UKRAINIAN CATHOLIC UNIVERSITY

MASTER THESIS

3D Head Model Estimation from a Single Photo

Author:
Rostyslav ZATSERKOVNYI

Supervisor:
Orest KUPYN

A thesis submitted in fulfillment of the requirements
for the degree of Master of Science

in the

Department of Computer Sciences
Faculty of Applied Sciences

Lviv 2021

Declaration of Authorship

I, Rostyslav ZATSERKOVNYI, declare that this thesis titled, “3D Head Model Estimation from a Single Photo” and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

UKRAINIAN CATHOLIC UNIVERSITY

Faculty of Applied Sciences

Master of Science

3D Head Model Estimation from a Single Photo

by Rostyslav ZATSERKOVNYI

Abstract

Today, 3D human head models are widely used in fields such as computer vision, entertainment, healthcare, and biometrics. Since a high-quality scan of a human head is expensive and time-consuming to obtain, machine learning algorithms are used to estimate the shape and texture of a 3D model from a single "in-the-wild" photograph, often taken at extreme angles or with non-uniform illumination. However, as a full head texture cannot be trivially inferred from a single photograph due to self-occlusion, many only focus on modeling an incomplete and partially textured model of the human head.

This work proposes a machine learning pipeline that reconstructs a fully textured 3D head model from a single photograph. We collect a novel dataset of 99.3 thousand high-resolution human head textures created from synthetic celebrity photographs. To the best of our knowledge, this is the first UV texture dataset of a similar scale and fidelity. Using this dataset, we train a free-form inpainting GAN that learns to recreate full head textures from partially obscured projections of the input photograph.

Acknowledgements

I would like to thank my thesis advisor Orest Kupyn, whose ideas and advice was invaluable in the research project and who has provided the computational resources that allowed this project to succeed. I also thank Oleksii Molchanovskyi and the Ukrainian Catholic University for organizing an excellent Master's program, especially under challenging distance learning conditions. Finally, I would like to thank my coursemates for their continual support and for making the two years of this program an unforgettable experience.

Contents

List of Figures

List of Tables

List of Abbreviations

3DDFA	**3D** **D**ense **F**ace **A**lignment
3DMM	**3D** **M**orphable **M**odel
CNN	**C**onvolutional **N**eural **N**etwork
DNN	**D**eep **N**eural **N**etwork
GAN	**G**enerative **A**dversarial **N**etwork
LSFM	**L**arge-**S**cale **F**acial **M**odel
PAF	**P**ose-**A**daptive **F**eature
UV texture	refers to **U** and **V** axes in 2D space

Chapter 1

Introduction

1.1 Motivation

In recent years, computer models of human heads have found widespread use in many applications related to image processing. Among these uses are authentication methods based on 3D biometrics (Park and Jain, 2007), partial head prosthetics sculpted to match a patient's head shape (Guo et al., 2020), or realistic video game avatars representing the user as a stylized character (Lin, Yuan, and Zou, 2021).

For these and many other applications, a single two-dimensional photograph is often insufficient to capture the full range of information about a head. A picture does not contain information about a single dimension of the original 3D object, which needs to be computationally inferred. It is also heavily affected by variance in pose, expression or illumination. On the other hand, a three-dimensional model of a human head can represent it in a form invariant to these factors. Via 3D object manipulation, this model can be projected onto different angles, re-textured, illuminated, and, depending on its complexity, even fully animated. However, capturing a high-quality 3D scan requires both expensive scanning equipment and heavily calibrated working conditions - while this may be acceptable in fields such as medicine, setting up a 3D scanner is often a non-feasible approach.

In machine learning, head reconstruction algorithms aim to solve this problem, combining the simplicity of taking a single photograph and the benefits gained from a full 3D model. They accomplish this by recreating a three-dimensional model of the human head from one or several images. Some head reconstruction algorithms learn to map a photograph to a traditional 3D model: a polygon mesh alongside a set of textures. However, these algorithms most often use a statistical model known as a 3D Morphable Model (3DMM).

In essence, a 3DMM is both a data structure containing compressed, constrained information about a human head, and a method for constructing this data structure (Booth et al., 2018). A 3DMM is created from an extensive database of high-quality facial scans, using dimensionality reduction to compress them into a small but representative set of parameters. A single instance of a 3DMM is a parameter vector denoting face shape, texture, and sometimes optional parameters such as expression or illumination. A head reconstruction algorithm will be trained to infer this set of parameters from a 2D photograph. During training, some models use a set of ground truth 3D scans that are compared to the network's output. Without access to such scans, others opt for a more indirect approach. As an example, the 3DMM reconstructed by the algorithm may be projected to a 2D image at the same angle as the original "in-the-wild" photograph, then compared with the initial output through pixel loss.

However, many machine learning pipelines in the field only focus on reconstructing a partial version of the 3D model. Due to the issue of self-occlusion - that

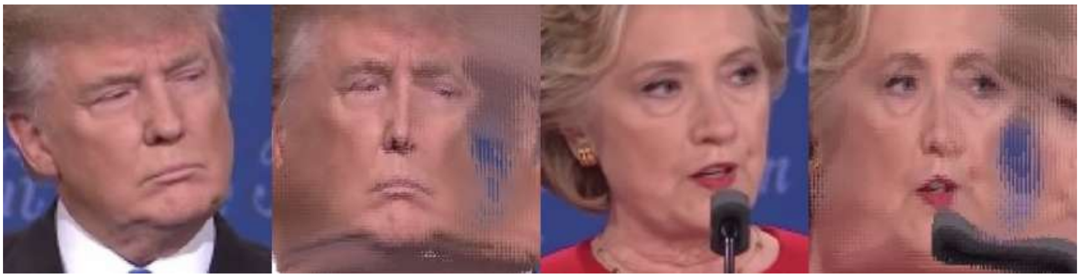

FIGURE 1.1: Face textures such as those in the 3DDFA pipeline (Zhu et al., 2019) appear notably distorted in occluded face regions.

is, the fact that a human head can never be fully displayed on a single photograph, as it is obscured by itself - such pipelines often output a partial texture limited by the angles visible on the photograph. While sufficient for some applications, heavy and naive interpolation is used to fill in the gaps on these textures, and the areas of the head not visible on the photograph often appear strange and unrealistic. In order to derive a full photorealistic head model, it is necessary to extend the head reconstruction pipeline with a novel step which, given a partial head texture, infers and inpaints its missing regions.

1.2 Goals

This thesis project focuses on extending a face reconstruction pipeline to recreate fully textured 3D head models. Briefly, it consists of two main parts:

1. Creating a novel dataset of facial UV textures. Due to the sensitive nature of biometric data, datasets containing high-resolution facial scans such as MeIn3D (Booth et al., 2018) are only available for medical research purposes; to the best of our knowledge, no publically accessible, anonymized datasets of this nature are currently available. Therefore, we introduce our own dataset based on high-quality synthetic data. Using the EigenGAN generative CNN (He, Kan, and Shan, 2021), we create several photographs of a single fictional person at different horizontal angles. We then convert these into high-quality yet incomplete head textures using a head reconstruction pipeline. Knowing exactly which regions of these textures are accurate and which are occluded, we combine several textures of a single synthetic identity into one high-quality texture through Poisson blending. 99.3 thousand of these ground truth textures are used for model training.

2. Training a model that converts incomplete into complete textures. Filling missing pixels on an image is a well-defined problem in computer vision known as "image inpainting," and several inpainting GANs have been designed to solve it. We use a modified version of the deepfillv2 GAN Yu et al., 2019, based on gated convolution - a partial convolution mechanism that takes image masks into account. Rather than learning to fill in random brush strokes, we create a large set of occlusion masks from natural partial head textures, then randomly select one of these occlusions as a mask during training. The result is a model which successfully inpaints a texture even on partial textures where only 40-50% of the face is clearly visible.

1.3 Thesis structure

The remaining portion of this thesis is structured as follows:

- In chapter 2 we provide a detailed overview of existing research related to our project. In particular, we touch on 3D morphable models, face reconstruction, and texture inpainting.
- In chapter 3 we define our machine learning pipeline. This includes the methods used to generate a dataset of completed face textures, and the model for inpainting partially occluded textures.
- In chapter 4 we describe the pipeline's training process, and showcase qualitative evaluations of our pipeline used on real-world training data.
- Finally, chapter 5 draws conclusions about the completed work, and presents some opportunities for future improvements in the area.

Chapter 2

Related work

2.1 3D Morphable Models

In a 3D head reconstruction pipeline, a machine learning model is trained to learn a mapping between an input 2D photograph and an output 3D model by minimizing the difference between its reconstructed output and some given ground truth. Rather than 3D models, this fitting process most often produces statistical data structures known as 3D Morphable Models, or 3DMMs.

A 3DMM refers to a process used to parameterize a large dataset of head models into a set of principal components,as well as the data structures produced by this process. It consists of several high-dimensional spaces representing shape, texture and optionally, more complex parameters such as illumination or expression. The basis for this space is created by performing dimensionality reduction on a large set of 3D models, optimizing the number of parameters needed to accurately represent a model while maintaining as many distinguishing characteristics as possible.

This representation of a face model presents several advantages. First, it constrains a model's outputs to viable human faces. In the worst-case scenario, a face reconstruction model that outputs 3DMM parameters will produce an inaccurate face, while a model trained to into a set of vertices and textures can generate arbitrarily inaccurate outlier outputs. Second, it allows a model to use loss functions more complex than spatial distances between the generated and reference models. Finally, by partially randomizing the parameters of a 3DMM, large numbers of constrained synthetic faces can be generated. One example of this use case in action is the 300W-LP dataset (Zhu et al., 2019), where 3DMMs were projected onto in-the-wild photos at uncommon angles to aid in training 2D facial processing algorithms.

