Analysis of 2D and 3D images of the human head for shape, expression and gaze

Hao Sun

Doctor of Philosophy

University of York

Computer Science

28 April, 2022

Abstract

Analysis of the full human head in the context of computer vision has been an ongoing research area for years. While the deep learning community has witnessed the trend of constructing end-to-end models that solve the problem in one pass, it is challenging to apply such a procedure to full human heads. This is because human heads are complicated and have numerous relatively small components with high-frequency details. For example, in a high-quality 3D scan of a full human head from the Headspace dataset, each ear part only occupies 1.5% of the total vertices. A method that aims to reconstruct full 3D heads in an end-to-end manner is prone to ignoring the detail of ears. Therefore, this thesis focuses on the analysis of small components of the full human head individually but approaches each in an end-to-end training manner. The details of these three main contributions of the three individual parts are presented in three separate chapters. The first contribution aims at reconstructing the underlying 3D ear geometry and colour details given a monocular RGB image and uses the geometry information to initialise a model-fitting process that finds 55 3D ear landmarks on raw 3D head scans. The second contribution employs a similar pipeline but applies it to an eye-region and eyeball model. The work focuses on building a method that has the advantages of both the model-based approach and the appearance-based approach, resulting in an explicit model with state-of-the-art gaze prediction precision. The final work focuses on the separation of the facial identity and the facial expression via learning a disentangled representation. We design an autoencoder that extracts facial identity and facial expression representations separately. Finally, we overview our contributions and the prospects of the future applications that are enabled by them.

Acknowledgement

I would like to give the main thanks to my supervisor, Dr. Nick Pears for his commitments and guidance on my research and career. I would not have completed my PhD without his supervision, insights and extensive knowledge. Nick constantly provides me innovative and helpful advice when I came across problems in my research. Especially when I begin my research in the computer vision area, I merely know anything about computer vision. During that period, Nick provided me with more than enough of patience and guidance to help me get on track and be capable enough to finish my PhD journey. I would also like to give thanks to my internal assessor, Dr. Adrian Bors for his insight questions during all the assessment meetings. I want to give appreciation to my first PhD supervisor, Dr. Daniel Kudenko who taught me a lot during both the first half year of my PhD and my undergraduate final year project. Thanks to all my colleagues and friends, to name a few, Jie Zou, Zongyu Yin, Di Wang, Mao Li, Mark Ferguson and Weicong Luo.

Finally, thanks to my parents and grandparents, I would never go this far without their tremendous support and unconditional love.

Declaration

I declare that this thesis is a presentation of original work and I am the sole author. This work has not previously been presented for an award at this, or any other, University. Main contents have been published in the following papers. For all these works I made the major contribution in design, implementation, experiments and writing. All sources are acknowledged as References.

1. Sun, H.; Pears, N. and Dai, H. (2021). A Human Ear Reconstruction Autoencoder. In Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 5: VISAPP, ISBN 978-989-758-488-6; ISSN 2184-4321, pages 136-145. DOI: 10.5220/0010249901360145

2. Sun, H., Pears, N., and Gu, Y. (2022). Information Bottlenecked Variational Autoencoder for Disentangled 3D Facial Expression Modelling. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (pp. 157-166).

Contents

List of Figures

List of Tables

1 Introduction

Understanding and reconstructing the full human head in the context of computer vision and graphics has been an active research area for decades. The recent advancement of deep learning technologies has enabled a learning methodology that has strong feature extraction abilities, thus deep learning methods have been one of the most popular, efficient and high performance tools for analysis of human-related shapes and structures in computer vision. As a broad and ongoing topic, analysis of the full human head's shape and appearance has numerous unsolved problems that are needed to be addressed. There are a number of components that compose the full human head, including ears, nose, eyes, mouth, face, eyeballs and cranium. As some components of human heads are movable, to represent full heads, the variations introduced by eye gaze, facial expressions and head movement should be taken into consideration. The nature of human heads implies various difficulties for solving the problem. For multiple sub-components, one issue as pointed out by [4, 5] is that methods that learn the full head as a whole may have less variance to express the sub-parts. Capturing additional variations challenges the capacity of the learning model and can significantly reduce the reliability of some methods (*e.g.* facial expression can make the face identification task extra difficult).

The ultimate goal of the topic of analysis of the full head is that given a very high quality scan of a subject's head either in 2D or 3D forms, the algorithm is able to extract semantically meaningful properties and alter those properties. An enormous amount of applications can be enabled by this analysis. These include medical applications - such as ear reconstruction surgery, entertainment - such as face reenactment, forensics, the understanding of human face perception, tiredness detection in autonomous driving, design of wearables - such as VR/AR headsets and spectacles, biometrics - such as 3D face recognition, and so on. To define this in the context of this thesis, we approach analysis by means of analysis-by-synthesis, which aims to understand the raw inputs by reproducing them via models.

To achieve the ultimate goal, we propose a plan in this thesis and demonstrate 3 complete sub-parts in 3 technical chapters. We design the plan to be two-stage, sub-part analysis and part composition. We define sub-part of a human head as a region of a specific component that is either on the surface of a head (*e.g.* ears, nose) or inside the head (*e.g.* eyeballs). Also, a sub-part can be overlap, subset or superset to another sub-part, *e.g.* an eye-region can be defined to contain the face structure around both eyeballs. We design a list of sub-parts, such that the union of the sub-parts forms the complete human head. The stage one is to solve an individual analysis problem for each sub-part for all the human head sub-parts in the list to be given in the next paragraph. The stage two is then to combine each single solved model into the full human head model.

