distribution parameters that make the $\boldsymbol{\mu}_{gt}$ labels most probable. For this formulation different losses are different assumptions on the probability distribution. If we assume that the landmark positions follow a Gaussian distribution, then the predicted values from the network are the mean $\hat{\boldsymbol{\mu}}$ and variance $\hat{\boldsymbol{\sigma}}$. The negative log-likelihood loss then becomes:

$$\mathcal{L} \propto \log \hat{\boldsymbol{\sigma}} + \frac{(\boldsymbol{\mu}_{gt} - \hat{\boldsymbol{\mu}})^2}{2\hat{\boldsymbol{\sigma}}^2}. \tag{3.14}$$

We call this loss the Negative Gaussian Log Likelihood (NGLL). Note that this is the loss used by Wood et al. in [114] for landmark prediction. For constant variance this reduces to standard L_2 regression loss. If we instead use a Laplacian distribution (NLLL) with constant variance, then the loss function is simply the L_1 regression loss.

Li et al. [64] choose to use Normalizing Flows [91] to learn a more expressive distribution function, rather than a pre-defined one. The main idea of Normalizing Flows is to represent a complex distribution by transforming a simple distribution through invertible mappings. Let $\boldsymbol{z}$ be the random variable that follows the initial simple distribution $P_\theta(\boldsymbol{z}|I)$. For example, this distribution can be a Gaussian with mean $\hat{\boldsymbol{\mu}}$ and variance $\hat{\boldsymbol{\sigma}}$ dependent on the model parameters θ as before. Let $\boldsymbol{x} = f_\phi(\boldsymbol{z})$ be the random variable that follows a new distribution that better suits the landmarks and found through the invertible mapping f_ϕ, dependent on the flow model parameters ϕ. The distribution of $\boldsymbol{x}$ is now:

$$\log P_{\theta,\phi}(\boldsymbol{x}|I) = \log P_\phi(\boldsymbol{z}|I) + \log \left| \det \frac{\partial f_\phi^{-1}(\boldsymbol{x})}{\partial \boldsymbol{x}} \right| \tag{3.15}$$

$z = f_\phi^{-1}(\boldsymbol{x})$ can be computed given any $\boldsymbol{x}$ and hence $\log P_{\theta,\phi}(\boldsymbol{x}|I)$ can be estimated. The MLE loss becomes:

$$\begin{aligned} \mathcal{L} &= -\log P_{\theta,\phi}(\boldsymbol{x}|I)|_{\boldsymbol{x}=\boldsymbol{\mu}_{gt}} \\ &= -\log P_\theta(f_\phi^{-1}(\boldsymbol{\mu}_{gt})|I) - \log \left| \det \frac{\partial f_\phi^{-1}(\boldsymbol{\mu}_{gt})}{\partial \boldsymbol{x}} \right| \end{aligned} \tag{3.16}$$

Figure 3.10 shows a visual representation of this basic framework. f_ϕ^{-1} is going to be learned by a flow network. Li et al. [64] use an architecture called RealNVP [23], where at layer k the output of the D dimensional input vector is computed in the following way:

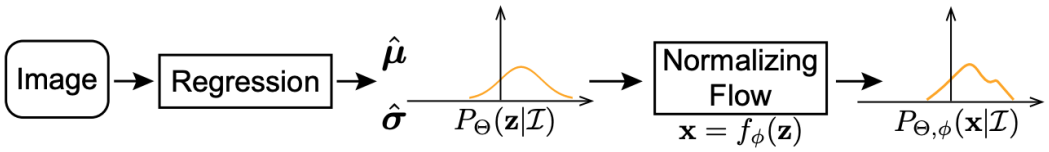

Figure 3.10: Basic framework with normalizing flow. Source Li et al. [64].

$$(\boldsymbol{z}^k_{0:d}\ ,\ \boldsymbol{z}^k_{d:D}) = (\boldsymbol{z}^{k-1}_{0:d}\ ,\ \boldsymbol{z}^{k-1}_{d:D} \odot \exp\left(s(\boldsymbol{z}^{k-1}_{0:d})\right) + t(\boldsymbol{z}^{k-1}_{0:d})), \tag{3.17}$$

where $\boldsymbol{z}^{k-1}_{0:d}$ are the first d entries of the output of the previous layer, likewise $\boldsymbol{z}^{k-1}_{d:D}$ are the remaining entries. $\odot$ represents the point wise product and $s, t : \mathbb{R}^d \to \mathbb{R}^{D-d}$ are two arbitrary neural networks. The input is permuted before each layer. These mappings allow for exact log-likelihood computation and exact and efficient sampling.

In Equation (3.16) the learning of the parameters ϕ by the flow network is driven by the terms $f^{-1}_\phi(\boldsymbol{\mu}_{gt})$ and $\dfrac{\partial f^{-1}_\phi}{\partial \boldsymbol{x}}(\boldsymbol{\mu}_{gt})$ in the loss that only depend on $\boldsymbol{\mu}_{gt}$. This way the flow network is going to learn about the variation of the ground truth landmarks across the images. What we want instead is the distribution of how the ground truth landmarks deviate from the output prediction given an image. We further apply the following reparameterization:

$$\boldsymbol{x} = \bar{\boldsymbol{x}} * \hat{\boldsymbol{\sigma}} + \hat{\boldsymbol{\mu}} \tag{3.18}$$

where $\bar{\boldsymbol{x}} = f_\phi(\boldsymbol{z})$ with starting distribution $\boldsymbol{z} \sim N(0, I)$. The MLE loss becomes:

$$\begin{aligned}
\mathcal{L} &= -\log P_{\theta,\phi}(\boldsymbol{x}|I)|_{\boldsymbol{x}=\boldsymbol{\mu}_{gt}} \\
&= -\log P_\phi(\bar{\boldsymbol{x}}(\boldsymbol{\mu}_{gt})) - \log \left| \det \frac{\partial \bar{\boldsymbol{x}}(\boldsymbol{\mu}_{gt})}{\partial \boldsymbol{x}} \right| \\
&= -\log P_\phi(\bar{\boldsymbol{x}}(\boldsymbol{\mu}_{gt})) + \log \hat{\boldsymbol{\sigma}}
\end{aligned} \tag{3.19}$$

With this reparameterization, parameters ϕ with parameters θ are coupled through $\bar{\boldsymbol{x}}$ and the flow network can learn how to model the right distribution. Figure 3.11 illustrates the new framework. The training of the landmarks model now entirely relies on the distribution estimated by the flow model. At the beginning of training the shape of the distribution might be too off and degrade the model performance. To address this Li et al. [64] further analyze $P_\phi(\bar{\boldsymbol{x}})$. We can split $\log P_\phi(\bar{\boldsymbol{x}})$ into three terms:

$$\begin{aligned}
\log P_\phi(\bar{\boldsymbol{x}}) &= \log \left(s * Q(\bar{\boldsymbol{x}}) * \frac{P_\phi(\bar{\boldsymbol{x}})}{s * Q(\bar{\boldsymbol{x}})} \right) \\
&= \log s + \log Q(\bar{\boldsymbol{x}}) + \log \left(\frac{P_\phi(\bar{\boldsymbol{x}})}{s * Q(\bar{\boldsymbol{x}})} \right) \\
&= \log s + \log Q(\bar{\boldsymbol{x}}) + \log G_\phi(\bar{\boldsymbol{x}})
\end{aligned} \tag{3.20}$$

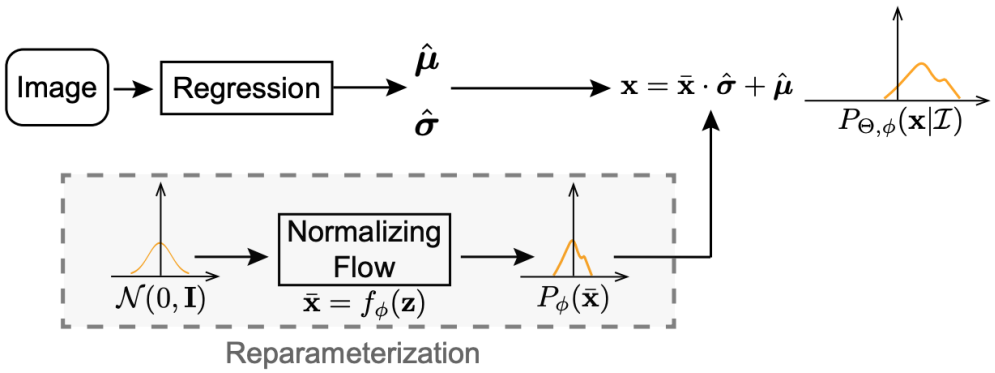

Figure 3.11: Framework with re-parameterization. Source Li et al. [64].

$Q(\bar{\boldsymbol{x}})$ is a distribution, $G_\phi(\bar{\boldsymbol{x}})$ is what we call the residual distribution and s is a constant in order to make $G_\phi(\bar{\boldsymbol{x}})$ a distribution. The final MLE loss called Residual Log-Likelihood (ResLL) loss becomes:

$$\begin{aligned}
\text{ResLL} &= -\log P_{\theta,\phi}(\boldsymbol{x}|I)|_{\boldsymbol{x}=\boldsymbol{\mu}_{gt}} \\
&= -\log P_\phi(\bar{\boldsymbol{x}}(\boldsymbol{\mu}_{gt})) - \log \left| \det \frac{\partial \bar{\boldsymbol{x}}(\boldsymbol{\mu}_{gt})}{\partial \boldsymbol{x}} \right| \\
&= -\log P_\phi(\bar{\boldsymbol{x}}(\boldsymbol{\mu}_{gt})) + \log \hat{\boldsymbol{\sigma}} \\
&= -\log s - \log Q(\bar{\boldsymbol{x}}(\boldsymbol{\mu}_{gt})) - \log G_\phi(\bar{\boldsymbol{x}}(\boldsymbol{\mu}_{gt})) + \log \hat{\boldsymbol{\sigma}} \\
&= -\log Q\left(\bar{\boldsymbol{x}}(\boldsymbol{\mu}_{gt})\right) - \log G_\phi(\bar{\boldsymbol{x}}(\boldsymbol{\mu}_{gt})) + \log \hat{\boldsymbol{\sigma}} \\
&= -\log Q\left(\frac{\boldsymbol{\mu}_{gt} - \hat{\boldsymbol{\mu}}}{\hat{\boldsymbol{\sigma}}}\right) - \log G_\phi\left(\frac{\boldsymbol{\mu}_{gt} - \hat{\boldsymbol{\mu}}}{\hat{\boldsymbol{\sigma}}}\right) + \log \hat{\boldsymbol{\sigma}}
\end{aligned} \tag{3.21}$$