3DMMs can be broadly separated into two subsets: models that use a single linear space to model three-dimensional shapes and alternative, non-linear models. The Surrey Face Model (Huber et al., 2016) is one example of a classic linear 3DMM.

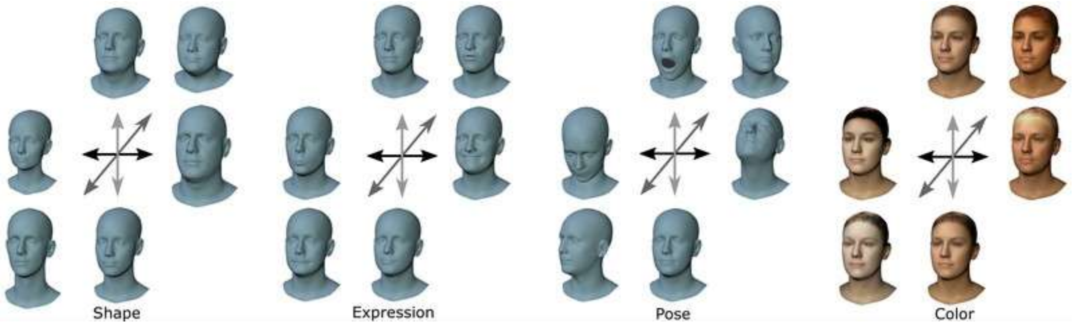

FIGURE 2.1: Visualizations of principal component spaces for the FLAME 3DMM (Sanyal et al., 2019a).

This model defines two linear spaces - one to represent the shape of the face and another for the color and texture. Both sub-models are built by running PCA on a set of 169 high-resolution face scans, which outputs three components: a mean of the recorded facial scans, a set of principal components, and standard deviations for each component. A face under the 3DMM space is represented by a linear product of relative coordinates and principal components, both in the shape and texture spaces. The Large Scale Facial Model (LSFM) (Booth et al., 2018) uses a similar approach on a far grander scale. This model is created from 9663 facial identities of various ethnicities and genders; alongside a global PCA space, this quantity of data allows it to be split into sub-models tailored to different ages, genders, and ethnicities.

Although morphable models based on a single linear space are still used in state-of-the-art face reconstruction pipelines (e.g., Lattas et al., 2020a), alternatives have been designed to address deficiencies in this approach. For instance, the GM-3DMM Gaussian Mixture model (Koppen et al., 2018) uses a combination of Gaussian distributions with different means rather than a single distribution. While otherwise people of different ethnicities would be merged into a single mean face, they can be far more accurately represented by an approach such as this.

3DDFA, the face recognition pipeline which serves as a basis for this project, uses the Basel Face Model (Paysan et al., 2009) as a baseline 3DMM. The model has been trained on 200 participants - 100 female and 100 male, most of them European. After scanning by a low-latency coded light system, model smoothing, and texture extraction, the model's creators parameterize the faces as triangular meshes with $m = 53490$ vertices. Each vertex is associated with an RGB color representing face texture. Thus, a face under this model is initially represented by independent shape and texture vectors of size $3m$:

$$\mathrm{s} = (x_1, y_1, z_1, \ldots, x_m, y_m, z_m)^T \tag{2.1}$$

$$\mathrm{t} = (r_1, g_1, b_1, \ldots, r_m, g_m, b_m)^T \tag{2.2}$$

After PCA is applied to the dataset of 300 input models, the authors provide a best-fit Gaussian distribution for each of the two spaces:

$$M_s = (\mu_s, \sigma_s, u_s); M_t = (\mu_t, \sigma_t, u_t) \tag{2.3}$$

Here, $\mu_{s,t}$ are the parameters of the mean faces, $\sigma_{s,t}$ are the standard deviations and $u_{s,t}$ are the orthonormal basis of principal components. A 3D face within the 3DMM's shapes is represented as a linear combination of the principal components within the texture and shape spaces:

$$s(\alpha) = \mu_s + u_s \mathrm{diag}(\sigma_s)\alpha \tag{2.4}$$

$$t(\beta) = \mu_t + u_t \mathrm{diag}(\sigma_t)\beta \tag{2.5}$$

Other models enhance a linear shape and texture space with additional spaces and parameters. For instance, the FLAME model (Sanyal et al., 2019a) includes an articulated jaw, neck, and eyeballs, alongside a global "expression" space that enables the model to reconstruct a set of standardized facial expressions. Formally, it is described by the function

$$M(\overrightarrow{\beta}, \overrightarrow{\theta}, \overrightarrow{\psi}) = W(T_P(\overrightarrow{\beta}, \overrightarrow{\theta}, \overrightarrow{\psi}), J(\beta), \theta, \omega) \tag{2.6}$$

which maps shape $\overrightarrow{\beta}$, pose $\overrightarrow{\theta}$ and expression $\overrightarrow{\psi}$ vectors into N three-dimensional vertices. The template function T adds shape, pose, and expression offsets to a template mesh $\overline{T}$, which is learned from a set of three-dimensional scans when training the model - shapes are then defined as displacements from this mean. The skinning function W rotates the vertices of this template around joints J and smooths them by the blendweights ω .

While much research has been focused on facial models, it should be noted that comparatively fewer models have been designed to encompass full head anatomy. One notable example of such a model is the Liverpool-York cranial model (Dai et al., 2019), designed to be used as a reference tool by craniofacial surgeons to determine whether a reconstructed head shape is considered 'normal'. Several recent papers in the field have been based on extending this baseline model with additional shape spaces, such as an ear space (Dai, Pears, and Smith, 2019) or face and eye space (Ploumpis et al., 2020). However, most existing techniques and pipelines are still limited to the reconstruction of the facial region.

2.2 Morphable model fitting methods

A 3D reconstruction pipeline trains a machine learning model to discover a linear combination of 3D morphable model parameters that best fits a given input image. This process involves defining a loss function that determines the difference between two parameterized morphable models - either a general-purpose metric such as Euclidean distance and cross-entropy loss; or a custom loss function. Models that have been used to optimize this 3D model's parameters range from simple regressors to deep neural networks and CNNs.

Before the advent of deep learning in computer vision, classic regression algorithms were a commonly used approach for training 3DMM reconstruction models. As an example, Huber et al., 2015 propose a cascaded regression method based on a series of manually selected local features, which finds the most likely vector of PCA shape space coefficients. The landmark points used by this algorithm correlate with distinct facial characteristics, like eyes or mouth corners, although they can be equidistantly spaced on a 3D mesh or selected using other sampling methods. Xiangyu Zhu et al., 2015 use a similar set of fixed landmark features but project both the ground truth and model-synthesized images into a new, transformed space to smoothen the loss function and prevent it from converging on a suboptimal local minimum.

A loss function based on the difference between reconstructed and ground-truth 3DMMs requires a set of ideal 3D scans, but these are difficult to obtain and cannot be acquired for "in-the-wild" input images. This lack of data may lead to inaccuracies when trying to recreate a model from a low-quality image. Therefore, some algorithms use a loss function based on a projected 2D image synthesized from the regressed 3DMM, rather than differences between the model coefficients themselves. As an example, Piotraschke et al. Piotraschke and Blanz, 2016 use a distance metric based on the reconstructed image alongside an automatic landmark detector; an improvement over previous models, where landmarks were based on fixed facial features. A unique aspect of their approach is the ability to enhance a single-image reconstruction with multiple images of the same person, which are merged into a single weighted model.

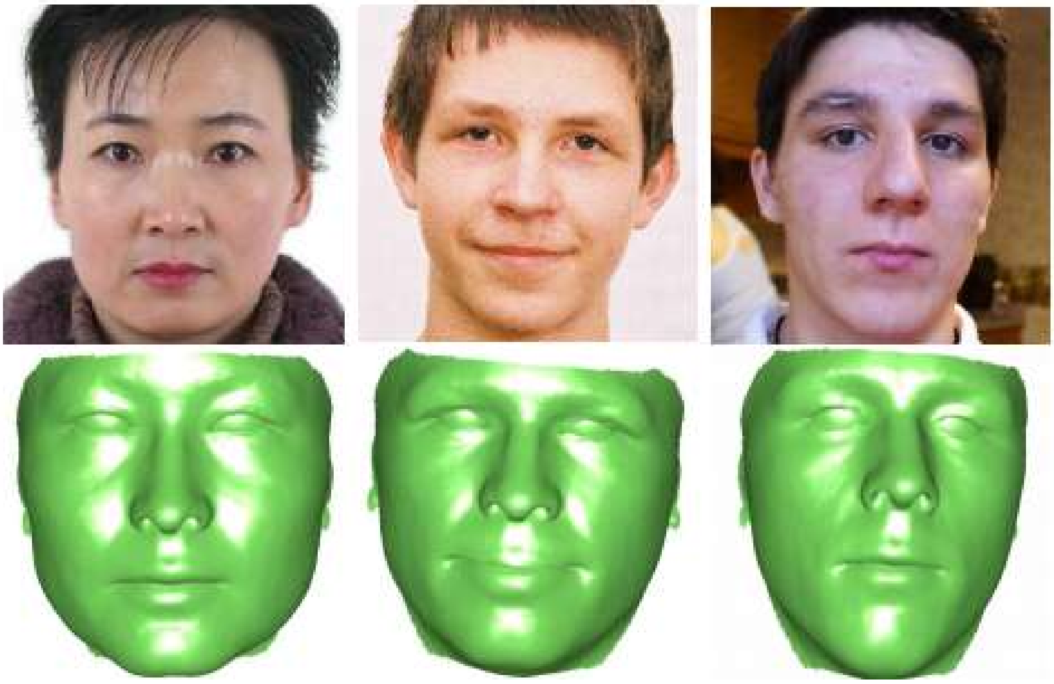

FIGURE 2.2: Fitting results of a regressor-based pipeline (Xiangyu Zhu et al., 2015) on real-world data. This model accurately derives a facial angle from an image, but does not yet represent finer details or expressions.