Our proposed first stage contains a list of sub-parts. We categorise the sub-parts into two categories: bigger/coarse category and smaller/fine category. The bigger and coarse sub-parts include full head, face region with neck and cranial region. The smaller and fine sub-parts include ears, eyes, eye region, nose, mouth, hair and facial expression. We can model each single sub-part using the most suitable modelling method for it. For example, we can use model-fitting for ears, deep learning based monocular 3D reconstruction for eye region and non-rigid iterative closest point for full head. Our proposed second stage contains two proposed methods for compositing multiple sub-parts into a full human head model. The first method is to use the individual sub-part model to replace the original part on a full head coarse model. However, such method requires sticking the individual model to the coarse head model, thus we propose a second method that does not require sticking two models. The second method is to upsample and morph the original part on a full head coarse model directly. Note that the original part is extracted so we are still effectively modelling individual sub-parts.

In this thesis, we cover the following sub-parts: ears, eye region with nose, face and facial expressions for the first stage. For ears, we model the right ear and mirror the right ear model to produce a model for the left ear. The model learning process is a deep learning based monocular 3D reconstruction pipeline that learns to find corresponding 3D ear on 2D images. For eye region with nose, we model the eye region, eyeballs and nose in a unified model using the similar method compared to the ear work to produce accurate gaze estimations. Finally, for faces and facial expressions, we model both simultaneously but separately in 3D. For the second stage, we fit our 3D ear model to 3D full head scans. Our eye region model can directly apply to a full head model. Our expression model can be considered as a set of vertex offsets to the original full head model. We leave the modelling of cranial region, eyeballs, nose, mouth and hair parts as future works. Also, for stage two, although we try both composition methods, the final model is still limited in resolution (only maximum 5000 vertices for the full head model).

After drawing the big picture, we now cover more detail about individual analysis problems. The major part of analysis-by-synthesis is that the algorithm that synthesises the raw inputs. The choice of the analysis-by-synthesis method can be diverse. Some traditional methods often use the model-fitting approach. They construct a model to fit the original data, and then extract the desired properties from the constructed model. For example, a model-fitting approach can have a composition of two PCAs to model 3D face geometry and face textures separately. The two PCAs and a lighting model can jointly models the appearance of human faces and produce rendered facial images. The model-fitting process then tries to optimise a set of parameters that controls all three models, such that the finally synthesised facial image looks as similar as the original facial image. However, using the model-fitting process as the 'analysis' method is often time-consuming and the fitted model often cannot predict desired properties in high accuracy. For example, the gaze vectors cannot be accurately obtained with above whole face model-fitting approach in most cases. In the meantime, recent works focus on directly predict desired properties (*i.e.* gaze vectors in this example) accurately by harnessing the raw feature extraction ability of deep neural networks. Given the increasing popularity in these 'direct prediction' approaches, we argue that having an underlying model still has numerous advantages. For example, new data samples can be generated by modifying properties of the model, and other properties (such as skin tones in this example) can be obtained along with the desired properties (*i.e.* gaze vectors in this example). In a result, as proposed by [6], we use a mixture of the two approaches to harness advantages of both. In this case, to replace the model-fitting process, we use deep neural networks to perform the analysis task to predict a set of model parameters, and we use a model to synthesise the original data. Under this setting, we can obtain a fitted model without time-consuming model-fitting process. The obtained model can provide desired properties with comparable accuracy against ‘direct prediction’ approaches and all other useful properties from the model. With modifying certain properties, the obtained model can generate new data samples, too.

For the analysis-by-synthesis methods, the general *autoencoder* architecture implements this idea and enables an end-to-end complete pipeline that performs the analysis-by-synthesis task automatically from data without additional supervision [7]. The architecture contains two parts, the first part is called the *encoder*, which is often a feature extraction module that extracts some compressed latent representation. The second part is the aforementioned generator, also known as the *decoder*, which attempts to synthesise the input data from the latent representation. These two parts form a complete pipeline that takes raw data as inputs and attempts to generate the same data as output, where the output can be compared against the input to improve both system components simultaneously. Deep neural networks naturally form strong feature extraction modules and are usually used as encoders. All the three contributions of our thesis utilise the autoencoder idea and use a state-of-the-art deep neural network as the encoder.

As discussed earlier, most model-based methods have semantically meaningful latent representations that control certain properties, while most deep learning methods cannot naturally generate explainable representations because of the black box property of them. The black box property means that the neural network acts like a black box when it synthesise raw data from a set of features without any semantical meanings, and without unveiling any details of the mapping. A research area that focuses on learning more explainable representation by learning more independent latent variables has become popular recently [8]. Such methods are termed *disentangled representation* learning methods. The disentangled representation means that each of the representation's variable varies one and only one factor of the synthesised data. For example, the width of the face can be controlled by multiple features in a trained deep neural network. An ideally disentangled deep neural network represents the variance of the width of the face using only one feature (*i.e.* one number). This will imply a certain level of explainability and encourage the method to learn a more concise representation [9]. In general, disentangled methods aim to mitigate the black box property of the deep neural networks. With this property, the learned deep learning models can behave more like model-based approaches. We explore this property in Chapter 5 by attempting to separately generate face identity and expression.