Effectively now the flow network learns only the residual distribution G_ϕ which makes training faster. Q can be any distribution such as a Gaussian or Laplacian. The same argument made for the ResNet [40] in Sec. 3.3 for residual learning, can be made here. It is easier to learn the residual than the whole distribution or in the case that Q is optimal push the residual to zero. In Fig. 3.12 the final framework is shown. In the training phase both the predictor and flow model are optimized simultaneously. During inference the predicted landmarks $\hat{\boldsymbol{\mu}}$ serve as output so the flow model does not need to be run. Other than landmarks, the predictor estimates $\hat{\boldsymbol{\sigma}} \in [0, 1]^{2\text{L}}$ trough a sigmoid function on the last layer. It can be seen as a scaling factor for the variance of the final distribution. By looking at the reparameterization in Eq. (3.18) it easy to notice that the more the model is confident on the prediction of $\hat{\boldsymbol{\mu}}$ the closer to zero $\hat{\boldsymbol{\sigma}}$ is going to be. Therefore we can directly acquire confidences for the landmarks and use them as weights for the fitting of the FLAME [65] model.

$$\hat{\boldsymbol{C}} = 1 - \hat{\boldsymbol{\sigma}} \tag{3.22}$$

Where $\hat{\boldsymbol{C}} \in [0, 1]^{2\text{L}}$. On the other hand the usage of the sigmoid function to predict $\hat{\boldsymbol{\sigma}} \in [0, 1]^{2\text{L}}$

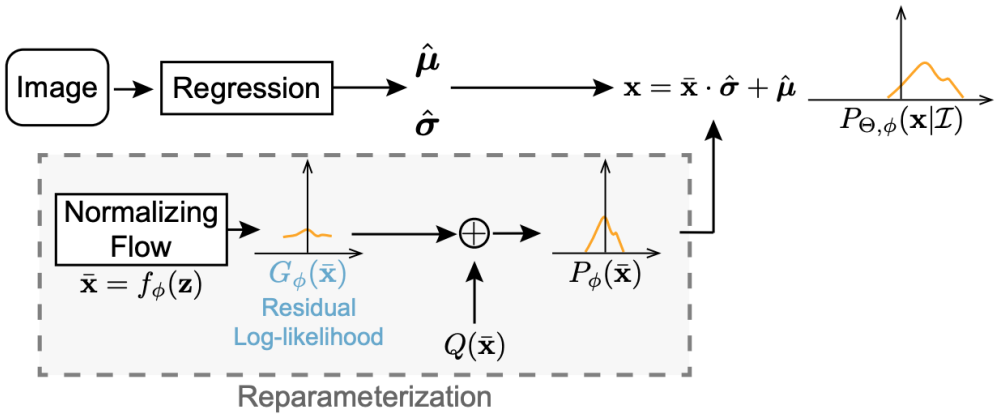

Figure 3.12: Final framework with reparameterization and residual distribution. Source Li et al. [64].

may be detrimental for learning. In fact due to the shape of the sigmoid function the learning could get stuck at the end points of the domain. For this reason we experiment on removing the sigmoid function only enforcing $\hat{\boldsymbol{\sigma}} \geq 0$ by exponentiating the prediction. We call this variant ResLL Exp. Because of Eq. (3.18) the predictor is still going to output values close to zero and confidences $\hat{C}$ can still be computed with Eq. (3.22) making sure to clip values at zero if it is the case. In addition to using the ResLL loss we experiment on adding a loss over the distances between landmarks. The reason is that by enforcing the model to keep the distances we would allow it to better learn positions for challenging points that may be occluded. As with the addition of the graph decoder this is another way to enforce the geometric structure of the landmarks given by the face. Let E be the edge set given by the face mesh and $e \in E$ an edge. We consider $\hat{e}$ to be the predicted landmark difference of the corresponding vertices in e and e_{gt} to be the ground truth landmark difference. Then we define the Vertex-Edge loss to be:

$$\mathcal{L}_{VE} = \sum_{e \in E} \left(\hat{e} - e_{gt} \right)^2 \tag{3.23}$$

3.4 3D Face Estimation From 2D Landmarks

3.4.1 Non-Linear Least Squares

Non-linear least squares (NLS) is a class of optimization problems that arise when we want to fit non-linear functions $\boldsymbol{f}_i(\boldsymbol{x}|\boldsymbol{\nu}^i)$ to some data $\boldsymbol{y}_i$. We denote by $\boldsymbol{\nu}^i \subset \boldsymbol{\nu}$ non-disjoint subsets of a shared parameter space $\boldsymbol{\nu}$. These problems consist in minimizing the square residual difference between the estimated values and the data:

$$
\begin{aligned}
\boldsymbol{\nu}^{\star} &= \arg\min_{\boldsymbol{\nu}} \mathcal{S}(\boldsymbol{\nu}) \\
&= \arg\min_{\boldsymbol{\nu}} \frac{1}{2} \sum_{i} ||\boldsymbol{r}_i(\boldsymbol{\nu}^i)||^2 \\
&= \arg\min_{\boldsymbol{\nu}} \frac{1}{2} \sum_{i} ||\boldsymbol{f}_i(\boldsymbol{x}|\boldsymbol{\nu}^i) - \boldsymbol{y}_i||^2
\end{aligned}
\tag{3.24}
$$

Direct minimization of Eq. (3.24) can not usually be applied since the system of equations formed by setting the derivatives of the objective $\mathcal{S}$ to zero may not in general have a closed solution. Instead iterative procedures that make use of Taylor approximations are usually implemented. If we consider the second order Taylor approximation of $\mathcal{S}$ around a point $\boldsymbol{\nu}_0$ in $\Delta\boldsymbol{\nu} = \boldsymbol{\nu}_0 - \boldsymbol{\nu}$ we obtain:

$$
\begin{aligned}
\mathcal{S}(\boldsymbol{\nu}) &= \mathcal{S}(\boldsymbol{\nu}_0 - \Delta\boldsymbol{\nu}) \\
&\approx \mathcal{S}(\boldsymbol{\nu}_0) - \frac{\partial \mathcal{S}(\boldsymbol{\nu}_0)}{\partial \boldsymbol{\nu}} \Delta\boldsymbol{\nu} + \frac{1}{2} \Delta\boldsymbol{\nu}^T \frac{\partial^2 \mathcal{S}(\boldsymbol{\nu}_0)}{\partial^2 \boldsymbol{\nu}} \Delta\boldsymbol{\nu}
\end{aligned}
\tag{3.25}
$$

Since we want to find the best update $\Delta\boldsymbol{\nu}$ of the parameters $\boldsymbol{\nu}$ we consider this a function of $\Delta\boldsymbol{\nu}$ and minimize it by setting its derivative to zero. We obtain the Normal Equation:

$$
-\frac{\partial \mathcal{S}(\boldsymbol{\nu}_0)}{\partial \boldsymbol{\nu}} + \frac{\partial^2 \mathcal{S}(\boldsymbol{\nu}_0)}{\partial^2 \boldsymbol{\nu}} \Delta\boldsymbol{\nu} = 0 \tag{3.26}
$$

Using the definition of the objective in Eq. (3.24) the first term, the gradient of the objective, can be computed:

$$
\begin{aligned}
\frac{\partial \mathcal{S}(\boldsymbol{\nu}_0)}{\partial \boldsymbol{\nu}} &= \sum_{i} \frac{\partial \boldsymbol{r}_i^T(\boldsymbol{\nu}_0^i)}{\partial \boldsymbol{\nu}_0} \boldsymbol{r}_i(\boldsymbol{\nu}_0^i) \\
&= \sum_{i} J^T(\boldsymbol{\nu}_0^i) \boldsymbol{r}_i(\boldsymbol{\nu}_0^i)
\end{aligned}
\tag{3.27}
$$

and the second term which correspond to the Hessian of the objective is:

$$
\frac{\partial^2 \mathcal{S}(\boldsymbol{\nu}_0)}{\partial^2 \boldsymbol{\nu}} = \sum_{i} J^T(\boldsymbol{\nu}_0^i) J(\boldsymbol{\nu}_0^i) + \frac{\partial J(\boldsymbol{\nu}_0^i)}{\partial \boldsymbol{\nu}} \boldsymbol{r}_i(\boldsymbol{\nu}_0^i) \tag{3.28}
$$

If the Hessian is not positive semi definite then the update could lead to convergence to a local optimum such as a saddle point or a maximum. To guarantee convergence to a minimum the Hessian can be approximated by dropping the second term in the sum:

$$
\frac{\partial^2 \mathcal{S}(\boldsymbol{\nu}_0)}{\partial^2 \boldsymbol{\nu}} \approx \sum_{i} J^T(\boldsymbol{\nu}_0^i) J(\boldsymbol{\nu}_0^i) \tag{3.29}
$$

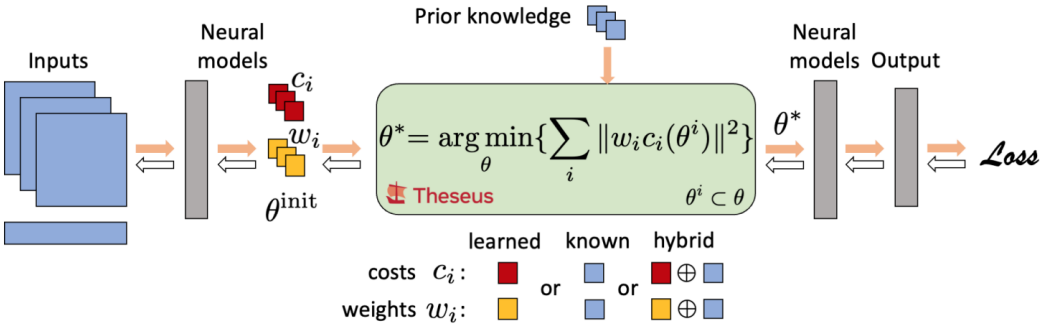

Figure 3.13: Example of a Theseus layer inside a neural network. Source Theseus [84].