2.3 Deep learning in 3DMM fitting methods

In the last few years, approaches based on deep neural networks have steadily gained popularity in computer vision, and the facial reconstruction problem is no exception. One major caveat of classical regressors is that some prior statistical knowledge of the 3D morphable model, such as mean images and standard deviation, must be built into the algorithm itself. If the input dataset of a 3DMM is insufficiently robust, this may lead to subpar performance when working with underrepresented ethnicities or groups. DNNs have no such constraints - an existing network architecture can be used to regress any set of 3DMM parameters. Some methods such as Jackson et al., 2017 avoid using a 3DMM entirely, directly regressing the 3D facial geometry from a single image with the use of a generalized cross-entropy loss function. While this allows DNNs to capture fine details that would otherwise be lost due to a 3DMM's reliance on dimensionality reduction, it leads to subpar results when dealing with outlier inputs.

The networks and architectures used in 3D reconstruction are often quite similar to computer vision's state-of-the-art. Savov et al., 2019 use an AlexNet-based CNN model to fit an image into a low-dimensional shape space. Unlike traditional 3DMMs, their representation includes separate feature vectors for face shape, expression, skin reflection, and illumination. Aside from face reconstruction, the model is trained to predict age from the input image; the network used for this auxiliary task shares the weights of the main CNN. The generative nature of facial reconstruction models also makes them a good fit for GAN-based models. GANFIT (Gecer et al., 2019) is an example of a generative adversarial network fitted for facial reconstruction, using several variations of content loss between the input image and a projected rendered image as cost functions to regress a 3D face model. One

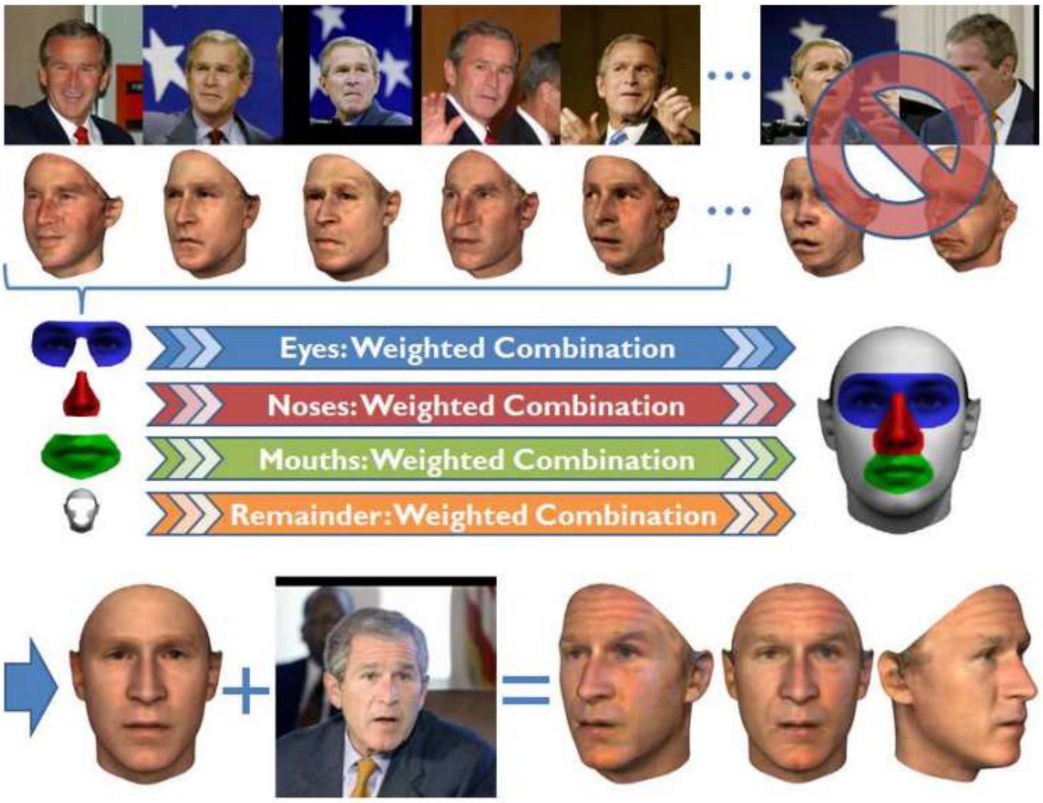
Eyes: Weighted Combination
Noses: Weighted Combination
Mouths: Weighted Combination
Remainder: Weighted Combination

FIGURE 2.3: An example of the pipeline defined in Piotraschke and Blanz, 2016, which combines multiple facial reconstructions based on their accuracy while rejecting poor-quality samples.

of the most recent models in the field, AvatarMe (Lattas et al., 2020b), builds upon GANFIT to reconstruct initial shape texture models, then extends its output texture with a super-resolution network and de-lights the texture to mitigate the effects of uneven illumination.

2.4 Deep learning for image inpainting

Textures produced by traditional head reconstruction pipelines often ignore or naively interpolate areas that cannot be derived from the original photograph. Indeed, one of the weak points of a loss function based on the difference between an input image and interpolated projection is that it does not consider obscured areas at all. To recreate a full face texture, a reconstruction pipeline must utilize models for image inpainting - the task of recreating mission regions in a 2D image while maintaining visual fidelity and scene correctness. The most common use for image inpainting is removing some unwanted objects from a photo and filling the remaining space in with a plausible background. However, the technique can be extended to many other tasks such as image stitching, background harmonization, and text removal.

Early image inpainting algorithms have exploited PatchMatch's (Barnes et al., 2009) ability to establish correspondence between small image regions, or "patches", by iteratively filling in the missing region of an image with patches taken from the surrounding background. However, this approach is limited by visible areas of the image itself and cannot inpaint very large patches, nor does it learn a high-level visual understanding of an image. On the other hand, methods based on DNNs

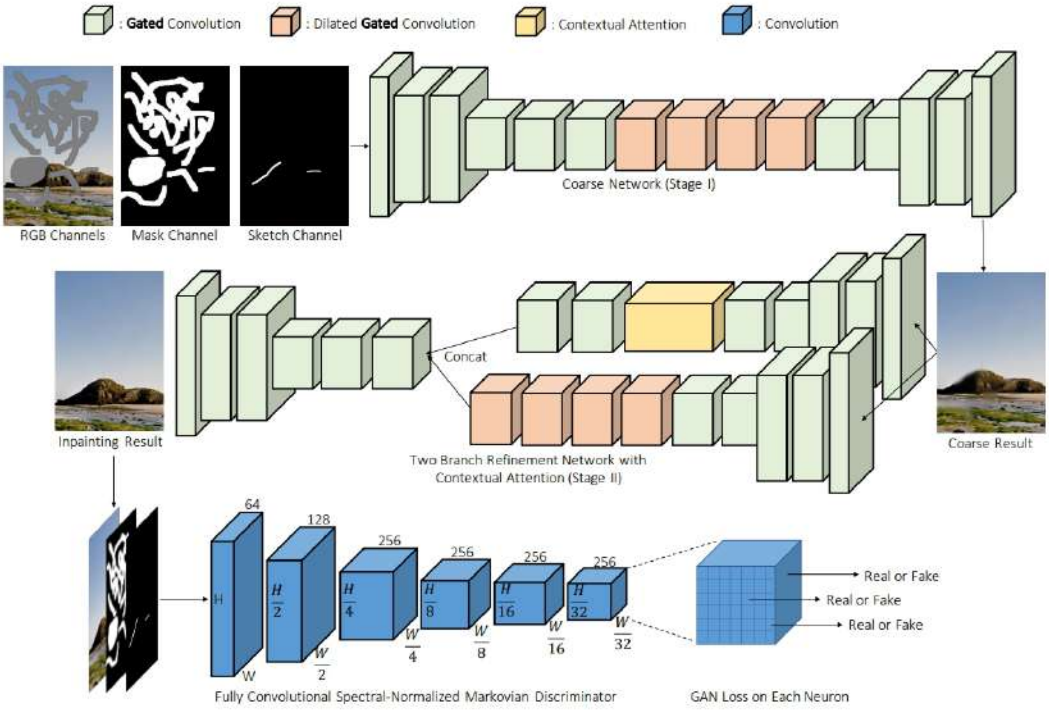

FIGURE 2.4: An overview of the image inpainting network defined by Yu et al., 2018, based on a coarse-to-refined architecture and using novel "gated" convolution in place of regular image convolutions.

typically use image convolutions to fill in the missing regions, sometimes after they are pre-filled with some placeholder values (e.g., the image's mean value).