In this thesis, we address the often ignored sub-components including ears, eyes, eye-region, eye gaze and facial expressions individually, mainly using the autoencoder architecture, deep learning and model-based methods. We have done three works that aim to separately analyse either single or a closely grouped components. The first work focuses on 3D ears. We start from an augmented ear model that is built from 20 high quality 3D ear scans [10]. The final algorithm consists of a model that can generate in-correspondence 3D ear shapes from 2D ear images in-the-wild. An ear in-the-wild image colour model is also built to colour the 3D ears. We also extend this system to initialise a fitting of 3D ears to 3D full head raw scans. Our second work aims at analysing human eyes and gaze directions by using an eye-region and eyeball model and the same style of training process as the ear autoencoder. Finally, the third work focuses on modelling 3D face identity and 3D facial expressions separately when provided with 3D faces with expressions.

1.1 Outline

The remainder of this thesis is structured as follows:

1. Chapter 2 reviews all of the essential basic concepts for the theoretical groundings of all three works described by this thesis. A number of recent research works are included at the end of each section to give an overview of the current development of the research areas.

2. Chapter 3 contains the first technical contribution, which is a complete 3D human ear reconstruction system that is end-to-end trainable and contains a new colour model. It also demonstrates a further model-fitting application that is initialised with this system.

3. Chapter 4 describes the second technical contribution that models the eye, eye-region and gaze in a similar approach to the previous chapter. We build a hybrid method that uses a novel eye-region model and can provide rich semantically meaningful information, as with model-based methods, and which has excellent gaze prediction accuracy, as with appearance-based methods.

4. Chapter 5 describes the third and final technical contribution, which focuses on 3D face reconstruction with facial expression disentanglement. The method takes a 3D face with expression as input and reconstructs the 3D neutral face and 3D facial expression separately. It uses a simple but effective novel mutual information regulariser to improve disentanglement ability by a large margin.

5. Chapter 6 concludes our contributions, lists applications and limitations, and discusses the prospects for future work.

2 Literature Review

This chapter will cover all the related work for our three contributions. We start with fundamental ideas and methods in each section, then introduce state-of-the-art methods. In the Section 2.1, we introduce the idea of the autoencoder, which covers the basic concepts and the extended works of the variational autoencoder (VAE) [11] and the disentangled VAE. The autoencoder is the cornerstone architecture design choice for the whole thesis to achieve the analysis-by-synthesis pipeline and to enable end-to-end training. The next section, Section 2.2, will cover the fundamentals of the 3D Morphable Models (3DMM) [12], which is a statistical way of modelling 3D objects and forms an important part in the pipelines of all three works presented in this thesis. Finally, in Section 2.3, we show the most relevant recent works and describe their core ideas concisely.

2.1 Autoencoders

Most of the work presented in this thesis is based on an architecture design called an *autoencoder*. It is an unsupervised learning method that learns a compressed representation of a group of data. In this section, the idea of the autoencoder architecture will be elaborated in Section 2.1.1. Principal Component Analysis (PCA) as an autoencoder and one of the autoencoder's most popular variants, Variational Autoencoder (VAE), will be introduced in

Section 2.1.2 and Section 2.1.4 respectively. Meanwhile, the autoencoder has a close relation with deep learning in recent years, and this will be discussed in Section 2.1.3. Finally, the literature on disentangled VAEs will be introduced in Section 2.1.5.

2.1.1 General Interpretation

The autoencoder, firstly proposed by Holyoak *et al.* [7], describes a process that analyses the input data by extracting a latent representation, then syntheses the input data from that extracted latent representation. As depicted in Fig. 2.1, an autoencoder has two main components: encoder and decoder. The latent code is designed to be a latent representation of the input data. Thus, the encoder is a feature extractor that regresses the latent representation from the input data, and the decoder is a generator that syntheses the input data. With appropriate encoders and decoders, the autoencoder can process various types of data. The data types shown in the figures are ear images (top), 3D ear and head meshes or point clouds (bottom).

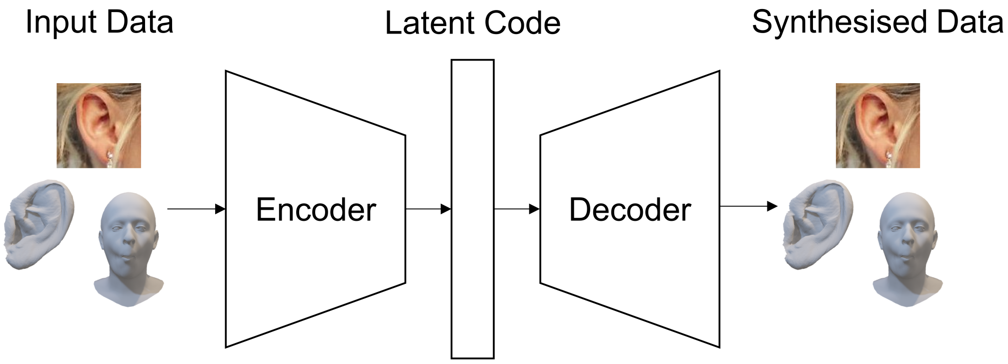


Figure 2.1. General autoencoder architecture.