Finally by rewriting Eq. (3.26) and dropping the evaluation in $\boldsymbol{\nu}_0$ we obtain the classic Normal Equations:

$$(\sum_i J_i^T J_i)\Delta\boldsymbol{\nu} = (\sum_i J_i^T \boldsymbol{r}_i) \tag{3.30}$$

At each iteration we solve the normal equations for $\Delta\boldsymbol{\nu}$ and update $\boldsymbol{\nu} \leftarrow \boldsymbol{\nu} - \gamma\Delta\boldsymbol{\nu}$ until convergence. γ is the step length. This method is called Gauss-Newton and is a second order approximation method to solve NLS. The first order approximation method, Gradient Descent is obtained only by using the first order Taylor approximation, obtaining $\Delta\boldsymbol{\nu} = (\sum_i J_i^T \boldsymbol{r}_i)$. The Levenberg-Marquardt [62, 76] method improves on the Gauss-Newton method and changes Eq. (3.30) by adding a damping parameter λ:

$$(\sum_i J_i^T J_i)\Delta\boldsymbol{\nu} + \lambda\Delta\boldsymbol{\nu} = (\sum_i J_i^T \boldsymbol{r}_i) \tag{3.31}$$

In our work we build on Theseus [84], an application agnostic differentiable non-linear optimization library built on top of Pytorch [82]. Theseus provides a framework for solving Eq. (3.24), e. g. using the Gauss-Newton method described above. One important advantage is that it enables differentiation through the NLS problem and hence can be used as a layer in a neural network. Figure 3.13 illustrates this idea. Specifically, it provides gradients for the optimal solution $\boldsymbol{\nu}^\star$ with respect to any parameter network $\boldsymbol{\phi}$ for which the objective can now depend. Equation (3.24) becomes:

$$\boldsymbol{\nu}^\star(\boldsymbol{\phi}) = \arg\min_{\boldsymbol{\nu}} \mathcal{S}(\boldsymbol{\nu}|\boldsymbol{\phi}) \tag{3.32}$$

As a consequence the objective $\mathcal{S}$ can be treated as an inner optimization problem and be part of an outer classical neural network optimization $\mathcal{N}$ with gradient descent for $\boldsymbol{\phi}$.

$$\boldsymbol{\phi}^\star = \arg\min_{\boldsymbol{\phi}} \mathcal{N}(\boldsymbol{\nu}^\star(\boldsymbol{\phi})) \tag{3.33}$$

In Theseus the residuals $\boldsymbol{r}_i$ are decoupled in weights $\boldsymbol{w}_i$ and cost vector functions $\boldsymbol{c}_i$ allowing a more flexible representation:

$$\boldsymbol{r}_i(\boldsymbol{\nu}^i) = \boldsymbol{w}_i \boldsymbol{c}_i(\boldsymbol{\nu}^i) \tag{3.34}$$

Theseus has efficient design choices. Contrary to Pytorch native linear solvers that are dense it implements differentiable sparse linear solvers to solve Eq. (3.30). Moreover it adds batching and automatic vectorization of cost functions decreasing computation time. It introduce implicit differentiation and direct loss minimization. Automatic differentiation through Pytorch as well as common cost functions with analytical jacobians are provided. For a detailed analysis of the library we direct the interested reader to the original paper of Pineda et al. [84].

3.4.2 Camera

We use a perspective projection camera model Π with the following camera intrinsics:

$$K = \begin{pmatrix} f & 0 & W/2 \\ 0 & f & H/2 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.35}$$

where f is the focal length and H, W are the height and width of the image. The projection of a 3D point $v = (x, y, z)$ to the image plane is then:

$$\Pi(v; K) = \left[f\frac{x}{z} + \frac{W}{2}, f\frac{y}{z} + \frac{H}{2}, \right]^T \tag{3.36}$$

Since we do not know the camera parameters and we do not estimate them we use approximations taken from Kissos et al. [53]. Focal length can be computed from the camera field of view:

$$f = \frac{\sqrt{H^2 + W^2}}{2\tan(\alpha/2)} \tag{3.37}$$

Not knowing the diagonal field of view α we can approximate the focal length by dropping it:

$$f \approx \sqrt{H^2 + W^2} \tag{3.38}$$

Kissos et al. [53] show that this is a reasonable approximation since a model is usually not affected by the exact value of the focal length within a wide range and hence accurate approximation does not deteriorate the results.

3.4.3 Model Fitting

We compute the residual vectors $\boldsymbol{r}$ between predicted landmarks $\hat{\boldsymbol{L}}$ and FLAME [65] projected face vertices $\Pi(\boldsymbol{V}_L(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\psi}); K)$, using our previously defined camera projection model Π for a given image by:

$$\boldsymbol{r}(\hat{\boldsymbol{L}}, \boldsymbol{V}_L(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\psi}); K) = \hat{\boldsymbol{L}} - \Pi(\boldsymbol{V}_L(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\psi}); K) \tag{3.39}$$

where the predicted landmarks are equal to the mean of the predicted distribution: $\hat{\boldsymbol{L}} = \hat{\boldsymbol{\mu}}$. The re-projection loss is defined using the predicted confidences $\hat{\boldsymbol{C}}$ as weights:

$$\mathcal{L}_{\text{RP}}\left(\hat{\boldsymbol{L}}, \hat{\boldsymbol{C}}, \boldsymbol{V}_L(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\psi}); \Pi\right) = \frac{1}{2} \|\hat{\boldsymbol{C}} \boldsymbol{r}\left(\hat{\boldsymbol{L}}, \boldsymbol{V}_L(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\psi}); \Pi\right)\|^2 \tag{3.40}$$

We estimate a 3D face by fitting FLAME parameters to the detected 2D landmarks in two stages. First, we estimate the root rotation and translation of the model:

$$\text{Rigid Fitting (RF)} := \arg\min_{\boldsymbol{\theta}_r} \mathcal{L}_{\text{RP}}\left(\hat{\boldsymbol{L}}, \hat{\boldsymbol{C}}, \hat{\boldsymbol{V}}_{\boldsymbol{L}}(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\psi}); \Pi\right) \tag{3.41}$$

In the second stage, we optimize over all FLAME parameters :

$$\text{Non-Rigid Fitting (NRF)} := \arg\min_{\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\psi}} \mathcal{L}_{\text{RP}}\left(\hat{\boldsymbol{L}}, \hat{\boldsymbol{C}}, \hat{\boldsymbol{V}}_{\boldsymbol{L}}(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\psi}); \Pi\right) \tag{3.42}$$

The RF stage helps avoid bad minima by first aligning the face mesh to the image, so that the second stage, NRF, can explain shape, expression, and articulation details.

4. Experiments and Results

4.1 Evaluation Metrics

To evaluate the accuracy of landmarks prediction we use the Mean Landmark Error (MLME), computed as the mean distance between the estimated $\hat{\boldsymbol{L}}$ and pseudo ground-truth $\boldsymbol{L}$, 2D landmarks:

$$\text{MLME} = \frac{1}{\texttt{NL}} \sum_{i=1}^{\texttt{N}} \sum_{k=1}^{\texttt{L}} ||\hat{\boldsymbol{L}}_{ik} - \boldsymbol{L}_{ik}||_2 \tag{4.1}$$

where $\texttt{N} = 2156$ is the number of images and $\texttt{L} = 1609$ is the number of landmarks.

We use two metrics to measure the quality of 3D estimation. The Mean Vertex Error (MVE) between a subset $S = H, L$ of face or head estimated $\hat{\boldsymbol{V}}_S$ and pseudo ground-truth $\boldsymbol{V}_S$, 3D vertices, and the Mean Re-projected Landmark Error (MRLE) between estimated projected face vertices $\Pi(\hat{\boldsymbol{V}}_{\boldsymbol{L}}; K)$ and estimated 2D landmarks $\hat{\boldsymbol{L}}$. The definition of the metrics are:

$$\text{MVE} = \frac{1}{\texttt{NV}} \sum_{i=1}^{\texttt{N}} \sum_{k=1}^{\texttt{V}_S} \| \hat{\boldsymbol{V}}_{S,ik} - \boldsymbol{V}_{S,ik} \|_2 \tag{4.2}$$

$$\text{MRLE} = \frac{1}{\texttt{NL}} \sum_{i=1}^{\texttt{N}} \sum_{k=1}^{\texttt{L}} \| \Pi(\hat{\boldsymbol{V}}_{L,ik}; K) - \hat{\boldsymbol{L}}_{ik} \|_2 \tag{4.3}$$

The number of vertices $\texttt{V}_S$, is either equal to $\texttt{V}_\texttt{L} = 1609$ when computing the error on the face subset mesh, or $\texttt{V}_\texttt{H} = 4757$ when instead we consider the full FLAME [65] head without the neck. We are going to clearly separate the two cases in the results tables by referring to the error MVE Face, considering the face mesh and MVE Head when using the full head without neck. Refer to Fig. 3.4 for a visualization of the two vertex sets.

4.2 Landmark Prediction

As a baseline for evaluation, we recreate the pipeline implemented by Wood et al. in [114]. They use a ResNet architecture as predictor and a NGLL loss, defined in Eq. (3.14). We first evaluate the impact of the loss by implementing the ResLL loss introduced by Li et al. in [64], described in Eq. (3.21) and the unrestricted variant, ResLL Exp, introduced in Sec. 3.3.3. We include a secondary loss, the Vertex-Edge loss, described in Eq. (3.23). For this experiment, we only use a ResNet-50 [40] as predictor that outputs confidences $\hat{\boldsymbol{C}}$ and landmarks $\hat{\boldsymbol{L}}$. Table 4.1 contains the results. We observe that NGLL performs better than ResLL, but that the variant with unrestricted confidences, ResLL Exp, is the most accurate. We attribute this to two main factors. The first is that ResLL relies on the normalizing flow network to combine learned mappings to make the simple starting distribution more complex. The overall learning of the distribution may

Table 4.1: From left to right: (1) the loss used, ResLL of Li et al. [64], NGLL used by Wood et al. [114], (2) whether the Vertex-Edge loss is used, (3) Mean landmark error.

Loss	VE Loss	MLME (px) ↓
NGLL	✗	2.17
NGLL	✓	2.18
ResLL	✗	2.20
ResLL	✓	2.18
ResLL Exp	✗	**2.11**
ResLL Exp	✓	2.13

therefore be slower and require longer training time than for NGLL. The second factor is the sigmoid function that restricts the confidences to $[0, 1]$. In particular, because of its structure, the sigmoid function may have an impact on the gradients, resulting in too small updates and therefore making learning slower. This idea is enforced by ResLL Exp scoring the most accurate result. ResLL enables more powerful and accurate probabilistic descriptions and over longer training we would expect the accuracy, even for the original formulation, to be better than for the simpler NGLL. The same argument can be made for the Vertex-Edge loss. It should help in accuracy but it may require longer training times. Since we are evaluating the average landmark error it could also be the case that the positions of landmarks with respect to each other are better preserved with the addition of the Vertex-Edge loss, despite the fact that mean accuracy is worse.

We investigate the impact of network architecture by evaluating average accuracy for the different architectures introduced in Sec. 3.3.1. We use ResLL Exp as primary loss for these experiments. Results for backbone experiments can be seen in Tab. 4.2. We notice that convolution based architectures perform better. In particular the ConvNeXt [73] architecture reaches the best accuracy, further suggesting that convolution operations may be better able to extract meaningful features for image analysis than self-attentions, but at the same time, that the design choices of transformers may be superior. The choice of optimizer, Adam[52] in this case, may also play a role in the performance difference. We note that because of hardware constraints we were only able to train the small network variants and that bigger models may lead to different results. We further evaluate architecture performance by adding the GNN decoder SpiralNet++ [35] described in Sec. 3.3.1 using ResNet and ConvNeXt as backbones. The experiment's results are shown in Tab. 4.3. We find that the best average landmark 2D accuracy is reached by enforcing the geometric structure of the face through the GNN. It is an interesting result since geometric deep learning methods are usually applied for 3D problems and as shown here, can help improve results even for the 2D case.

Table 4.2: Left to right: (1) Backbone used, (2) whether the Vertex-Edge loss is used, (3) Mean landmark error.

Backbone	VE Loss	MLME (px) ↓
ResNet-50	✗	2.11
ResNet-50	✓	2.13
ConvNeXt Tiny	✗	2.13
ConvNeXt Tiny	✓	**2.07**
Swin Transformer Tiny	✗	2.38
Swin Transformer Tiny	✓	2.35
MobileNet V3 Large	✗	2.55
MobileNet V3 Large	✓	2.57

Table 4.3: From left to right: (1) Backbone used, (2) whether the GNN decoder SpiralNet++ is used, (3) whether the Vertex-Edge loss is used, (4) Mean landmark error.