Unlike regular convolution, the masked areas of an image should not be factored into the model's output - thus, a modified convolution operator is typically used by the models. For instance, Liu et al., 2018 define a custom convolution referred to as partial convolution: given a set of pixel values X, a feature mask M, and a convolution filter with weights W and a bias b, they first calculate a convolution which does not factor in the masked values:

$$X' = \begin{cases} W^T(X \cdot M)\frac{sum(1)}{sum(M)} + b & \text{if } sum(M) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.7}$$

Afterwards, the mask is updated in locations where the convolution has successfully conditioned its input on some valid value. In an iterative process, this eventually ends with the entire mask being filled in:

$$M' = \begin{cases} 1 & \text{if } sum(M) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.8}$$

Yu et al., 2019 propose a further extension referred to as "gated convolution" - rather than classifying all locations in an image to be either valid (unmasked) or invalid (masked) through a rule-based approach, it learns a partial mask based on two different convolutional filters W_g and W_f:

$$M = \phi(W_f^T \cdot X) \cdot \sigma(W_g^T \cdot X) \tag{2.9}$$

Here, ϕ is any activation function and σ is a sigmoid function - which constrains the overall inputs of the mask to values between 0 and 1. The authors use this proposed convolution, alongside a custom SN-PatchGAN loss, to create a generative model with coarse and refinement networks (largely based on Yu et al., 2018). The generative network takes in an image with an arbitrarily shaped missing region, alongside a mask specifying the location of the region, and outputs a completed image. An initial coarse network is trained with reconstruction loss only and makes a first rough prediction, while the refinement network, using both reconstruction and GAN losses, improves the model's results. During training, random masks are generated by applying a series of random brush strokes to the image. (However, our pipeline uses an alternative, semi-random set of image masks as described in the following chapter.)

Chapter 3

Proposed method

A notable omission in many 2D-photograph-to-3D-model reconstruction pipelines is the ability to create a fully textured model of a human head. In these pipelines, textures of the ear, neck, and hair that cannot be directly derived from the photograph are replaced by a generic placeholder or, in some cases, omitted entirely. Research related to deriving complete models of the human head such as (e.g., Dai, Pears, and Smith, 2019, Ploumpis et al., 2020) has so far primarily focused on defining an alternative 3D morphable model of the human head rather than reconstruction from in-the-wild images.

The main objective of our pipeline is specifically designed to reconstruct fully textured 3D head models, narrowing this research gap.

3.1 Dataset collection

The first step of our work was to review existing datasets of 2D face images and 3D head models in order to find a large set of "ground truth" textures usable for large-scale GAN training. To the best of our knowledge, no publicly available dataset with identities numbering in the tens of thousands exists. Some relevant datasets contain a very small number of test subjects - 938 for FaceScape (Yang et al., 2020), 100 for "Not quite in-the-Wild" (Sanyal et al., 2019b), while others have only been made available for medical research purposes (e.g. the "MeIn3D" dataset used for training LSFM by Booth et al., 2018).

An alternative solution was to create a synthetic dataset containing a sufficient number of ground truth textures. From several different images of the same person at different angles, but ideally, under the same lighting conditions and the same expression, we can create a single high-quality UV texture through Poisson blending.

We start this process by generating a large number of synthesized face images via the EigenGAN generative CNN (He, Kan, and Shan, 2021), using a version of the model pre-trained on the Celeb Face A dataset. Studies aiming to analyze and understand GAN learning (Bau et al., 2018) have shown that different layers within a GAN generator are responsible for different attributes and properties of the synthetic images; generally, deeper layers relate to the spatial layout of the image, while shallower layers - to the color. EigenGAN's most notable feature is the ability to control these attributes. Alongside a chain of convolutional blocks, the model embeds a linear subspace with an orthonormal basis $U = [u_{i1}, \ldots, u_{iq}]$ into the pipeline; each of the basis vectors u_{iq} aims to discover an interpretable dimension that relates to some attribute of the image.

EigenGAN's pre-trained pipeline generates 256×256 images and features 36 dimensions that can be used to alter their attributes while maintaining the same overall synthetic identity. To obtain a wide range of photographs for the same person,

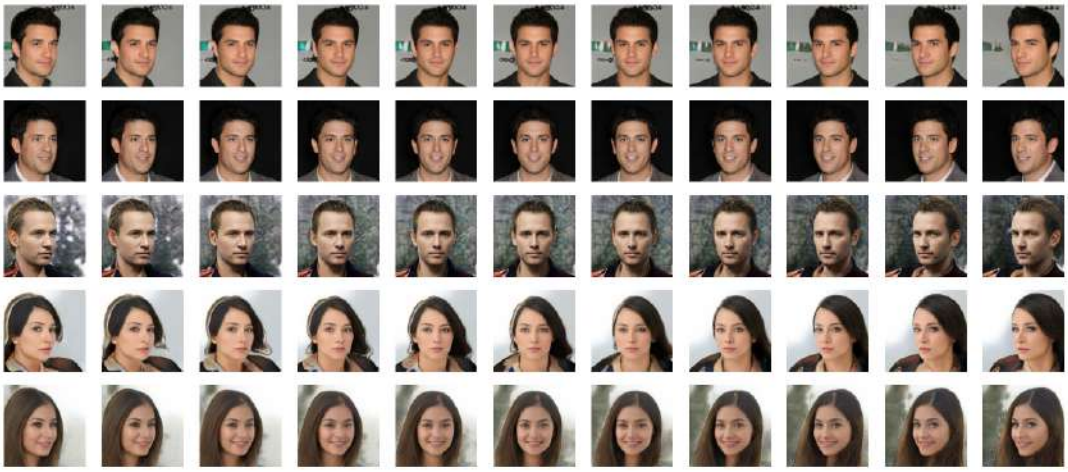

FIGURE 3.1: A sample from our initial photo dataset, with five synthetic identities generated by EigenGAN at 11 different angles.

we manipulate an EigenGAN dimension which is used to alter the generated photograph's horizontal angle or yaw pose (the first dimension of the fourth layer - L4D1). Our pipeline generates 11 different angles of 100,000 synthetic identities - one image with an angle pre-selected by the model and 10 variations in yaw pose between -2σ and 2σ, where σ is the model's standard deviation.

3.2 UV texture generation

The next step of our pipeline is to convert the synthetic images into high-quality, yet incomplete, PAFs, or Pose-Adaptive Features. To achieve this, we turned to the 3DDFA implementation of Zhu et al., 2019. Given an in-the-wild input image, the pipeline can derive and render an instance of the 2017 Basel Face Model 3DMM as a set of .ply and .obj files, which can be rendered by an open-source mesh processing application such as MeshLab. Additionally, several auxiliary outputs can be obtained from the pipeline: an estimation of the image's depth, 2D projections referred to as PNCCs (Projected Normalized Coordinate Code), and 2D UV textures as PAFs (Pose-Adaptive Features).

In our pipeline, we limit the 3DDFA code to outputting 3D face estimations as a set of 68 points projected onto the original image and, most notably, the UV textures projected from the input photograph. Using our synthetic dataset, we generate high-quality - yet incomplete - PAFs for each of the 1.1 million photographs. This generation process also serves as an initial quality gate: if the model is unable to detect a face on one of the 11 generated photographs, the entire identity is discarded, as it is likely to be poorly generated. 725 photos, or 0.06% of the original input dataset, are rejected at this step. We also modify the model training script to function in batch mode, significantly speeding up the training process; batches of sizes 5-6 were used during initial experimentation, and 50 for large-scale training.

3.3 UV texture blending

The partial texture generation step of the pipeline leaves us with a set of similar UV textures for each synthetic identity, each with some degree of artifacting caused by self-occlusion. However, the areas of the image containing poor-quality texturing

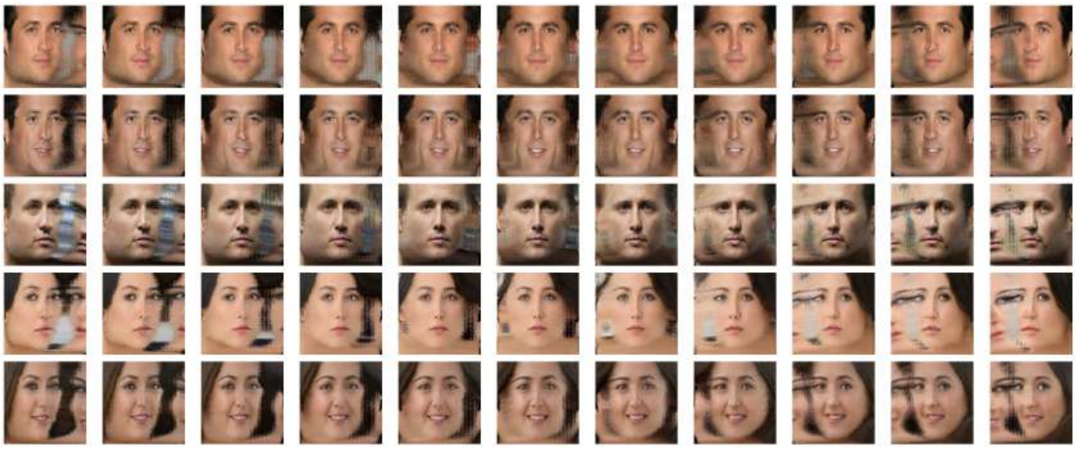

FIGURE 3.2: A sample from the incomplete UV texture dataset, with all faces from Figure 3.1 converted to PAFs by 3DDFA.

are generally well-known - images angled to the right will display a high-quality texture on the left and vice versa. Therefore, the good-quality areas of the textures can be merged into a single cohesive texture.