Baldi [13] formally defines the autoencoder as an architecture that learns the encoder function $Q\colon \mathbb{R}^n \to \mathbb{R}^k$ and the decoder function $P\colon \mathbb{R}^k \to \mathbb{R}^n$

jointly, such that:

$$\arg\min_{Q,P} L\left(P \circ Q\left(\mathbf{X}\right), \mathbf{X}\right), \tag{2.1}$$

where L is the loss function that defines the divergence between original data $\mathbf{X}$ and reconstructed data. An intermediate code vector, *i.e.* latent code $\mathbf{z}$, can be obtained by $\mathbf{z} = Q\left(X\right)$. Such a latent code is a compressed representation of the original data and can be used to recover the original data with the decoder function by $\hat{\mathbf{X}} = P\left(\mathbf{z}\right)$.

2.1.2 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) can be used to construct 3D object models and can be used as a decoder in autoencoder architecture to serve as a linear generator. It is an algorithm that calculates a linear transformation of data points from one coordinate system to the other. The new coordinate system's coordinates have a descending order of variance of their values. The first facial 3D Morphable Model (3DMM) applies PCA to 3D human face images. This will be elaborated in Section 2.2. Also, it can be shown that a linear autoencoder with linear activation functions and squared loss function can be transformed into PCA [14].

This section presents the mathematical details of PCA. PCA can derive the linear transformation in two ways: performing the Singular Value Decomposition (SVD) of the observations or computing the eigendecomposition of the observations' covariance matrix. Practically, we use a Python module named SciPy [15] to calculate the transformation. We expand the SVD approach as it is the same approach taken by SciPy. Assuming a set of N observations each has n features, arranged in a matrix $\mathbf{X} \in \mathbb{R}^{N\times n}$, the observations are zero-meaned initially:

$$\mathbf{X}' = \mathbf{X} - \bar{\mathbf{X}}, \tag{2.2}$$

where $\bar{\mathbf{X}} \in \mathbb{R}^n$ is the mean data point of all data samples. The zero-mean data matrix $\mathbf{X}'$ is then factorised by SVD as:

$$\mathbf{X}' = \mathbf{U}\mathbf{S}\mathbf{V}^T, \tag{2.3}$$

where $\mathbf{U} \in \mathbb{R}^{N\times n}$, $\mathbf{S} \in \mathbb{R}^{n\times n}$ and $\mathbf{V} \in \mathbb{R}^{n\times n}$, such that $\mathbf{US}$ are principal components and $\mathbf{V}^T$ are principal axes. Assume it is desired for a dimension reduction to k dimensions ($k < n$), the principal axes can be truncated to first k rows, end up in $\mathbf{W} \in \mathbb{R}^{k\times n}$. Then to reconstruct the original data from any principal components *i.e.* latent code $\mathbf{z} \in \mathbb{R}^{k\times 1}$, the linear transformation can be formed as:

$$\hat{\mathbf{X}} = \bar{\mathbf{X}} + \mathbf{z}^T\mathbf{W}. \tag{2.4}$$

For extracting the latent code of a new observation $\mathbf{X}_t$, the inverse of the PCA linear transformation in Eq. (2.4) can be applied as:

$$\hat{\mathbf{z}} = \mathbf{W}^{-1}\left(\mathbf{X}_t - \bar{\mathbf{X}}\right). \tag{2.5}$$

Therefore, Eq. (2.5) and Eq. (2.4) forms an encoder and a decoder respectively, then jointly form an autoencoder with latent code $\mathbf{z}$ as the contracted representation with dimension k. It is also effectively a dimension reduction transformation that reduces the dimension from n to k.

2.1.3 Autoencoder and Deep Learning

The idea of the autoencoder has an inseparable relation with deep learning in recent years [8]. When the autoencoder was firstly proposed, both the encoder and the decoder were implemented with neural networks. When deep learning introduces neural networks that have much more capacity, it is natural to apply various deep models to form different autoencoders for different purposes. Meanwhile, it is possible to use a fixed model-based decoder instead of a learnable decoder to constrain the learning process and to improve the trained encoder's performance [6].

In the 3D deep learning context, a large number of specially-designed deep neural networks have been proposed for handling various input types. For point cloud data, the most notable encoder networks are PointNet [16] and point transformers [17, 18]. For the point cloud decoders, multilayer perceptrons are the most used networks to produce a fixed-order point set [19, 20]. For mesh data, the literature focuses on exploiting the topology information, which jointly defines an undirected graph with the point set. Thus, Graph Neural Networks (GNNs) are applied for both encoding and decoding [21, 22, 23, 24]. For volume data, the canonical approach is to treat them as 3D images and use 3D convolution layers to extract features and 3D deconvolution layers to reconstruct volumes. A notable approach for medical 3D images is V-Net [25], which is an extension to the 2D version U-Net [26]. They apply residual connections between layers from the encoder and the decoder to provide better context and yield better performance with less training data. Meanwhile, due to the curse of dimensionality, networks that process volume data directly can only handle volumes with very limited resolutions [27, 28, 29]. Thus, Park *et al.* [30] propose to use an encoder-less structure and represent the shape of objects using a neural network predicted signed distance field.