Backbone	GNN Decoder	VE Loss	MLME (px) ↓
ResNet-50	✗	✗	2.11
ResNet-50	✗	✓	2.13
ResNet-50	✓	✗	2.26
ResNet-50	✓	✓	**2.01**
ConvNeXt Tiny	✗	✗	2.13
ConvNeXt Tiny	✗	✓	2.07
ConvNeXt Tiny	✓	✗	2.10
ConvNeXt Tiny	✓	✓	2.18

4.3 3D Estimation

For 3D estimation we first evaluate the average vertex error MVE and average reprojected error MRLE of the reconstructions obtained with the landmarks predicted by the models in Tab. 4.3. 2D-3D combined results can be seen in Tab. 4.4 and timings of the estimations in Tab. 4.5. With respect to the timings it seems ResNet takes a little more time to estimate the landmarks than ConVnext whereas for the fitting the times are very similar. In general the fitting optimization takes a long time at the moment, we will discuss in Chapter 5 ways to improve on this. With respect to estimation accuracy, as expected since the model is fit to the landmarks, the errors for the face set are smaller compared to the errors over the head set. We notice that the best average vertex errors for both head and face are achieved with the addition of the GNN decoder. Further indicating that the decoder can help for the overall 3D structure accuracy. With respect to the backbone, ConvNext is superior in performance, suggesting that that the over-all structure of the landmarks is better retained with this network than with ResNet. Moreover we see that the

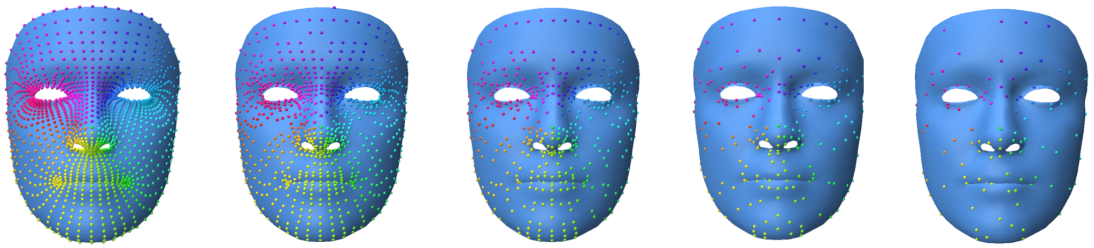

Figure 4.1: From left to right: (1) 1609, (2) 805, (3) 403, (4) 202, (5) 101 landmark subsets of the FLAME [65] face mesh used for fitting.

best average vertex accuracy does not correspond to best average landmark accuracy suggesting that vertex estimation may be rather unsensitive to very accurate landmarks, provided that there are many of them and the structure is preserved. To further investigate this we experiment on how many landmarks are needed for an accurate reconstruction. We use the mesh operations described in Sec. 3.3.2 to define subsets from the full mask of $\mathtt{L} = 1609$ of $805, 403, 202, 101$ landmarks. Note that we still compute the MVE over the original sets but we use the smaller subsets for fitting the model. Visualization of the subsets can be seen in Fig. 4.1 over the face mesh and results of the experiments in Tab. 4.6. Interestinlgy we obtain the subset of landmarks that performs better, which is of around 800 points. The reason is perhaps because some of the points of the contour of the face are missing in this subset. We have noticed that the network struggles at being accurate for the outer region of the face more than the inner region. Another consideration to make is that the landmark networks estimate the full set. Hence they may have learned to position the landmarks with accuracy because they were trained on the full face and that we can drop some of the landmarks for reconstruction because they are all quite accurate. In general, even if slowly, accuracy of estimation drops by dropping landmarks and we would expect it to drop more consistently if we were to etimate less landmarks. We need to conduct further experiments in order to find the best subset of landmarks. Counterintuitively the times increase when dropping landmarks. This can perhaps be explained by the fact that with less information the optimizer struggles more in finding the optima than with more information.

Finally for comparison on how other 3D reconstruction methods perform over this dataset we run DECA [27] over the test set for which we evaluate the MVE on the coarse reconstruction. We also run DAD-3DNet [77] that has been trained on the same dataset. Results are shown in Table 4.7. On this test set we obtain better accuracy then the other methods, suggesting that dense landmarks can indeed be a viable, more accurate and simpler way to obtain 3D estimation from single images.

4.4 Qualitative Results

In Fig. 4.2 we have gathered example images from the test set. We can see that our network is able to learn meaningful confidence values when there are occlusions. Moreover jointly using the confidences with the fitting pipeline allows for reconstructions that sometimes are perhaps more plausible than the pseudo labels, for example in reconstructing the shape of the head.

Table 4.4: From left to right: (1) Backbone used, (2) whether the GNN decoder SpiralNet++ is used, (3) whether the Vertex-Edge loss is used, (4) Mean vertex error over face and head vertices, (5) Mean reprojected error over landmarks.

				MVE (mm) ↓		
Backbone	GNN Decoder	VE Loss	MLME (px) ↓	Face	Head	MRLE (px)↓
ResNet-50	✗	✗	2.11	07.9	10.7	**0.48**
ResNet-50	✗	✓	2.13	08.2	11.2	0.49
ResNet-50	✓	✗	2.26	08.0	11.2	0.76
ResNet-50	✓	✓	**2.01**	08.1	10.9	0.51
ConvNeXt Tiny	✗	✗	2.13	07.9	10.8	0.53
ConvNeXt Tiny	✗	✓	2.07	08.0	11.0	0.49
ConvNeXt Tiny	✓	✗	2.10	**07.8**	**10.6**	0.51
ConvNeXt Tiny	✓	✓	2.18	08.1	11.1	0.52

Table 4.5: From left to right: (1) Backbone used, (2) whether the GNN decoder SpiralNet++ is used, (3) whether the Vertex-Edge loss is used, (4) time per image for landmarks estimation, fitting optimization, and overall time. The timings are evaluated for NVIDIA GeForce GTX TITAN X.

				Time	
Backbone	GNN Decoder	VE Loss	2D (ms/I)	(s/I) 3D	Total (s/I)
ResNet-50	✗	✗	20	1.0	1.0
ResNet-50	✗	✓	21	0.9	0.9
ResNet-50	✓	✗	30	1.0	1.0
ResNet-50	✓	✓	21	1.0	1.0
ConvNeXt Tiny	✗	✗	15	0.9	0.9
ConvNeXt Tiny	✗	✓	15	1.0	1.0
ConvNeXt Tiny	✓	✗	19	0.9	0.9
ConvNeXt Tiny	✓	✓	15	1.0	1.0

In Fig. 4.3 we show two example images from the test set for a qualitative comparison of the different landmark subsets. The first image contains a face with pronounced expression in almost frontal position. The second image contains a neutral expression in a more challenging pose. All subsets are quite capable of representing the expression of the face while progressively struggle at keeping head alignment. These results are conforming to the numeric results of Tab. 4.6. In both Fig. 4.2 and Fig. 4.3 we can get a sense of the extensive variety of DAD-3DHeads [77] dataset and note how challenging it would be to synthesize such realistic and diverse examples.

In Fig. 4.4 we show comparisons of our method with DECA [27], DAD-3DNet [77] on in

Table 4.6: From left to right: (1) Subset used, (2) Mean vertex error over face and head vertices, (3) Time of overall estimation per image. The timings are evaluated for NVIDIA GeForce GTX TITAN X.

	MVE (mm) ↓		
Subset	Face	Head	Time (s/I)
1609	07.8	10.6	0.9
805	**07.5**	**10.4**	1.0
403	07.5	10.6	1.1
202	07.6	10.8	1.1
101	07.8	10.8	1.2

Table 4.7: From left to right: (1) Method used, (2) Mean vertex error over face and head vertices, (3) Time of overall estimation per image. The timings are evaluated for NVIDIA GeForce GTX TITAN X.

	MVE (mm) ↓		
Method	Face	Head	Time (s/I)
DAD-3DNet [77]	11.8	18.2	0.6
DECA [27]	11.3	16.0	0.1
ours	**07.5**	**10.4**	1.0

the wild images. Our method achieves visual superior performance.

4.5 Implementation Details

We train all our landmarks prediction models using the training set of DAD-3DHeads [77] for 100 epochs, using Adam [52] as the optimizer, fixed learning rate of $1e^{-4}$ and batch size of 32. We performed data augmentation by randomly flipping vertically, horizontally, applying random scale and color alteration on the input images. All images are cropped to $256 \times 256 \times 3$. Since we did not have access to the test set ground truths we split the validation set in half and use one half as validation set and the other half as test set, they contain 2156 images each. To facilitate training, the landmarks are moved to $[-0.5, 0.5]$ interval.

Theseus setup: We use Levenberg-Marquardt (LM) [62, 76] to solve the rigid and non-rigid fitting problems. For both problems, we run LM for 20 iterations. For speed and storage efficiency we make use of FuncTorch functionalities for the jacobians of the objectives. These are available through Theseus and made automatic optimization feasible, avoiding manual computations for the jacobians. Finally, we use 100 parameters for shape and expression of the FLAME model and add L_2 regularizers to the objective for them.