To generate complete UV textures, we select the second, sixth, and eighth images - roughly corresponding to horizontal angles at -35°, 0°and 35°- and combined into a single UV texture as follows:

- Select the left horizontal third of the first image, the central third of the second image, and the right third of the final image. Separate them into three partial images.
- Calculate the center point of the left and right images, to be used later on.
- Overlay the central image onto the center of the left image; then, overlay the central image onto the center of the right image. OpenCV's seamlessClone function, a Poisson blending algorithm (Pérez, Gangnet, and Blake, 2003), is used to seamlessly combine the images.
- Combine the left and right images into a single texture, once again using Poisson blending to smooth out any possible edges.
- Overlay the central image onto the texture through a final iteration of Poisson blending. This removes any vertical seams or variations in lighting found in the first two blending rounds.
- Serialize and save the final result.

Full source code for this algorithm is provided in Appendix A. 521 poor-quality textures are manually discarded at this stage, usually due to GAN artifacts being overlaid onto the leftmost or rightmost images. The result is a final dataset of 99381 UV textures at 192 × 192 resolution.

3.4 UV texture completion

Once our dataset of ground truth UV textures is available, we move on to the second phase of our work - training an inpainting GAN that, given a partially completed texture and a UV mask, fills in the occluded regions with a best possible estimation.

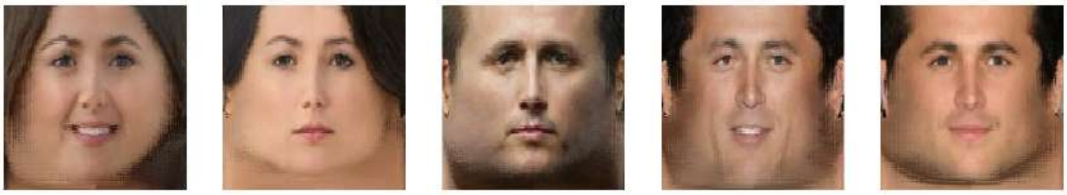

FIGURE 3.3: Several real-world occlusion masks used during model training.

As the basis for this inpainting GAN, we have used deepfillv2, an implementation of the gated convolution-based pipeline created by Yu et al., 2019. The most notable feature of this model is gated convolution: a convolution function that takes into account both masked and unmasked areas of an image as described in chapter 2. Rather than the SN-PatchGAN loss used by the original paper, the authors opt to use a modified WGAN-GP, or Wasserstein GAN loss (Arjovsky, Chintala, and Bottou, 2017). Normally, Wasserstein distance measures the difference between two probability distributions, being interpreted as the minimal amount of 'effort' needed to convert one distribution to another. Within the context of a GAN, it tries to minimize the distance between the distributions of real and fake samples. The authors have shown this loss to result in more stable training than Kullback–Leibler and Jensen–Shannon divergence as described in the original implementation of a GAN (Weng, 2019).

One notable modification to the initial pipeline is the algorithm used for generating masks. In order to properly adapt to real-world data, masks used while training the model should be, to some extent, randomized to avoid overfitting but also sufficiently similar to those found in real-world use-cases. The free-form inpainting paper and its implementation introduce an algorithm that randomly generates masks during training by drawing several random straight lines and smoothing out their intersections. However, occlusions on a real-world UV texture are likely to be concentrated on one side of the image and feature much sharper edges than masks produced by the randomization algorithm.

To mitigate this issue, we first introduce an algorithm to convert incomplete UV textures generated by 3DDFA to masked images rather than interpolated projections. We achieve this by masking parts of the original photograph that are not considered part of a face and, as such, are likely projected to background noise. Afterwards, we generate 1000 partially textured UV textures that are not used during the main training loop and extract their masked regions as 192x192 images, where white pixels denote unmasked areas, and black pixels signify masked areas. During training, we modify the random mask generation function to set an 80% chance of selecting a real-world mask and a 20% chance of falling back to the original inpainting generation function. Additionally, a set of hyperparameters found in Appendix B is used, with alterations to the number of epochs, GAN loss and loss weights.

3.5 End-to-end 3D model reconstruction

After the GAN is trained and evaluated, we modify the 3DDFA face reconstruction pipeline to output full 3D head textures. To do this, we introduce two custom steps into the facial reconstruction pipeline as follows:

- A single in-the-wild image is received as an input for the timeline.

FIGURE 3.4: A sample from the complete UV texture dataset, containing merged textures from three angles of Figure 3.2.

- The dlib (King, 2009) face reconstruction library is used to detect a face on the image and output a bounding box, defined as a four-point rectangle. Based on this bounding box, the photograph is cropped to the face only.

- Using the pre-trained MobileNetV1 model, 3DDFA outputs a set of parameters for the Basel Face Model 3DMM. Based on these, we predict 68 landmarks defined by the 3DMM (denoting the face, eyes, and ears) and project them onto the image.

- *Rather than outputting a partially interpolated UV texture, we use the 68-landmark projection to crop out the photograph to only those points found on the 3D face mesh directly. This is used to output a UV texture where the facial regions that would otherwise be interpolated are instead left masked.*

- *Both the masked UV texture and the mask itself are passed to the deepfillv2 GAN model as inputs. Using the model, we predict a complete UV texture (defined as the output of the model's rough layer) and save it as an image of the same resolution as the inputs.*

- The 3D model and UV textures are returned as joint outputs of 3DDFA and deepfillv2. Additional visualization-related information such as predicted pose, image depth, and PNCC are saved as needed. The combination of the 3D model and textures can be visualized in Meshlab or similar 3D modelling software.

Chapter 4

Training and Evaluation

4.1 Training and computational details

To generate the dataset used for the inpainting network, we have first generated 1.1 million images, then converted them to partial UV textures based on the 3DDFA pipeline. On a GTX 1070 GPU, the image generation step took 6.5 hours of training, while the combination of UV texture generation and Poisson blending has been a 36-hour-long process. Our version of the 3DDFA pipeline has been modified to enable batch training - as such, we were able to process the images in batches of size 8, notably speeding up the process.

The modified version of the deepfillv2 image inpainting pipeline has been trained for 20 epochs on a single RTX A6000 GPU, using 99,368 images of size 192×192 as a training set. We have opted to use WGAN as the baseline architecture for training, setting a generator learning rate of 10^{-4} for the generator and 4×10^{-4} for the discriminator; this learning rate has been decayed by a factor of 2 on the 10th epoch.

4.2 Quantitative results

The evaluation of our model is primarily focused on qualitative results. To the best of our knowledge, no inpainting GAN similar to ours is publically available for comparison purposes and, as such, we cannot compare our inpainted textures with those generated by other pipelines. However, we do measure GAN losses on a test set of 500 images. The results below are interpreted as follows:

1. **Mask L1 loss** - in accordance with the original paper, the deepfillv2 implementation features a coarse-to-refined architecture, where an initial first result is further refined by a second refinement layer and returned as the final input. First mask L1 loss denotes the loss of the coarse layer mask, while the second mask L1 loss - that of the refined layer. Interestingly, the refinement layer did not lead to significant improvements on the resulting textures and, in fact, increased the number of visual artifacts on many results; as such, the 'rough' first layer results are used further in this section and in the final pipeline.

2. **D loss** - The overall loss between the real and fake scalar images generated by the GAN.

3. **GAN loss** - The GAN loss between the real and fake probability distributions. In our pipeline, the 1-Wasserstein distance is used as a probability distribution difference measure.

TABLE 4.1: Quantitative results of our final model (20 epochs, batch size 50).

Metric	Value
First mask L1 loss	1.36%
Second mask L1 loss	2.02%
D Loss	2.17%
GAN Loss	0.8474
Perceptual loss	7.09%

4. **Perceptual loss** - The L1 loss between deep semantic feature maps of the VGG-16 network; essentially, a patch-level L1 loss rather than a function pertaining to the entire image.

4.3 Qualitative results

The results of our network, tested on a set of synthetic UV textures that were not used for training, are presented below. The first part of each image represents the ground truth UV texture, the second - a masked texture, and the third displays the generator′s "rough layer" output, which is also used as our final result.

From these, we can observe the following patterns:

- On results with relatively small masks, the model inputs are close to indistinguishable from the ground truth textures: skin color, texture, and lighting are all successfully extended by the model. Ear textures are successfully generated from scratch, and despite the jagged artifact-like appearance of the masks, no trace of their rectangular patterns is left on the final image.
- With medium-sized masks, the model still performs well with certain caveats. If a large portion of the face is missing, a "mean face" that does not preserve as many details as the original texture yet accurately mirrors the visible portion of the texture. If some part of the hair is unmasked, the overall hairstyle is generally preserved; otherwise, it is interpreted as a more simple, diagonal "mean hair." However, there are few to no instances where the model generates an inappropriate or unrealistic face inpainting.
- Perhaps most surprisingly, the model is capable of generalizing well on images where $40 - 50\%$ of the image is occluded - this is common if a photograph is taken at an extreme side angle. It successfully reconstructs entire eye details, ears and hair with no prior reference largely by mirroring the visible texture. However, at this stage of occlusion, face color is somewhat distorted into a common beige mean and does not always match the original.