Amid all the autoencoder-based 3D deep learning approaches described above, there is one unique kind that processes and reconstructs 2D images but provides strong 3D assumptions in intermediate steps. Our work presented in Chapter 3 and Chapter 4 are based on such an architecture. Tewari *et al.* [6] firstly propose this work (named MoFA), which uses a fixed model-based decoder that differentiably produces a 2D image given the latent code. This work has been adopted and extended by many other works [31, 32].

2.1.4 Variational Autoencoder (VAE)

Numerous variations to the autoencoder architecture have been proposed over the last few years, including Sparse Autoencoder that aims to learn a sparse latent code [33]; Denoising Autoencoder that aims to build a regularised autoencoder for denoising purpose [34]; Contractive Autoencoder that adds additional constraints to further improve robustness to noise and outliers [35] and Variational Autoencoder that constrains the latent code to follow predefined distribution such that sampling the latent code space becomes easier [11]. In the following, we closely follow the mathematical development from the original paper [11] and the tutorial [36]. VAE is used in Chapter 5 as a basic deep learning pipeline for learning 3D faces and facial expressions.

A VAE starts by assuming the observed data $\mathbf{X}$ to be generated by an unobserved random variable $\mathbf{z}$. Given that we are interested in learning the likelihood of the observed data $p\left(X\right)$, we can rearrange the term as:

$$p\left(X\right) = \int p\left(\mathbf{X}, \mathbf{z}\right)\, d\mathbf{z} \tag{2.6}$$

$$= \int p\left(\mathbf{X} \mid \mathbf{z}\right) p\left(\mathbf{z}\right)\, d\mathbf{z}. \tag{2.7}$$

Following the autoencoder design presented earlier, we use a neural network decoder to approximate the conditional probability distribution, as:

$$p\left(\mathbf{X} \mid \mathbf{z}, \boldsymbol{\theta}\right), \tag{2.8}$$

where $\boldsymbol{\theta}$ are the weights of the decoder neural network, and this forward calculation of the neural network is denoted as p_θ to avoid notational clutter. As for the design of the latent code $\mathbf{z}$, each latent variable is desired to be independent of each other to better represent each *factor* that generates the observed data $\mathbf{X}$. A VAE [11] proposes to solve this issue by assuming the latent code's prior as a unit isotropic Gaussian distribution:

$$p\left(\mathbf{z}\right) = \mathcal{N}\left(\mathbf{0}, \mathbf{I}\right), \tag{2.9}$$

where $\mathbf{I}$ is an identity matrix. Given Eq. (2.7), our goal for the decoder to learn optimal weights $\boldsymbol{\theta}^*$ can then be arranged in the Maximum-a-posteriori (MAP) fashion as:

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \int p_\theta\left(\mathbf{X} \mid \mathbf{z}\right) p\left(\mathbf{z}\right)\, d\mathbf{z}. \tag{2.10}$$

Then, the difficulty left for this learning model is that analytically calculating the integral term is not practically feasible. Given the intuition that $p\left(\mathbf{X} \mid \mathbf{z}\right)$ is nearly zero for most of the $\mathbf{z}$ since $\mathbf{X}$ has a very high dimension. VAE [11] proposes to learn another distribution $q_\phi(\mathbf{z} \mid \mathbf{X})$ with neural network of weights $\boldsymbol{\phi}$ to approximate a $\mathbf{z}$ given every observed data sample. Ideally, the distribution q_ϕ can approximate the intractable posterior distribution $p(\mathbf{z} \mid \mathbf{X})$ given high-capacity q_ϕ. We denote this distribution as encoder. Given the encoder distribution, we can now calculate the expectation of the log-likelihood of the decoder distribution $E_{\mathbf{z}\sim q_\phi}[\log p_\theta(\mathbf{X} \mid \mathbf{z})]$.

Finally, as a cornerstone of the VAE, relating the expectation term and the data log-likelihood $\log p(\mathbf{X})$ bypasses the calculation of the intractable integral term and enables an end-to-end trainable model by backpropagation. Start with the log-likelihood of the data distribution:

$$\begin{aligned}
\log p(\mathbf{X}) &= E_{\mathbf{z}\sim q_\phi(\mathbf{z}\mid\mathbf{X})}[\log p(\mathbf{X})] \quad (p(\mathbf{X}) \text{ does not depend on } \mathbf{z}) && (2.11)\\
&= E_{\mathbf{z}}\left[\log \frac{p_\theta(\mathbf{X} \mid \mathbf{z})\, p(\mathbf{z})}{p(\mathbf{z} \mid \mathbf{X})}\right] && (2.12)\\
&= E_{\mathbf{z}}\left[\log\left(\frac{p_\theta(\mathbf{X} \mid \mathbf{z})\, p(\mathbf{z})}{p(\mathbf{z} \mid \mathbf{X})} \frac{q_\phi(\mathbf{z} \mid \mathbf{X})}{q_\phi(\mathbf{z} \mid \mathbf{X})}\right)\right] && (2.13)\\
&= E_{\mathbf{z}}\left[\log p_\theta(\mathbf{X} \mid \mathbf{z}) + \log \frac{p(\mathbf{z})}{q_\phi(\mathbf{z} \mid \mathbf{X})} + \log \frac{q_\phi(\mathbf{z} \mid \mathbf{X})}{p(\mathbf{z} \mid \mathbf{X})}\right] && (2.14)\\
&= E_{\mathbf{z}}\left[\log p_\theta(\mathbf{X} \mid \mathbf{z}) - \log \frac{q_\phi(\mathbf{z} \mid \mathbf{X})}{p(\mathbf{z})} + \log \frac{q_\phi(\mathbf{z} \mid \mathbf{X})}{p(\mathbf{z} \mid \mathbf{X})}\right] && (2.15)\\
&= E_{\mathbf{z}}[\log p_\theta(\mathbf{X} \mid \mathbf{z})] - E_{\mathbf{z}}\left[\log \frac{q_\phi(\mathbf{z} \mid \mathbf{X})}{p(\mathbf{z})}\right] + E_{\mathbf{z}}\left[\log \frac{q_\phi(\mathbf{z} \mid \mathbf{X})}{p(\mathbf{z} \mid \mathbf{X})}\right] && (2.16)\\
&= E_{\mathbf{z}}[\log p_\theta(\mathbf{X} \mid \mathbf{z})] - \mathrm{KL}(q_\phi(\mathbf{z} \mid \mathbf{X}) \parallel p(\mathbf{z}))\\
&\qquad + \mathrm{KL}(q_\phi(\mathbf{z} \mid \mathbf{X}) \parallel p(\mathbf{z} \mid \mathbf{X})), && (2.17)
\end{aligned}$$

where $\mathrm{KL}(P(y) \parallel Q(y))$ is the Kullback–Leibler (KL) divergence between distributions P and Q, defined as:

$$\mathrm{KL}(P(y) \parallel Q(y)) = E_{y\sim P}\left[\log \frac{P(y)}{Q(y)}\right]. \tag{2.18}$$

After the above derivations, it ends up in three terms, and the data log-likelihood $\log p(\mathbf{X})$ is related to the expectation $E_{\mathbf{z}\sim q_\phi}[\log p_\theta(\mathbf{X} \mid \mathbf{z})]$. Our objective is then:

$$\boldsymbol{\theta}^*, \boldsymbol{\phi}^* = \underset{\boldsymbol{\theta},\boldsymbol{\phi}}{\arg\max} \log p(\mathbf{X}). \tag{2.19}$$

However, there is still an intractable distribution $p(\mathbf{z} \mid \mathbf{X})$ in the equation. Since all KL divergences are greater than or equal to zero, we eliminate the second KL divergence term that contains the intractable distribution, and optimise a lower bound for the data log-likelihood instead. However, in the ideal situation that the encoder distribution q_ϕ is close enough to the true posterior $p(\mathbf{z} \mid \mathbf{X})$, the second KL divergence term is nearly zero, we are effectively optimising the data log-likelihood directly again. The remaining two terms can be further interpreted. The expectation term $E_{\mathbf{z}}[\log p_\theta(\mathbf{X} \mid \mathbf{z})]$ can be seen as the reconstruction objective as it maximises the likelihood of the generated data. The KL term $\mathrm{KL}(q_\phi(\mathbf{z} \mid \mathbf{X}) \parallel p(\mathbf{z}))$ makes the encoder distribution similar to its prior.

To fit the above objective in the neural network training context, several amendments have to be applied. Firstly, most of the modern deep networks are trained using the backpropagation algorithm, which implies that the whole forward process has to be differentiable. Our encoder network produces a probability distribution of the latent code $\mathbf{z}$ instead of the latent code directly. Thus, the reparameterisation trick [11, 37] is applied to get $\mathbf{z}$ in a differentiable manner. The Gaussian distribution version is demonstrated. Let $\mathbf{z} \sim q_\phi(\mathbf{z} \mid \mathbf{X}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$,

$$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma}\boldsymbol{\epsilon}, \quad \text{where } \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \tag{2.20}$$

Secondly, most deep networks are trained with gradient descent, which implies the goal has to be a loss function to be minimised. We can then derive the Evidence Lower Bound (ELBO) function:

$$\mathrm{ELBO} = -E_{\mathbf{z}}\left[\log p_{\theta}\left(\mathbf{X} \mid \mathbf{z}\right)\right] + \mathrm{KL}\left(q_{\phi}\left(\mathbf{z} \mid \mathbf{X}\right) \parallel p\left(\mathbf{z}\right)\right). \tag{2.21}$$

Thirdly, the commonly used training scheme, minibatch gradient descent, allows a small subset of the data to be trained on each pass. That implies inefficiency in calculating the full expectation term. Thus, the current data and corresponding $\mathbf{z}$ are used to estimate the expectation term. It can then be simplified to $\log p_{\theta}\left(\mathbf{X} \mid \mathbf{z}\right)$. This is reasonable since minibatch sampling is applied already. In practice, this term is further simplified to a deterministic version, *i.e.* predict $\hat{\mathbf{X}}$ directly. Then the commonly used loss functions by deep learning, *e.g.* L1 loss, mean squared error loss, can be directly used. The KL term remains unmodified.