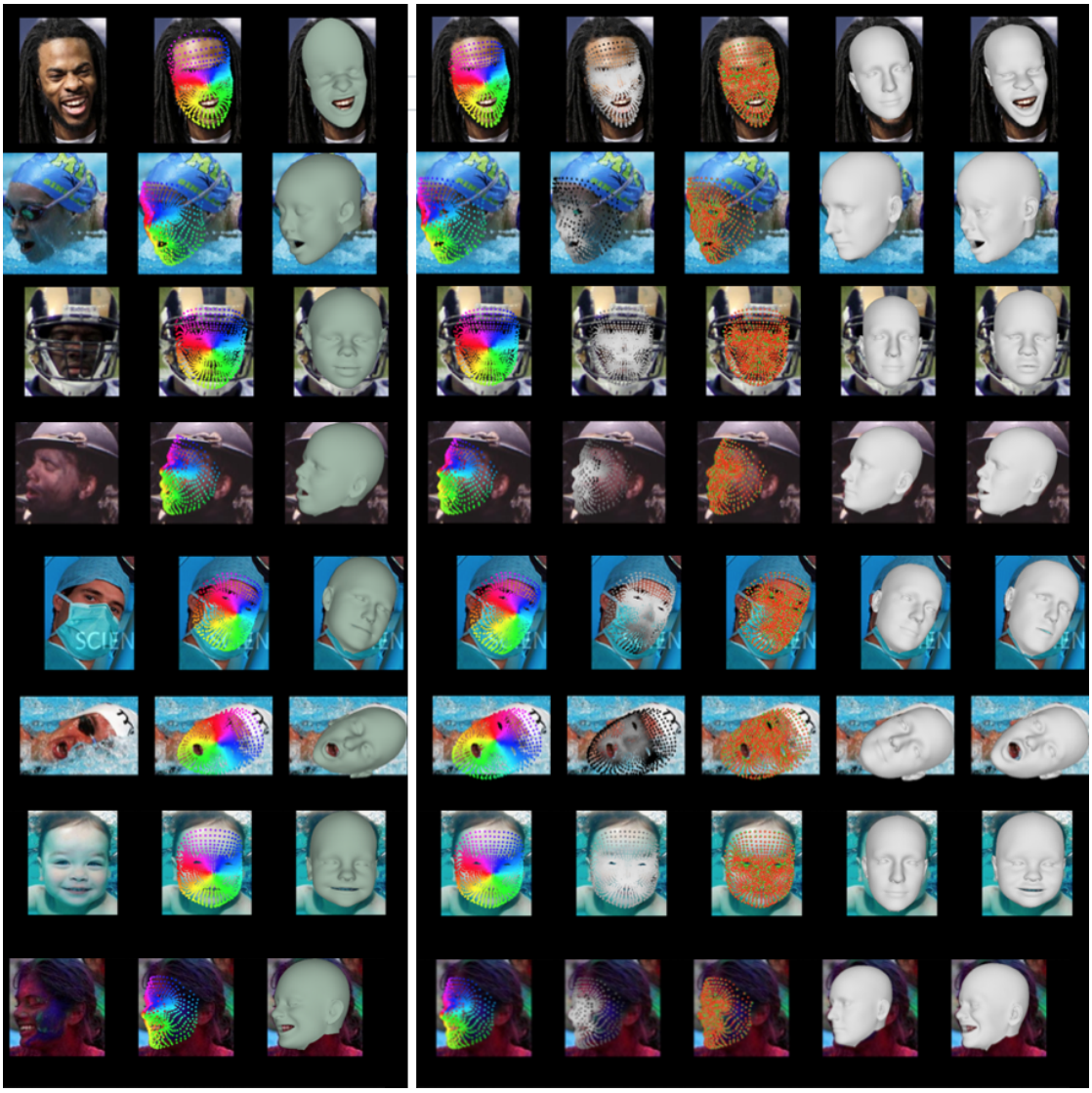

Figure 4.2: The left column contains the input image with pseudo 2D and 3D ground truth. In the right column, from left to right: (1) Estimated landmarks, (2) confidences brighter (light gray) means more confident, (3) re-projected landmarks on top of estimated landmarks, (4) rigid fitting, (5) non-rigid fitting. The network is able to estimate meaningful confidence values. For some images our reconstruction is perhaps more similar to the input image for eyes silhouette, shape, orientation or expression than the pseudo-ground truth. We used a ConvNeXt backbone, SpiralNet++ decoder, ResLL Exp loss with 1609 landmarks subset for this results.

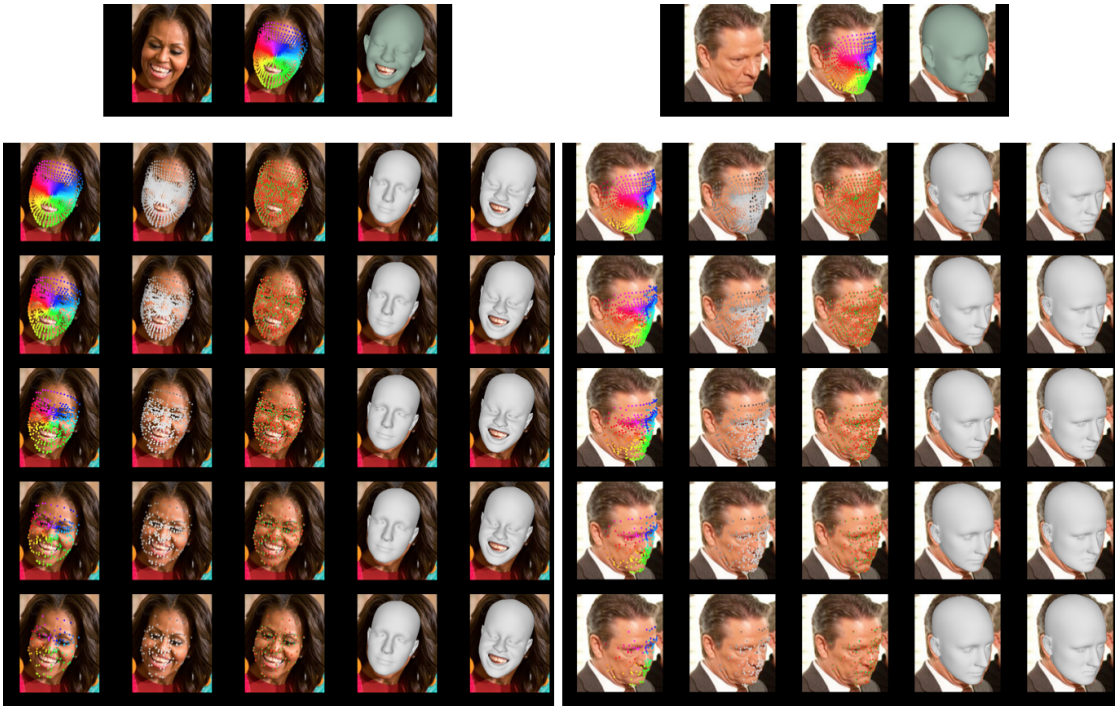

Figure 4.3: On the left, an example for an image with an expressive face in an almost frontal position. On the right an example for an image with a more challenging head pose. The first row corresponds to the pseudo-ground truth. Then, in order from top to bottom, different subsets $(1609, 805, 403, 202, 101)$ of landmarks used for fitting. The different subsets are quite able to capture the expressive frontal face but struggle progressively in head pose for the more challenging case (by looking at the ear of the person in the right image we can see the reconstruction progressively detaching from the head in the image). We used a ConvNeXt backbone, SpiralNet++ decoder, ResLL Exp loss with the above set of landmarks for these results.

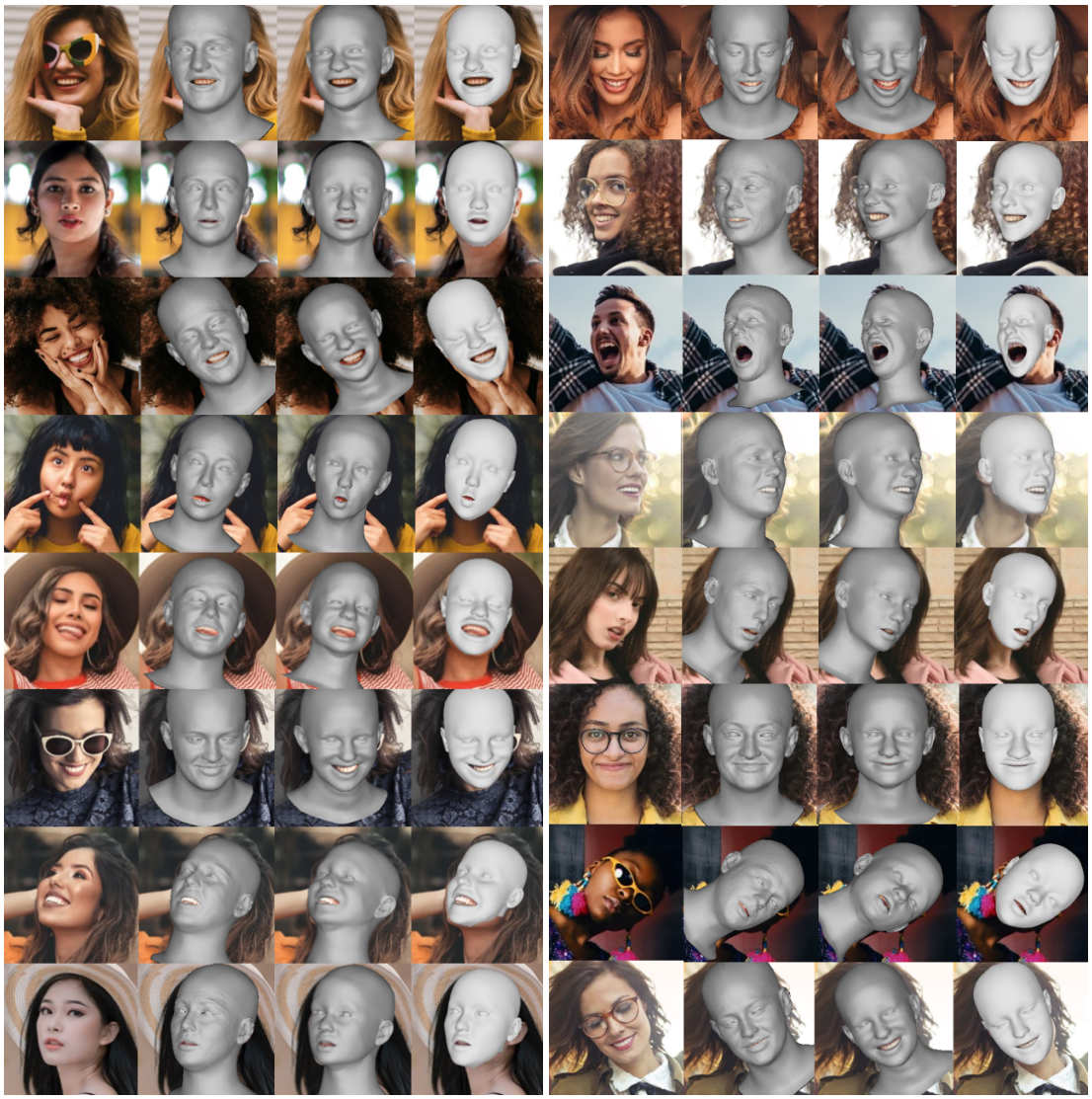

Figure 4.4: (1) RGB image, (2) DECA [27], (3) DAD-3DNet [77], (4) ours (head only). Our proposed fitting method produces head reconstructions with equal or better accuracy compared to the other methods. In addition, our reconstructions better align with the image evidence, thanks to the iterative fitting process. We used a ConvNeXt backbone, SpiralNet++ decoder, ResLL Exp loss and 805 landmarks subset for these results.

5. Conclusion

We propose a method that reconstructs a 3D face from monocular RGB images that is on par with state-of-the-art estimation techniques while being simpler. Our method surpasses DECA's [27] and DAD-3DNet's [77] reconstruction accuracy on DAD-3DHeads dataset. Note that DECA has been trained on 2 million images from different datasets and DAD-3DNet has been trained on the same dataset that we use of $\sim 44k$ images. Qualitative comparisons show robustness of our method under different poses, occlusions, and illumination conditions. In addition, expressions can be captured in finer detail compared to the other methods. The baseline methods use more powerful techniques in order to obtain the reconstructions. For the coarse reconstruction DECA estimates albedo, FLAME [65], camera and lightning parameters from multiple images of the same person. It uses multiple losses, namely a photometric loss between the input and the synthesized image, a loss on the parameters, and one on sparse landmarks. In addition to the image formation loss, it introduces a loss that disentangles static from dynamic details of person, in order to refine the coarse estimation. Note that the facial expression branch of DECA could also be used on top of our estimated FLAME parameters. DAD-3DNet combines learned landmark heatmaps and features to regress both sparse landmarks and FLAME parameters. This combination of 2D and 3D information should in theory give more robustness to the reconstructions compared to only learning landmarks but surprisingly just dense landmarks fitting works better. We find that adding uncertainty to the predictions by modeling the landmarks as random variables results in a more robust direct regression pipeline that gives flexibility during the fitting phase. It allows to focus on certain points and obtain an overall good reconstruction of the face even when there are occlusions or the labels are unprecise. Finally, in this work, we introduce a competitive alternative to the method of Wood et al. [114] and avoid the complexity required to generate synthetic data or the assumptions that need to be taken for analysis by synthesis approaches such as DECA.