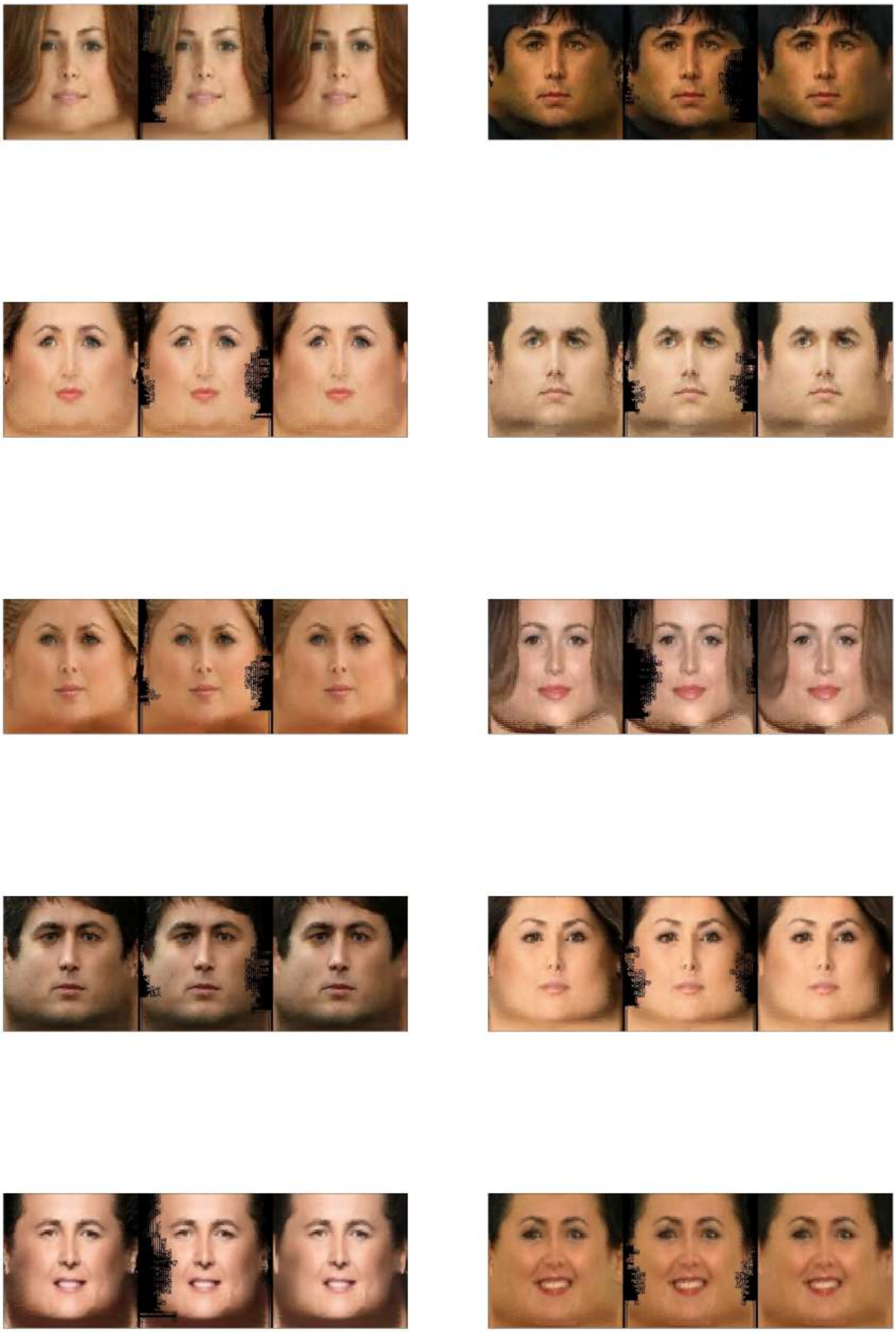

FIGURE 4.1: A sample of results on the test set that were reconstructed from small masks (those obscuring $< 20\%$ of an image)

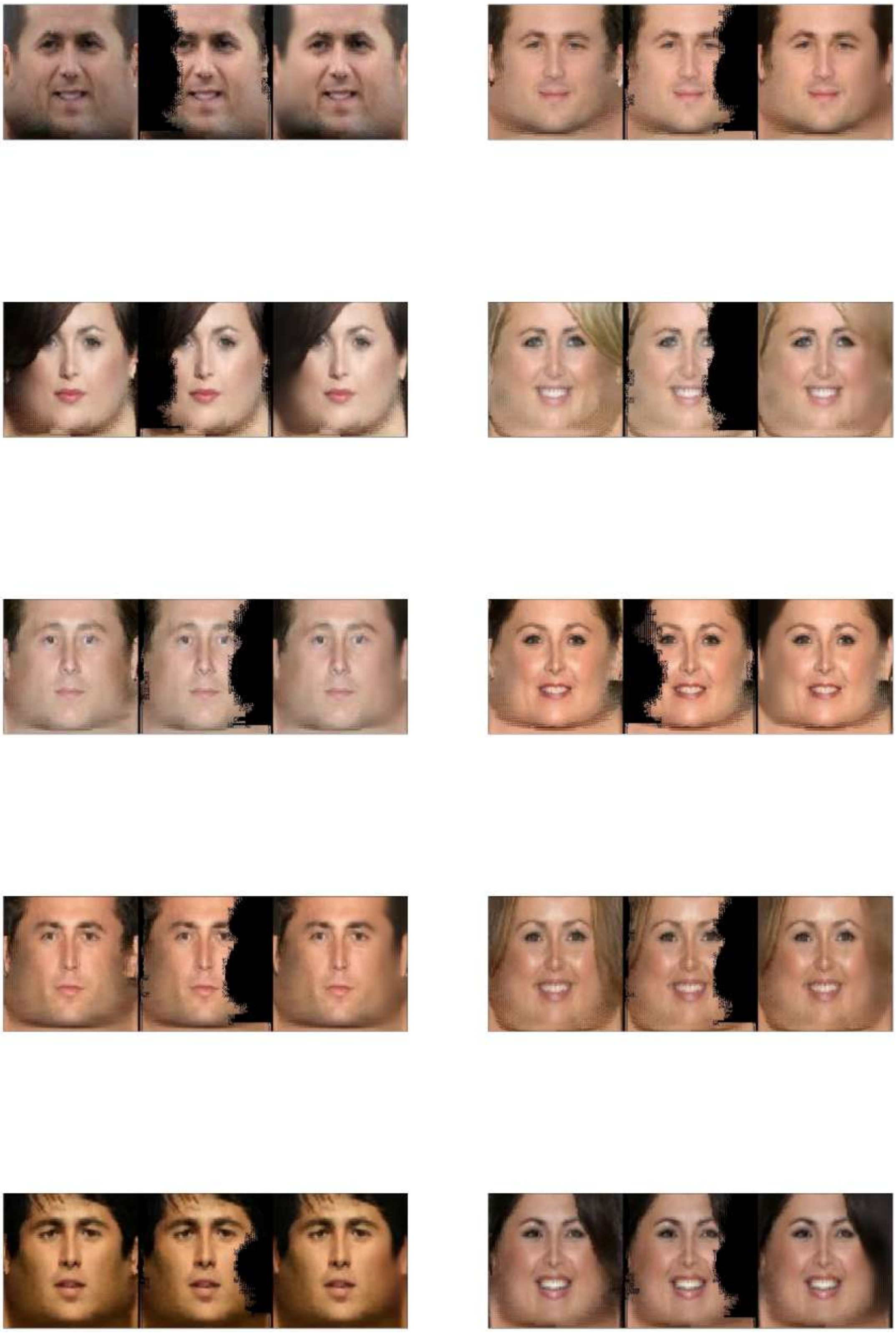

FIGURE 4.2: A sample of results on the test set that were reconstructed from medium-sized masks (those obscuring between 20% and 40% of an image)