2.1.5 Disentangled VAE

As a specialisation of the variational autoencoder, the disentangled VAE aims to learn a more structured and more concise latent code, such that each latent variable aims to represents a distinctive feature of both the input data and the generated data. We formally define the terminology *disentanglement* first, then introduce a number of important papers in this field and finally layout the mathematical background for the disentangled VAE. A basic assumption made by all the disentangled VAEs introduced in this section is that no labels related to each factor are provided, thus the learning completes in an unsupervised manner.

With the purpose of learning a better representation, disentanglement is defined as a desired property for each learned latent unit to represent the variance of a single generative factor and be irrelevant to other latent units [38]. The research on disentangled autoencoders can be dated back to 1992 by Schmidhuber [39]. Lots of recent works focus on learning a disentangled representation too. For example, Desjardins *et al.* [40] propose to use a variant of a Boltzmann machine for disentanglement. Makhzani *et al.* propose an adversarial autoencoder [41] and a PixelGAN autoencoder [42], both target at learning a disentangled latent representation via focusing on more about the relation between the representation and the generated data. In the meantime, due to the increasing popularity of the Generative Adversarial Network (GAN), some disentanglement works focus on building a disentangled generative model via GAN [43, 44]. Some ideas that are applied to GANs are also applicable to VAEs since they are two related approaches to the generative model.

To expand more on disentanglement in the context of the VAE, we start from a simple yet very effective method, β-VAE [45], followed by some further works that are inspired by this paper and aim at building a better objective function by decomposition of the original weighted ELBO loss [9, 46]. The weighted ELBO introduced by the β-VAE method is shown as follows:

$$\beta\text{-VAE-ELBO} = -E_{\mathbf{z}}\left[\log p_\theta\left(\mathbf{X} \mid \mathbf{z}\right)\right] + \beta \mathrm{KL}\left(q_\phi\left(\mathbf{z} \mid \mathbf{X}\right) \parallel p\left(\mathbf{z}\right)\right), \tag{2.22}$$

where the hyperparameter β is a factor multiplied by the KL loss function and balances the trade-off between latent code disentanglement with reconstruction quality and latent code capacity. In the β-VAE context, β is always set to a real number that is greater than 1 to add importance to the KL term during the network training process. This will result in a better disentangled latent code and compromised reconstruction accuracy. Further works aim to ameliorate the trade-off by decomposition of the KL loss function and applying weights on individual decomposed terms to achieve the same disentanglement quality with less compromising on reconstruction quality. The authors of β-VAE argue that the added scalar β is a Lagrangian multiplier that is under the KKT condition [47, 48]. They also show that the KL loss term encourages more conditional independence in the encoder conditional distribution $q_\phi(\mathbf{z} \mid \mathbf{X})$. Jointly with the reconstruction loss, applying a β that is greater than one can lead to a more efficient representation. Given the assumption that the objects are generated given some factors that have a lower dimension than the latent code, the authors hypothesised that the representation learned under such a setting has a better quality of disentanglement.

Based on the analysis made by β-VAE, two further works aim to decompose the KL term and encourage the method to have more conditional independence in the encoder distribution solely [9, 46]. Kim and Mnih [9] propose the FactorVAE to optimise a latent code independence directly, and Chen *et al.* [46] propose the β-TCVAE to decompose the KL loss into three terms to apply weights individually. They also decompose the KL loss in two different ways. Both are illustrated in this section.

FactorVAE decomposes the KL loss into two terms: i) the mutual information between the input data and the latent code, and ii) the KL divergence between the latent code distribution and its prior distribution. The detailed decomposition [42] of the expectation of the KL term *w.r.t.* observed data distribution is shown as follows:

$$
\begin{aligned}
& E_{p_{data}(x)}\left[\mathrm{KL}\left(q_\phi\left(\mathbf{z} \mid \mathbf{X}\right) \| p\left(\mathbf{z}\right)\right)\right] && (2.23)\\
&= E_{p_{data}(x)}\left[E_{q_\phi(\mathbf{z}|\mathbf{X})}\left[\log \frac{q_\phi\left(\mathbf{z} \mid \mathbf{X}\right)}{p\left(\mathbf{z}\right)}\right]\right] \qquad \textit{by definition} && (2.24)\\
&= E_{p_{data}(x)}\left[E_{q_\phi(\mathbf{z}|\mathbf{X})}\left[\log \frac{q_\phi\left(\mathbf{z} \mid \mathbf{X}\right)}{q_\phi\left(\mathbf{z}\right)} \frac{q_\phi\left(\mathbf{z}\right)}{p\left(\mathbf{z}\right)}\right]\right] && (2.25)\\
&= E_{p_{data}(x)}\left[E_{q_\phi(\mathbf{z}|\mathbf{X})}\left[\log \frac{q_\phi\left(\mathbf{z} \mid \mathbf{X}\right)}{q_\phi\left(\mathbf{z}\right)} + \log \frac{q_\phi\left(\mathbf{z}\right)}{p\left(\mathbf{z}\right)}\right]\right] && (2.26)\\
&= E_{p_{data}(x)}\left[\mathrm{KL}\left(q_\phi\left(\mathbf{z} \mid \mathbf{X}\right) \| q_\phi\left(\mathbf{z}\right)\right)\right] + \\
&\quad E_{p_{data}(x)}\left[E_{q_\phi(z)}\left[\log \frac{q_\phi\left(\mathbf{z}\right)}{p\left(\mathbf{z}\right)}\right]\right] && (2.27)\\
&= I_{q_\phi}\left(\mathbf{X};\mathbf{z}\right) + E_{p_{data}(x)}\left[E_{q_\phi(\mathbf{z})}\left[\log \frac{q_\phi\left(\mathbf{z}\right)}{p\left(\mathbf{z}\right)}\right]\right] && (2.28)\\
&= I_{q_\phi}\left(\mathbf{X};\mathbf{z}\right) + E_{q_\phi(\mathbf{z},\mathbf{X})}\left[\log \frac{q_\phi\left(\mathbf{z}\right)}{p\left(\mathbf{z}\right)}\right] && (2.29)\\
&= I_{q_\phi}\left(\mathbf{X};\mathbf{z}\right) + E_{q_\phi(\mathbf{z})}\left[\frac{q_\phi\left(\mathbf{z}\right)}{p\left(\mathbf{z}\right)}\right] && (2.30)\\
&= I_{q_\phi}\left(\mathbf{X};\mathbf{z}\right) + \mathrm{KL}\left(q_\phi\left(\mathbf{z}\right) \| p\left(\mathbf{z}\right)\right). && (2.31)
\end{aligned}
$$