Limitations: By qualitative analysis of the results, the main struggle of our reconstructions is head alignment. Moreover, our predictor is not able to widely spread the confidences over the $[0, 1]$ domain and tends to be too confident. The other main issue is the time required for the fitting optimization due to the poor initialization of the model parameters, which makes our method 10x slower than DECA and 2x slower than DAD-3DNet at the moment.

Future work: The current camera approximation does not take into consideration any information from the original input image dimensions and it may have a detrimental impact on the optimization of the model. Using image-conditioned intrinsics [54] should help improve the optimization results. Moreover, we only use the landmarks that cover the face, which might lead to inaccurate head shape and orientation estimates. Predicting landmarks for the ears could improve head alignment. Thus, further experiments are needed to identify the optimal subset of the landmarks to use. We could also completely discard some bad landmarks by placing a threshold on the confidences or could add importance weights on the loss for the landmarks that the network struggles to learn. We have noticed that on average the network is better able at learning the central part of the face than the outer region. This is probably due to the fact

that occlusions mostly happen on outer regions. It would also be interesting to experiment on how many pseudo-labels are required in order to learn a specific dense set of landmarks. This could be very useful given the difficulty in annotating dense datasets. An end-to-end network that predicts both landmarks and FLAME parameters would improve on the reconstructions and specifically on the initialization of the fitting pipeline. The network would predict landmarks, confidences and FLAME parameters. Since the optimization layer that we use for fitting the model is differentiable we can use it inside the network and place a loss on the estimated vertices using the 3D pseudo labels as ground truths in addition to the loss that we use for 2D. By doing this, 2D and 3D prediction would supervise and correct each other. Even though or network is able to predict confidences this pipeline could give more robust estimations for them widening the spectrum of estimated values and therefore solving the overconfidence problem. We could also greatly reduce inference time since the FLAME output of the network would in theory already be close to the optimum. Our method is body part agnostic, thus inference on other body parts , such as the hands, can easily be integrated into our pipeline.

6. Acknowledgements

I would like to thank my advisors, Dr. Gurkirt Singh and Vasileios Choutas for giving me the chance to work on this fascinating project. They are very competent in the field and motivated me during the work, while teaching how to properly handle these kind of projects. If I will continue to work in the field of Computer Vision it is going to be partly because of them. I especially thank Vasileios Choutas for the extensive help in editing this thesis and the care and directions he has provided during the months that led to this work.

Bibliography

[1] R Avenash and P Viswanath. Semantic Segmentation of Satellite Images using a Modified CNN with Hard-Swish Activation Function. In *VISIGRAPP (4: VISAPP)*, pages 413–420, 2019. 13

[2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 13

[3] Peter N Belhumeur, David W Jacobs, David J Kriegman, and Neeraj Kumar. Localizing Parts of Faces Using a Consensus of Exemplars. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 35(12):2930–2940, 2013. 3, 5

[4] Chandrasekhar Bhagavatula, Chenchen Zhu, Khoa Luu, and Marios Savvides. Faster Than Real-time Facial Alignment: A 3D Spatial Transformer Network Approach in Unconstrained Poses. In *Computer Vision and Pattern Recognition (CVPR)*, pages 3980–3989, 2017. 5

[5] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3D faces. In *ACM Transactions on Graphics (ToG), (Proc. SIGGRAPH)*, pages 187–194, 1999. 3, 5, 7

[6] Michael M. Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric Deep Learning: Going beyond Euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017. 13, 14

[7] Adrian Bulat, Enrique Sanchez, and Georgios Tzimiropoulos. Subpixel heatmap regression for facial landmark localization. In *British Machine Vision Conference (BMVC)*, 2021. 3, 5

[8] Adrian Bulat and Georgios Tzimiropoulos. How far are we from solving the 2D & 3D face alignment problem? (and a dataset of 230,000 3D facial landmarks). In *International Conference on Computer Vision (ICCV)*, pages 1021–1030, 2017. 3, 5

[9] Egor Burkov, Igor Pasechnik, Artur Grigorev, and Victor Lempitsky. Neural head reenactment with latent pose descriptors. In *Computer Vision and Pattern Recognition (CVPR)*, pages 13786–13795, June 2020. 3

[10] Chen Cao, Yanlin Weng, Shun Zhou, Yiying Tong, and Kun Zhou. FaceWarehouse: A 3D facial expression database for visual computing. *IEEE Transactions on Visualization and Computer Graphics*, 20(3):413–425, 2013. 4, 5

[11] Xuan Cao, Zhang Chen, Anpei Chen, Xin Chen, Shiying Li, and Jingyi Yu. Sparse photometric 3D face reconstruction guided by morphable models. In *Computer Vision and Pattern Recognition (CVPR)*, pages 4635–4644, 2018. 5

[12] Xudong Cao, Yichen Wei, Fang Wen, and Jian Sun. Face alignment by Explicit Shape Regression. *International Journal of Computer Vision (IJCV)*, 107(2):177–190, 2014. 5

[13] Prashanth Chandran, Derek Bradley, Markus Gross, and Thabo Beeler. Attention-Driven Cropping for Very High Resolution Facial Landmark Detection. In *Computer Vision and Pattern Recognition (CVPR)*, pages 5861–5870, 2020. 5

[14] Bindita Chaudhuri, Noranart Vesdapunt, Linda Shapiro, and Baoyuan Wang. Personalized Face Modeling for Improved Face Reconstruction and Motion Retargeting. In *European Conference on Computer Vision (ECCV)*, pages 142–160. Springer, 2020. 3, 6

[15] Bindita Chaudhuri, Noranart Vesdapunt, and Baoyuan Wang. Joint Face Detection and Facial Motion Retargeting for Multiple Faces. In *Computer Vision and Pattern Recognition (CVPR)*, pages 9719–9728, 2019. 3, 6

[16] Shiyang Cheng, Irene Kotsia, Maja Pantic, and Stefanos Zafeiriou. 4DFAB: A Large Scale 4D Database for Facial Expression Analysis and Biometric Applications. In *Computer Vision and Pattern Recognition (CVPR)*, pages 5117–5126, 2018. 4, 5

[17] Nikolai Chinaev, Alexander Chigorin, and Ivan Laptev. MobileFace: 3D face reconstruction with efficient CNN regression. In *European Conference on Computer Vision Workshops (ECCVw)*, pages 0–0, 2018. 6

[18] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs). In *International Conference on Learning Representations (ICLR)*, 2016. 15

[19] Darren Cosker, Eva Krumhuber, and Adrian Hilton. A FACS valid 3D dynamic action unit database with applications to 3D dynamic morphable facial modeling. In *International Conference on Computer Vision (ICCV)*, pages 2296–2303. IEEE, 2011. 4, 5, 10

[20] Radek Danecek, Michael J. Black, and Timo Bolkart. EMOCA: Emotion driven monocular face capture and animation. In *Computer Vision and Pattern Recognition (CVPR)*, 2022. 4, 6

[21] Jiankang Deng, George Trigeorgis, Yuxiang Zhou, and Stefanos Zafeiriou. Joint Multi-view Face Alignment in the Wild. *IEEE Transactions on Image Processing*, 28(7):3636–3648, 2019. 5

[22] Yu Deng, Jiaolong Yang, Sicheng Xu, Dong Chen, Yunde Jia, and Xin Tong. Accurate 3D Face Reconstruction with Weakly-Supervised Learning: From Single Image to Image Set. In *Computer Vision and Pattern Recognition Workshops (CVPRw)*, pages 0–0, 2019. 4, 6

[23] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using Real NVP. In *International Conference on Learning Representations (ICLR)*, 2017. 16

[24] Yuan Dong and Yue Wu. Adaptive cascade deep convolutional neural networks for face alignment. *Computer standards & interfaces*, 42:105–112, 2015. 5

[25] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations (ICLR)*, 2021. 12

[26] Bernhard Egger, William A. P. Smith, Ayush Tewari, Stefanie Wuhrer, Michael Zollhoefer, Thabo Beeler, Florian Bernard, Timo Bolkart, Adam Kortylewski, Sami Romdhani, Christian Theobalt, Volker Blanz, and Thomas Vetter. 3D Morphable Face Models - Past, Present and Future. *Transactions on Graphics (TOG)*, 39(5), August 2020. 3, 5, 7

[27] Yao Feng, Haiwen Feng, Michael J. Black, and Timo Bolkart. Learning an animatable detailed 3D face model from in-the-wild images. *ACM Transactions on Graphics (ToG), (Proc. SIGGRAPH)*, 40(4):88:1–88:13, 2021. 4, 6, 27, 28, 29, 32, 33

[28] Leonardo Galteri, Claudio Ferrari, Giuseppe Lisanti, Stefano Berretti, and Alberto Del Bimbo. Coarse-to-fine 3D face reconstruction. In *Computer Vision and Pattern Recognition Workshops (CVPRw)*, pages 25–31, 2019. 6

[29] Leonardo Galteri, Claudio Ferrari, Giuseppe Lisanti, Stefano Berretti, and Alberto Del Bimbo. Deep 3D morphable model refinement via progressive growing of conditional Generative Adversarial Networks. *Computer Vision and Image Understanding (CVIU)*, 185:31–42, 2019. 6

[30] Michael Garland and Paul S Heckbert. Surface simplification using quadric error metrics. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 209–216, 1997. 14

[31] Kyle Genova, Forrester Cole, Aaron Maschinot, Aaron Sarna, Daniel Vlasic, and William T Freeman. Unsupervised Training for 3D Morphable Model Regression. In *Computer Vision and Pattern Recognition (CVPR)*, pages 8377–8386, 2018. 6

[32] Thomas Gerig, Andreas Morel-Forster, Clemens Blumer, Bernhard Egger, Marcel Luthi, Sandro Schoenborn, and Thomas Vetter. Morphable Face Models - An Open Framework. In *International Conference on Automatic Face & Gesture Recognition (FG)*, pages 75–82, 2018. 3, 5, 7

[33] Nima Ghorbani and Michael J. Black. SOMA: Solving Optical Marker-Based MoCap Automatically. In *International Conference on Computer Vision (ICCV)*, pages 11117–11126, October 2021. 3

[34] Partha Ghosh, Pravir Singh Gupta, Roy Uziel, Anurag Ranjan, Michael J. Black, and Timo Bolkart. GIF: Generative interpretable faces. In *International Conference on 3D Vision (3DV)*, volume 1, pages 868–878, November 2020. 3

[35] Shunwang Gong, Lei Chen, Michael Bronstein, and Stefanos Zafeiriou. Spiralnet++: A fast and highly efficient mesh convolution operator. In *International Conference on Computer Vision Workshops (ICCVw)*, 2019. 4, 14, 15, 25