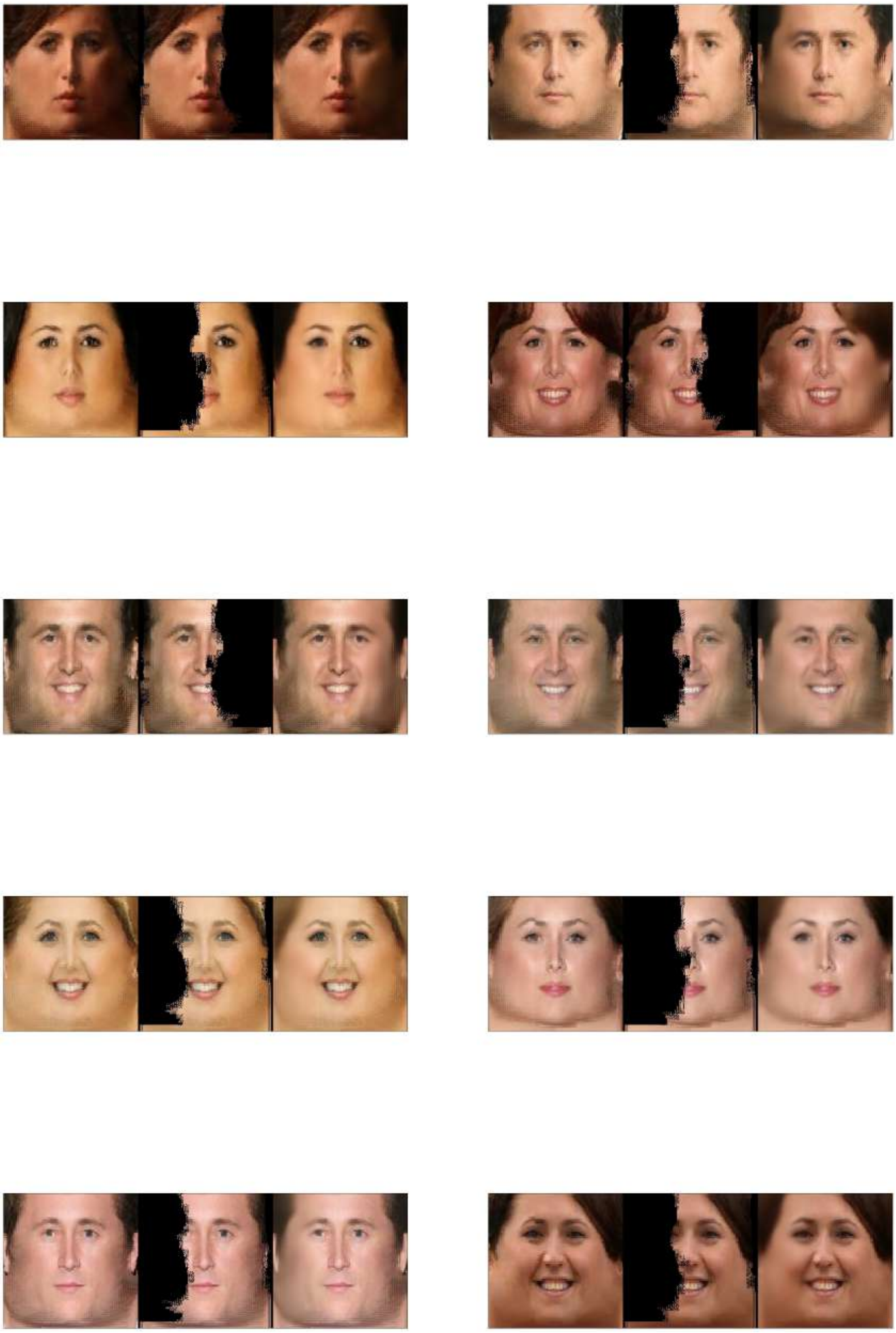

FIGURE 4.3: A sample of results on the test set that were reconstructed from large masks (those obscuring > 50% of an image)

Chapter 5

Conclusions and Future Work

5.1 Conclusions

The main goal of this thesis project was to enhance a typical 3D head reconstruction pipeline to be able to recreate a fully textured head model. To that end, the main steps that had to be undertaken were to create a large-scale database of UV textures of a size suitable for training GANs and other DNNs and train an inpainting network based on this database to be able to convert incomplete textures into complete ones.

During the dataset generation step, we have found that only a tiny percentage (<0.8%) of all generated images had to be rejected due to visual artifacts or missing facial detection, and the UV textures look overall natural and seamless despite the use of Poisson blending. The inpainting network itself has successfully managed to recreate facial details, including the finer details of the eyes and lips, even on masks as large as 40-50% of an image. The benchmark results for most inpainting networks use masks totaling 10-20% of the image, making this a surprisingly good result.

One notable limitation of our dataset is its tendency to generate generally attractive, "celebrity-looking" adult faces, as it has used a generative network pre-trained to on the Celeb Face A dataset. If tested on data with people of uncommon ethnicities or ages, the model may not successfully recreate finer facial details such as wrinkles or multicolored hair. Given additional "in-the-wild" photo sets a generative GAN can be trained on, it is possible to pass them the ′multiple images - multiple UV textures - single UV texture′ generation pipeline and increase the overall representativeness of our model.

Overall, we have shown that our approach to face texture recreation can be successfully used as part of a face reconstruction pipeline. Both our collected dataset and the pre-trained model can be used as a baseline for further research in this subject area.

5.2 Future work

Given time and opportunity for additional improvement, there are several potential improvements and directions for further work on this project:

1. **Additional datasets and data sources.** While the EigenGAN network used as a base for our synthetic data was pre-trained on the Celeb Face A dataset, it is not limited to this dataset alone - an alternative model for the Danbooru2019 anime dataset is provided by the researchers, though its subject matter is outside the scope of in-the-wild photography. Nevertheless, other large-scale face datasets such as Flickr-Faces-HQ (Karras, Laine, and Aila, 2019) or UMD Faces

(Bansal et al., 2017) can be used as a seed for the generative network to generate, for instance, non-"celebrity" photographs at a variety of different angles and increase the overall robustness of our model.

2. **Alternative face reconstruction pipeline.** The 3DDFA pipeline has been chosen largely for the quality of its partially generated UV texture, as well as ease of implementation and publically released network weights. Other face reconstruction pipelines meeting these criteria can be considered in its place - these include 3DDFA v2 (Guo et al., 2021), RingNet (Sanyal et al., 2019c) and DECA (Feng et al., 2020). However, all of these pipelines will need to be extended with a method of extracting UV textures from the input image, as this capability is not provided out of the box.

3. **Experiments with different inpainting networks.** Alternatives to the deepfillv2 pipeline such as the High-Resolution Image Inpainting GAN (Yi et al., 2020) may be used to improve the quality of the generated textures, and quantitative comparisons can be made between several different networks working on the same images and masks after a similar training time.

Appendix A

Partial texture blending algorithm

```
import numpy as np
import cv2 as cv

def blend_textures(left, center, right):
  # Take the middle third part from the central image
  center_mid = np.array_split(center, 3, axis=1)[1]

  # Define mask & center point for seamlessClone
  black = np.full(center_mid.shape, 0, dtype = np.uint8)
  white = np.full(center_mid.shape, 255, dtype = np.uint8)
  mask = np.concatenate([black, white, black], axis=1)
  center_point = (left.shape[1]//2, left.shape[0]//2)
  center_masked = np.concatenate([black, center_mid, black], axis=1)

  # Overlay center middle onto left image, then right image
  im_clone1 = cv.seamlessClone(center_masked, left, mask,
                               center_point, cv.NORMAL_CLONE)
  im_clone2 = cv.seamlessClone(center_masked, right, mask,
                               center_point, cv.NORMAL_CLONE)

  # Merge left & right halves of the results
  clones_with_seam = np.concatenate([
                                     np.array_split(im_clone1, 2, axis=1)[0],
                                     np.array_split(im_clone2, 2, axis=1)[1]
                                    ],axis=1)

  # Overlay center middle again to correct the seam in the middle
  clones_without_seam = cv.seamlessClone(center_masked,
                                         clones_with_seam,
                                         mask,
                                         center_point,
                                         cv.NORMAL_CLONE)

  # Get final result
  stitched_imgs.append(clones_without_seam)
```

Appendix B

DeepFill GAN Training Hyperparameters

TABLE B.1: Hyperparameters used to train the deepfillv2 model.

Metric	**Description**	**Value**
epochs	Number of epochs the model is trained for	40
batch_size	The number of images processed in a single batch	50
lr_g	Generator learning rate	1×10^{-4}
lr_d	Discriminator learning rate	4×10^{-4}
b1	First decay parameter β_1 used in the Adam optimizer	0.5
b2	Second decay parameter β_2 used in the Adam optimizer	0.999
weight_decay	Weight decay in the Adam optimizer; left unused	0
lr_decrease_epoch	The epoch at which learning rate will be decreased	10
lr_decrease_factor	The factor to decrease learning rate by	0.5
lambda_l1	The relative weight of L1 loss	100
lambda_perceptual	The relative weight of perceptual loss	10
lambda_gan	The relative weight of WGAN loss	10
in_channels	The number of input network channels (3ximage + 1xmask)	4
out_channels	The number of output network channels (3ximage)	3
latent_channels	The number of latent network channels	64
pad_type	The padding to be used in tensors	*zero*
activation	The activation function used by the network	*relu*
init_type	The function to be used for initializing network weights	*xavier*
init_gain	The base gain parameter of the initialization function	0.02

Bibliography

Arjovsky, Martin, Soumith Chintala, and Léon Bottou (Dec. 2017). “Wasserstein GAN”. In: *arXiv:1701.07875 [cs, stat]*. arXiv: 1701.07875. URL: http://arxiv.org/abs/1701.07875 (visited on 05/21/2021).

Bansal, Ankan et al. (May 2017). “UMDFaces: An Annotated Face Dataset for Training Deep Networks”. In: *arXiv:1611.01484 [cs]*. arXiv: 1611.01484. URL: http://arxiv.org/abs/1611.01484 (visited on 05/21/2021).

Barnes, Connelly et al. (July 2009). “PatchMatch: a randomized correspondence algorithm for structural image editing”. en. In: *ACM Transactions on Graphics* 28.3, pp. 1–11. ISSN: 0730-0301, 1557-7368. DOI: 10.1145/1531326.1531330. URL: https://dl.acm.org/doi/10.1145/1531326.1531330 (visited on 05/19/2021).

Bau, David et al. (Dec. 2018). “GAN Dissection: Visualizing and Understanding Generative Adversarial Networks”. In: *arXiv:1811.10597 [cs]*. arXiv: 1811.10597. URL: http://arxiv.org/abs/1811.10597 (visited on 05/20/2021).

Booth, James et al. (Apr. 2018). “Large Scale 3D Morphable Models”. en. In: *International Journal of Computer Vision* 126.2-4, pp. 233–254. ISSN: 0920-5691, 1573-1405. DOI: 10.1007/s11263-017-1009-7. URL: http://link.springer.com/10.1007/s11263-017-1009-7 (visited on 12/05/2020).

Dai, Hang, Nick Pears, and William Smith (Sept. 2019). “Augmenting a 3D Morphable Model of the Human Head with High Resolution Ears”. In: *Pattern Recognition Letters* 128. DOI: 10.1016/j.patrec.2019.09.026.

Dai, Hang et al. (2019). “Statistical Modeling of Craniofacial Shape and Texture”. In: *International Journal of Computer Vision* 128.2, pp. 547–571. ISSN: 1573-1405. DOI: 10.1007/s11263-019-01260-7. URL: https://doi.org/10.1007/s11263-019-01260-7.

Feng, Yao et al. (Dec. 2020). “Learning an Animatable Detailed 3D Face Model from In-The-Wild Images”. In: *arXiv:2012.04012 [cs]*. arXiv: 2012.04012. URL: http://arxiv.org/abs/2012.04012 (visited on 05/21/2021).