[36] Rıza Alp Güler, Natalia Neverova, and Iasonas Kokkinos. DensePose: Dense Human Pose Estimation in the Wild. In *Computer Vision and Pattern Recognition (CVPR)*, pages 7297–7306, June 2018. 3

[37] Jianzhu Guo, Xiangyu Zhu, Yang Yang, Fan Yang, Zhen Lei, and Stan Z Li. Towards Fast, Accurate and Stable 3D Dense Face Alignment. In *European Conference on Computer Vision (ECCV)*, pages 152–168. Springer, 2020. 6

[38] Vladimir Guzov, Aymen Mir, Torsten Sattler, and Gerard Pons-Moll. Human POSEitioning System (HPS): 3D Human Pose Estimation and Self-localization in Large Scenes from Body-Mounted Sensors . In *Computer Vision and Pattern Recognition (CVPR)*, pages 4318–4329, jun 2021. 3

[39] Dan Witzner Hansen and Qiang Ji. In the Eye of the Beholder: A Survey of Models for Eyes and Gaze. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 32(3):478–500, 2009. 5

[40] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 4, 5, 6, 11, 18, 24

[41] Dan Hendrycks and Kevin Gimpel. Gaussian Error Linear Units (GELUs). *arXiv preprint arXiv:1606.08415*, 2016. 12

[42] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, Quoc V. Le, and Hartwig Adam. Searching for MobileNetV3. In *International Conference on Computer Vision (ICCV)*, pages 1314–1324, 2019. 13

[43] Andrew Howard, Andrey Zhmoginov, Liang-Chieh Chen, Mark Sandler, and Menglong Zhu. Inverted Residuals and Linear Bottlenecks: Mobile Networks for Classification, Detection and Segmentation. In *Computer Vision and Pattern Recognition (CVPR)*, pages 4510–4520, 2018. 13

[44] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv preprint arXiv:1704.04861*, 2017. 13

[45] Chun-Hao P. Huang, Hongwei Yi, Markus Höschle, Matvey Safroshkin, Tsvetelina Alexiadis, Senya Polikovsky, Daniel Scharstein, and Michael J. Black. Capturing and Inferring Dense Full-Body Human-Scene Contact. In *Computer Vision and Pattern Recognition (CVPR)*, pages 13274–13285, June 2022. 3

[46] Sergey Ioffe. Batch Renormalization: Towards Reducing Minibatch Dependence in Batch-Normalized Models. *Advances in neural information processing systems*, 30, 2017. 13

[47] Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In Francis Bach and David Blei, editors, *International Conference on Machine Learning (ICML)*, volume 37 of *Proceedings of Machine Learning Research*, pages 448–456, Lille, France, 07–09 Jul 2015. PMLR. 13

[48] Amin Jourabloo and Xiaoming Liu. Large-pose Face Alignment via CNN-based Dense 3D Model Fitting. In *Computer Vision and Pattern Recognition (CVPR)*, pages 4188–4196, 2016. 5

[49] Amin Jourabloo, Mao Ye, Xiaoming Liu, and Liu Ren. Pose-Invariant Face Alignment with a Single CNN. In *International Conference on Computer Vision (ICCV)*, pages 3200–3209, 2017. 5

[50] Vahid Kazemi and Josephine Sullivan. One millisecond face alignment with an ensemble of regression trees. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1867–1874, 2014. 5

[51] Ira Kemelmacher-Shlizerman and Ronen Basri. 3D Face Reconstruction from a Single Image Using a Single Reference Face Shape. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 33(2):394–405, 2010. 5

[52] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015. 25, 29

[53] Imry Kissos, Lior Fritz, Matan Goldman, Omer Meir, Eduard Oks, and Mark Kliger. Beyond Weak Perspective for Monocular 3D Human Pose Estimation. *CoRR*, abs/2009.06549, 2020. 22

[54] Muhammed Kocabas, Chun-Hao P. Huang, Joachim Tesch, Lea Müller, Otmar Hilliges, and Michael J. Black. SPEC: Seeing people in the wild with an estimated camera. In *International Conference on Computer Vision (ICCV)*, pages 11035–11045, 2021. 33

[55] Martin Koestinger, Paul Wohlhart, Peter M Roth, and Horst Bischof. Annotated facial landmarks in the wild: A large-scale, real-world database for facial landmark localization. In *International Conference on Computer Vision Workshops (ICCVw)*, pages 2144–2151. IEEE, 2011. 3, 5

[56] Marek Kowalski, Stephan J. Garbin, Virginia Estellers, Tadas Baltrušaitis, Matthew Johnson, and Jamie Shotton. CONFIG: Controllable Neural Face Image Generation. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *European Conference on Computer Vision (ECCV)*, pages 299–315, Cham, 2020. Springer International Publishing. 3

[57] Marek Kowalski, Jacek Naruniec, and Tomasz Trzcinski. Deep Alignment Network: A Convolutional Neural Network for Robust Face Alignment. In *Computer Vision and Pattern Recognition Workshops (CVPRw)*, pages 88–97, 2017. 5

[58] Vuong Le, Jonathan Brandt, Zhe Lin, Lubomir Bourdev, and Thomas S Huang. Interactive facial feature localization. In *European Conference on Computer Vision (ECCV)*, pages 679–692. Springer, 2012. 3, 5

[59] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989. 11

[60] Gun-Hee Lee and Seong-Whan Lee. Uncertainty-Aware Mesh Decoder for High Fidelity 3D Face Reconstruction. In *Computer Vision and Pattern Recognition (CVPR)*, pages 6100–6109, 2020. 6

[61] Minsik Lee and Chong-Ho Choi. Fast facial shape recovery from a single image with general, unknown lighting by using tensor representation. *Pattern Recognition (PR)*, 44(7):1487–1496, 2011. 5

[62] Kenneth Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly of applied mathematics*, 2(2):164–168, 1944. 21, 29

[63] J. P. Lewis, Matt Cordner, and Nickson Fong. Pose Space Deformation: A Unified Approach to Shape Interpolation and Skeleton-Driven Deformation. In *ACM Transactions on Graphics (ToG), (Proc. SIGGRAPH)*, pages 165—172, USA, 2000. 9

[64] Jiefeng Li, Siyuan Bian, Ailing Zeng, Can Wang, Bo Pang, Wentao Liu, and Cewu Lu. Human Pose Regression with Residual Log-likelihood Estimation. In *International Conference on Computer Vision (ICCV)*, 2021. 4, 15, 16, 17, 18, 19, 24, 25

[65] Tianye Li, Timo Bolkart, Michael. J. Black, Hao Li, and Javier Romero. Learning a model of facial shape and expression from 4D scans. *ACM Transactions on Graphics (ToG), (Proc. SIGGRAPH Asia)*, 36(6):194:1–194:17, 2017. 3, 5, 6, 7, 8, 9, 10, 14, 18, 22, 24, 27, 33

[66] Yue Li, Liqian Ma, Haoqiang Fan, and Kenny Mitchell. Feature-preserving detailed 3D face reconstruction from a single image. In *Proceedings of the 15th ACM SIGGRAPH European Conference on Visual Media Production*, pages 1–9, 2018. 5

[67] Zhujin Liang, Shengyong Ding, and Liang Lin. Unconstrained facial landmark localization with backbone-branches fully-convolutional networks. *arXiv preprint arXiv:1507.03409*, 2015. 3, 5

[68] Jiangke Lin, Yi Yuan, Tianjia Shao, and Kun Zhou. Towards high-fidelity 3D face reconstruction from in-the-wild images using graph convolutional networks. In *Computer Vision and Pattern Recognition (CVPR)*, pages 5891–5900, 2020. 6

[69] Hao Liu, Jiwen Lu, Jianjiang Feng, and Jie Zhou. Two-Stream Transformer Networks for Video-Based Face Alignment. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 40(11):2546–2554, 2017. 5

[70] Peng Liu, Yao Yu, Yu Zhou, and Sidan Du. Single view 3D face reconstruction with landmark updating. In *2019 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, pages 403–408. IEEE, 2019. 5

[71] Yaojie Liu, Amin Jourabloo, William Ren, and Xiaoming Liu. Dense face alignment. In *International Conference on Computer Vision Workshops (ICCVw)*, pages 1619–1628, 2017. 5

[72] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. In *International Conference on Computer Vision (ICCV)*, pages 10012–10022, 2021. 4, 12

[73] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A ConvNet for the 2020s. In *Computer Vision and Pattern Recognition (CVPR)*, pages 11976–11986, 2022. 4, 12, 13, 25

[74] Jiangjing Lv, Xiaohu Shao, Junliang Xing, Cheng Cheng, and Xi Zhou. A Deep Regression Architecture with Two-Stage Re-initialization for High Performance Facial Landmark Detection. In *Computer Vision and Pattern Recognition (CVPR)*, pages 3317–3326, 2017. 5

[75] Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. AMASS: Archive of motion capture as surface shapes. In *International Conference on Computer Vision (ICCV)*, pages 5442–5451. IEEE, October 2019. 3

[76] Donald W Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the society for Industrial and Applied Mathematics*, 11(2):431–441, 1963. 21, 29

[77] Tetiana Martyniuk, Orest Kupyn, Yana Kurlyak, Igor Krashenyi, Jiří Matas, and Viktoriia Sharmanska. DAD-3DHeads: A Large-Scale Dense, Accurate and Diverse Dataset for 3D Head Alignment From a Single Image. In *Computer Vision and Pattern Recognition (CVPR)*, pages 20942–20952, June 2022. 4, 5, 6, 10, 11, 27, 28, 29, 32, 33

[78] Araceli Morales, Gemma Piella, and Federico M. Sukno. Survey on 3D face reconstruction from uncalibrated images. *CoRR*, abs/2011.05740, 2020. 5

[79] Vinod Nair and Geoffrey E Hinton. Rectified Linear Units Improve Restricted Boltzmann Machines. In *International Conference on Machine Learning (ICML)*, 2010. 12

[80] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked Hourglass Networks for Human Pose Estimation. In *European Conference on Computer Vision (ECCV)*, pages 483–499. Springer, 2016. 3, 5

[81] Maja Pantic and Leon J. M. Rothkrantz. Automatic analysis of facial expressions: The state of the art. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 22(12):1424–1445, 2000. 3, 5

[82] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Conference on Neural Information Processing Systems (NeurIPS)*, volume 32, 2019. 4, 21

[83] Pascal Paysan, Reinhard Knothe, Brian Amberg, Sami Romdhani, and Thomas Vetter. A 3D face model for pose and illumination invariant face recognition. In *IEEE International Conference on Advanced Video and Signal Based Surveillance*, pages 296–301, 2009. 3, 5, 7

[84] Luis Pineda, Taosha Fan, Maurizio Monge, Shobha Venkataraman, Paloma Sodhi, Ricky TQ Chen, Joseph Ortiz, Daniel DeTone, Austin Wang, Stuart Anderson, Jing Dong, Brandon Amos, and Mustafa Mukadam. Theseus: A Library for Differentiable Nonlinear Optimization. *Conference on Neural Information Processing Systems (NeurIPS)*, 2022. 4, 21, 22