Gecer, Baris et al. (June 2019). “GANFIT: Generative Adversarial Network Fitting for High Fidelity 3D Face Reconstruction”. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. arXiv: 1902.05978, pp. 1155–1164. DOI: 10.1109/CVPR.2019.00125. URL: http://arxiv.org/abs/1902.05978 (visited on 12/04/2020).

Guo, Jianzhu et al. (Feb. 2021). “Towards Fast, Accurate and Stable 3D Dense Face Alignment”. In: *arXiv:2009.09960 [cs]*. arXiv: 2009.09960. URL: http://arxiv.org/abs/2009.09960 (visited on 05/21/2021).

Guo, Xiao-Yu et al. (Mar. 2020). “The utility of 3-dimensional-printed models for skull base meningioma surgery”. In: *Annals of Translational Medicine* 8.6, pp. 370–370. ISSN: 23055839, 23055847. DOI: 10.21037/atm.2020.02.28. URL: http://atm.amegroups.com/article/view/36839/html (visited on 05/18/2021).

He, Zhenliang, Meina Kan, and Shiguang Shan (Apr. 2021). “EigenGAN: Layer-Wise Eigen-Learning for GANs”. In: *arXiv:2104.12476 [cs, stat]*. arXiv: 2104.12476. URL: http://arxiv.org/abs/2104.12476 (visited on 05/15/2021).

Huber, Patrik et al. (Sept. 2015). “Fitting 3D Morphable Models using Local Features”. In: *2015 IEEE International Conference on Image Processing (ICIP)*. arXiv: 1503.02330, pp. 1195–1199. DOI: 10.1109/ICIP.2015.7350989. URL: http://arxiv.org/abs/1503.02330 (visited on 12/04/2020).

Huber, Patrik et al. (2016). “A Multiresolution 3D Morphable Face Model and Fitting Framework:” in: *Proceedings of the 11th Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*. Rome, Italy: SCITEPRESS - Science and Technology Publications, pp. 79–86. ISBN: 978-989-758-175-5. DOI: 10.5220/0005669500790086. URL: https://www.scitepress.org/DigitalLibrary/Link.aspx?doi=10.5220/0005669500790086 (visited on 12/04/2020).

Jackson, Aaron S. et al. (Sept. 2017). “Large Pose 3D Face Reconstruction from a Single Image via Direct Volumetric CNN Regression”. In: *arXiv:1703.07834 [cs]*. arXiv: 1703.07834. URL: http://arxiv.org/abs/1703.07834 (visited on 12/04/2020).

Karras, Tero, Samuli Laine, and Timo Aila (Mar. 2019). “A Style-Based Generator Architecture for Generative Adversarial Networks”. In: *arXiv:1812.04948 [cs, stat]*. arXiv: 1812.04948. URL: http://arxiv.org/abs/1812.04948 (visited on 05/21/2021).

King, Davis E. (2009). “Dlib-ml: A Machine Learning Toolkit”. In: *Journal of Machine Learning Research* 10, pp. 1755–1758.

Koppen, Paul et al. (Feb. 2018). “Gaussian mixture 3D morphable face model”. en. In: *Pattern Recognition* 74, pp. 617–628. ISSN: 00313203. DOI: 10.1016/j.patcog.2017.09.006. URL: https://linkinghub.elsevier.com/retrieve/pii/S0031320317303527 (visited on 12/04/2020).

Lattas, Alexandros et al. (Mar. 2020a). “AvatarMe: Realistically Renderable 3D Facial Reconstruction "in-the-wild"”. In: *arXiv:2003.13845 [cs]*. arXiv: 2003.13845. URL: http://arxiv.org/abs/2003.13845 (visited on 12/04/2020).

— (Mar. 2020b). “AvatarMe: Realistically Renderable 3D Facial Reconstruction "in-the-wild"”. In: *arXiv:2003.13845 [cs]*. arXiv: 2003.13845. URL: http://arxiv.org/abs/2003.13845 (visited on 12/05/2020).

Lin, Jiangke, Yi Yuan, and Zhengxia Zou (Feb. 2021). “MeInGame: Create a Game Character Face from a Single Portrait”. In: *arXiv:2102.02371 [cs]*. arXiv: 2102.02371. URL: http://arxiv.org/abs/2102.02371 (visited on 05/18/2021).

Liu, Guilin et al. (Dec. 2018). “Image Inpainting for Irregular Holes Using Partial Convolutions”. In: *arXiv:1804.07723 [cs]*. arXiv: 1804.07723. URL: http://arxiv.org/abs/1804.07723 (visited on 05/19/2021).

Park, Unsang and Anil K. Jain (2007). “3D Model-Based Face Recognition in Video”. en. In: *Advances in Biometrics*. Ed. by Seong-Whan Lee and Stan Z. Li. Vol. 4642. ISSN: 0302-9743, 1611-3349 Series Title: Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 1085–1094. ISBN: 978-3-540-74548-8 978-3-540-74549-5. DOI: 10.1007/978-3-540-74549-5_113. URL: http://link.springer.com/10.1007/978-3-540-74549-5_113 (visited on 05/18/2021).

Paysan, Pascal et al. (Sept. 2009). “A 3D Face Model for Pose and Illumination Invariant Face Recognition”. In: *2009 Sixth IEEE International Conference on Advanced Video and Signal Based Surveillance*. Genova, Italy: IEEE, pp. 296–301. ISBN: 978-1-4244-4755-8. DOI: 10.1109/AVSS.2009.58. URL: http://ieeexplore.ieee.org/document/5279762/ (visited on 05/19/2021).

Piotraschke, Marcel and Volker Blanz (June 2016). “Automated 3D Face Reconstruction from Multiple Images Using Quality Measures”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV, USA: IEEE, pp. 3418–3427. ISBN: 978-1-4673-8851-1. DOI: 10.1109/CVPR.2016.372. URL: http://ieeexplore.ieee.org/document/7780741/ (visited on 12/04/2020).

Ploumpis, Stylianos et al. (Feb. 2020). “Towards a complete 3D morphable model of the human head”. In: *arXiv:1911.08008 [cs]*. arXiv: 1911.08008. URL: http://arxiv.org/abs/1911.08008 (visited on 12/04/2020).

Pérez, Patrick, Michel Gangnet, and Andrew Blake (July 2003). “Poisson image editing”. en. In: *ACM Transactions on Graphics* 22.3, pp. 313–318. ISSN: 0730-0301, 1557-7368. DOI: 10.1145/882262.882269. URL: https://dl.acm.org/doi/10.1145/882262.882269 (visited on 05/21/2021).

Sanyal, Soubhik et al. (May 2019a). “Learning to Regress 3D Face Shape and Expression from an Image without 3D Supervision”. In: *arXiv:1905.06817 [cs]*. arXiv: 1905.06817. URL: http://arxiv.org/abs/1905.06817 (visited on 12/04/2020).

Sanyal, Soubhik et al. (June 2019b). “Learning to Regress 3D Face Shape and Expression from an Image without 3D Supervision”. In: *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

Sanyal, Soubhik et al. (May 2019c). “Learning to Regress 3D Face Shape and Expression from an Image without 3D Supervision”. In: *arXiv:1905.06817 [cs]*. arXiv: 1905.06817. URL: http://arxiv.org/abs/1905.06817 (visited on 05/21/2021).

Savov, N. et al. (2019). “Pose and Expression Robust Age Estimation via 3D Face Reconstruction from a Single Image”. In: *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pp. 1270–1278.

Weng, Lilian (Apr. 2019). “From GAN to WGAN”. In: *arXiv:1904.08994 [cs, stat]*. arXiv: 1904.08994. URL: http://arxiv.org/abs/1904.08994 (visited on 05/21/2021).

Xiangyu Zhu et al. (May 2015). “Discriminative 3D morphable model fitting”. In: *2015 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*. Ljubljana: IEEE, pp. 1–8. ISBN: 978-1-4799-6026-2. DOI: 10.1109/FG.2015.7163096. URL: http://ieeexplore.ieee.org/document/7163096/ (visited on 12/04/2020).

Yang, Haotian et al. (Apr. 2020). “FaceScape: a Large-scale High Quality 3D Face Dataset and Detailed Riggable 3D Face Prediction”. In: *arXiv:2003.13989 [cs]*. arXiv: 2003.13989. URL: http://arxiv.org/abs/2003.13989 (visited on 12/24/2020).

Yi, Zili et al. (May 2020). “Contextual Residual Aggregation for Ultra High-Resolution Image Inpainting”. In: *arXiv:2005.09704 [cs]*. arXiv: 2005.09704. URL: http://arxiv.org/abs/2005.09704 (visited on 05/21/2021).

Yu, Jiahui et al. (Mar. 2018). “Generative Image Inpainting with Contextual Attention”. In: *arXiv:1801.07892 [cs]*. arXiv: 1801.07892. URL: http://arxiv.org/abs/1801.07892 (visited on 05/19/2021).

Yu, Jiahui et al. (Oct. 2019). “Free-Form Image Inpainting with Gated Convolution”. In: *arXiv:1806.03589 [cs]*. arXiv: 1806.03589. URL: http://arxiv.org/abs/1806.03589 (visited on 05/15/2021).

Zhu, Xiangyu et al. (Jan. 2019). “Face Alignment in Full Pose Range: A 3D Total Solution”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41.1. arXiv: 1804.01005, pp. 78–92. ISSN: 0162-8828, 2160-9292, 1939-3539. DOI: 10.1109/TPAMI.2017.2778152. URL: http://arxiv.org/abs/1804.01005 (visited on 05/18/2021).