[85] Chengchao Qu, Eduardo Monari, Tobias Schuchert, and Jürgen Beyerer21. Adaptive Contour Fitting for Pose-Invariant 3D Face Shape Reconstruction. In *British Machine Vision Conference (BMVC)*, 2015. 5

[86] Prajit Ramachandran, Barret Zoph, and Quoc V Le. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017. 13

[87] Ravi Ramamoorthi. Modeling illumination variation with spherical harmonics. *Face Processing: Advanced Modeling Methods*, pages 385–424, 2006. 4

[88] Anurag Ranjan, Timo Bolkart, Soubhik Sanyal, and Michael J Black. Generating 3D faces using convolutional mesh autoencoders. In *European Conference on Computer Vision (ECCV)*, pages 704–720, 2018. 14

[89] Rajeev Ranjan, Vishal M Patel, and Rama Chellappa. Hyperface: A deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 41(1):121–135, 2017. 5

[90] Shaoqing Ren, Xudong Cao, Yichen Wei, and Jian Sun. Face Alignment at 3000 FPS via Regressing Local Binary Features. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1685–1692, 2014. 5

[91] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International Conference on Machine Learning (ICML)*, pages 1530–1538. PMLR, 2015. 16

[92] Elad Richardson, Matan Sela, and Ron Kimmel. 3D face reconstruction by learning from synthetic data. In *International Conference on 3D Vision (3DV)*, pages 460–469. IEEE, 2016. 6

[93] Kathleen M. Robinette, Sherri Blackwell, Hein Daanen, Mark Boehmer, Scott Fleming, Tina Brill, David Hoeferlin, and Dennis Burnsides. Civilian American and European Surface Anthropometry Resource (CAESAR) final report. Technical Report AFRL-HE-WP-TR-2002-0169, US Air Force Research Laboratory, 2002. 10

[94] Gemma Rotger Moll, Francesc Moreno-Noguer, Felipe Lumbreras, and Antonio Agudo Martínez. Detailed 3D face reconstruction from a single RGB image. *Journal of WSCG (Plzen, Print)*, 27(2):103–112, 2019. 5

[95] Christos Sagonas, Georgios Tzimiropoulos, Stefanos Zafeiriou, and Maja Pantic. 300 Faces in-the-Wild Challenge: The First Facial Landmark Localization Challenge. In *International Conference on Computer Vision Workshops (ICCVw)*, pages 397–403, 2013. 3, 5

[96] Soubhik Sanyal, Timo Bolkart, Haiwen Feng, and Michael J. Black. Learning to regress 3D face shape and expression from an image without 3D supervision. In *Computer Vision and Pattern Recognition (CVPR)*, pages 7763–7772, 2019. 6

[97] Nedko Savov, Minh Ngô, Sezer Karaoglu, Hamdi Dibeklioglu, and Theo Gevers. Pose and expression robust age estimation via 3D face reconstruction from a single image. In *International Conference on Computer Vision Workshops (ICCVw)*, pages 0–0, 2019. 4, 6

[98] Jie Shen, Stefanos Zafeiriou, Grigoris G Chrysos, Jean Kossaifi, Georgios Tzimiropoulos, and Maja Pantic. The First Facial Landmark Tracking in-the-Wild Challenge: Benchmark and Results. In *International Conference on Computer Vision Workshops (ICCVw)*, pages 50–58, 2015. 3, 5

[99] Fuhao Shi, Hsiang-Tao Wu, Xin Tong, and Jinxiang Chai. Automatic Acquisition of High-fidelity Facial Performances Using Monocular Videos. *Transactions on Graphics (TOG)*, 33(6):1–13, 2014. 5

[100] Mingli Song, Dacheng Tao, Xiaoqin Huang, Chun Chen, and Jiajun Bu. Three-Dimensional Face Reconstruction From a Single Image by a Coupled RBF Network. *IEEE Transactions on Image Processing*, 21(5):2887–2897, 2012. 6

[101] Yi Sun, Xiaogang Wang, and Xiaoou Tang. Deep Convolutional Network Cascade for Facial Point Detection. In *Computer Vision and Pattern Recognition (CVPR)*, pages 3476–3483, 2013. 5

[102] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. MnasNet: Platform-Aware Neural Architecture Search for Mobile. In *Computer Vision and Pattern Recognition (CVPR)*, pages 2820–2828, 2019. 13

[103] Ayush Tewari, Florian Bernard, Pablo Garrido, Gaurav Bharaj, Mohamed Elgharib, Hans-Peter Seidel, Patrick Pérez, Michael Zöllhofer, and Christian Theobalt. FML: Face model learning from videos. In *Computer Vision and Pattern Recognition (CVPR)*, pages 10812–10822, 2019. 4

[104] Ayush Tewari, Mohamed Elgharib, Gaurav Bharaj, Florian Bernard, Hans-Peter Seidel, Patrick Perez, Michael Zollhofer, and Christian Theobalt. StyleRig: Rigging StyleGAN for 3D Control Over Portrait Images. In *Computer Vision and Pattern Recognition (CVPR)*, pages 6142–6151, June 2020. 3

[105] Ayush Tewari, Michael Zollhöfer, Pablo Garrido, Florian Bernard, Hyeongwoo Kim, Patrick Pérez, and Christian Theobalt. Self-supervised Multi-level Face Model Learning for Monocular Reconstruction at over 250 Hz . In *Computer Vision and Pattern Recognition (CVPR)*, pages 2549–2559, 2018. 4, 6

[106] Ayush Tewari, Michael Zollhöfer, Hyeongwoo Kim, Pablo Garrido, Florian Bernard, Patrick Perez, and Christian Theobalt. MoFA: model-based deep convolutional face autoencoder for unsupervised monocular reconstruction. In *International Conference on Computer Vision (ICCV)*, pages 3735–3744, 2017. 4, 6

[107] Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christoph Bregler. Efficient Object Localization Using Convolutional Networks. In *Computer Vision and Pattern Recognition (CVPR)*, pages 648–656, 2015. 3, 5

[108] George Trigeorgis, Patrick Snape, Mihalis A Nicolaou, Epameinondas Antonakos, and Stefanos Zafeiriou. Mnemonic Descent Method: A recurrent process applied for end-to-end face alignment. In *Computer Vision and Pattern Recognition (CVPR)*, pages 4177–4187, 2016. 5

[109] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Conference on Neural Information Processing Systems (NeurIPS)*, volume 30. Curran Associates, Inc., 2017. 12

[110] Timo von Marcard, Roberto Henschel, Michael J. Black, Bodo Rosenhahn, and Gerard Pons-Moll. Recovering accurate 3D human pose in the wild using IMUs and a moving camera. In *European Conference on Computer Vision (ECCV)*, pages 614–631, 2018. 3

[111] Pengrui Wang, Yi Tian, Wujun Che, and Bo Xu. Efficient and accurate face shape reconstruction by fusion of multiple landmark databases. In *International Conference on Image Processing (ICIP)*, pages 335–339. IEEE, 2019. 6

[112] Qiulin Wang, Lu Zhang, and Bo Li. SAFA: Structure Aware Face Animation. In *International Conference on 3D Vision (3DV)*, pages 679–688, 2021. 3

[113] Erroll Wood, Tadas Baltrusaitis, Charlie Hewitt, Sebastian Dziadzio, Matthew Johnson, Virginia Estellers, Thomas J. Cashman, and Jamie Shotton. Fake It Till You Make It: Face Analysis in the Wild Using Synthetic Data Alone. In *International Conference on Computer Vision (ICCV)*, pages 3681–3691, 2021. 3, 4, 5, 6

[114] Erroll Wood, Tadas Baltrusaitis, Charlie Hewitt, Matthew Johnson, Jingjing Shen, Nikola Milosavljevic, Daniel Wilde, Stephan Garbin, Chirag Raman, Jamie Shotton, Toby Sharp, Ivan Stojiljkovic, Tom Cashman, and Julien Valentin. 3D face reconstruction with dense landmarks. In *European Conference on Computer Vision (ECCV)*, 2022. 4, 5, 6, 16, 24, 25, 33

[115] Yue Wu and Qiang Ji. Facial landmark detection: A literature survey. *International Journal of Computer Vision (IJCV)*, 127(2):115–142, 2019. 5

[116] Xuehan Xiong and Fernando De la Torre. Supervised Descent Method and Its Applications to Face Alignment. In *Computer Vision and Pattern Recognition (CVPR)*, pages 532–539, 2013. 5

[117] Haotian Yang, Hao Zhu, Yanru Wang, Mingkai Huang, Qiu Shen, Ruigang Yang, and Xun Cao. FaceScape: A large-scale high quality 3D face dataset and detailed riggable 3D face prediction. In *Computer Vision and Pattern Recognition (CVPR)*, pages 598–607, 2020. 4, 5

[118] Jing Yang, Qingshan Liu, and Kaihua Zhang. Stacked Hourglass Network for Robust Facial Landmark Localisation. In *Computer Vision and Pattern Recognition Workshops (CVPRw)*, pages 79–87, 2017. 3, 5

[119] Jae Shin Yoon, Takaaki Shiratori, Shoou-I Yu, and Hyun Soo Park. Self-Supervised Adaptation of High-Fidelity Face Models for Monocular Performance Tracking. In *Computer Vision and Pattern Recognition (CVPR)*, pages 4601–4609, 2019. 6

[120] Jae Shin Yoon, Zhixuan Yu, Jaesik Park, and Hyun Park. HUMBI: A Large Multiview Dataset of Human Body Expressions and Benchmark Challenge. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, pages 1–1, 2021. 3

[121] Gang Zhang, Hu Han, Shiguang Shan, Xingguang Song, and Xilin Chen. Face alignment across large pose via MT-CNN based 3D shape reconstruction. In *International Conference on Automatic Face & Gesture Recognition (FG)*, pages 210–217. IEEE, 2018. 6

[122] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE signal processing letters*, 23(10):1499–1503, 2016. 5

[123] Zhanpeng Zhang, Ping Luo, Chen Change Loy, and Xiaoou Tang. Facial landmark detection by deep multi-task learning. In *European Conference on Computer Vision (ECCV)*, pages 94–108. Springer, 2014. 5

[124] Erjin Zhou, Haoqiang Fan, Zhimin Cao, Yuning Jiang, and Qi Yin. Extensive Facial Landmark Localization with Coarse-to-Fine Convolutional Network Cascade. In *International Conference on Computer Vision Workshops (ICCVw)*, pages 386–391, 2013. 5

[125] Xiangyu Zhu, Zhen Lei, Xiaoming Liu, Hailin Shi, and Stan Z Li. Face Alignment Across Large Poses: A 3D Solution. In *Computer Vision and Pattern Recognition (CVPR)*, pages 146–155, 2016. 3, 4, 5, 6

[126] Xiangyu Zhu, Zhen Lei, Junjie Yan, Dong Yi, and Stan Z Li. High-fidelity pose and expression normalization for face recognition in the wild. In *Computer Vision and Pattern Recognition (CVPR)*, pages 787–796, 2015. 5