With this decomposition, the expectation of the KL term over all the observed data is then divided into two terms: the mutual information term and the KL term. Mutual information can be interpreted as the common information contained in both random variables. The term in this equation represents the common information between observed data distribution and the latent code distribution predicted by the encoder network. The second term represents the KL divergence between the aggregated posterior distribution of the encoder and the predefined latent code prior. The term *aggregated posterior distribution* is defined as in [41]. After this is incorporated into the loss function, the first term penalises the amount of information contained in the latent code about the observed data, therefore worsening the reconstruction quality if too much weight is put on this term [42]. Meanwhile, the second term pushes the encoder distribution towards factor distribution. This makes the individual latent code independent of each other, thus achieving disentanglement [9]. However, in β-VAE, the added weighting β penalises both terms at the same time. Thus, Kim and Mnih [9] propose to penalise the second term solely by adding an additional loss function to the original KL loss function. Since the calculation of the aggregated posterior distribution of the encoder $q_\phi(\mathbf{z})$ requires going through the whole dataset and is practically inapplicable, the authors use the density ratio trick [49, 50] to mitigate this issue. This algorithm trains a classifier/discriminator neural network D to learn the distribution over the training time of the main VAE. The added loss function component directly predicts a loss of the total correlation of individual latent code as:

$$\begin{aligned} \mathrm{TC}(\mathbf{z}) &= \mathrm{KL}\left(q_\phi(\mathbf{z}) \parallel \bar{q}_\phi(\mathbf{z})\right) && (2.32) \\ &= E_{q_\phi(\mathbf{z})}\left[\frac{q_\phi(\mathbf{z})}{\bar{q}_\phi(\mathbf{z})}\right] && (2.33) \\ &\approx E_{q_\phi(\mathbf{z})}\left[\log \frac{D(\mathbf{z})}{1 - D(\mathbf{z})}\right], && (2.34) \end{aligned}$$

where the discriminator network D is trained to predict whether the input latent code is a sample from q_ϕ rather than $\bar{q}_\phi$.

There are some disputes about whether penalising the mutual information term is helpful for the overall disentanglement since it can encourage learning a more concise latent code [43, 46, 51]. We find that penalising the term can lead to a more concise latent code and is especially useful under the situation where multiple observations are mapped to a single latent code [51]. Details are in Chapter 5.

The other work, β-TCVAE [46] decomposes the KL term in a different way, where three terms are decomposed as i) index-code mutual information; ii) total correlation, and iii) dimension-wise KL. Authors additionally assume that an integer $n \in \{1 \cdots N\}$ is randomly assigned to each observed data point from N total observations. Further on that, the authors define that:

$$q_\phi(\mathbf{z} \mid n) = q_\phi(\mathbf{z} \mid \mathbf{X}_n) \tag{2.35}$$

$$\begin{aligned} q_\phi(\mathbf{z}, n) &= q_\phi(\mathbf{z} \mid n)\, p(n) && (2.36) \\ &= \frac{1}{N} q_\phi(\mathbf{z} \mid n). && (2.37) \end{aligned}$$

The decomposition of the expectation of the KL term *w.r.t.* data distribution from the ELBO loss function is then shown as follows:

$$E_{p(n)}\left[\mathrm{KL}\left(q_\phi(\mathbf{z} \mid \mathbf{X}) \parallel p(\mathbf{z})\right)\right] \tag{2.38}$$

$$= E_{p(n)}\left[E_{q_\phi(\mathbf{z}|n)}\left[\log \frac{q_\phi(\mathbf{z} \mid n)}{p(\mathbf{z})}\right]\right] \tag{2.39}$$

$$= E_{q_\phi(\mathbf{z},n)}\left[\log q_\phi(\mathbf{z} \mid n) - \log p(\mathbf{z}) + \log \frac{q_\phi(\mathbf{z})}{q_\phi(\mathbf{z})} + \log \prod_j q_\phi(\mathbf{z}_j) - \log \prod_j q_\phi(\mathbf{z}_j)\right] \tag{2.40}$$

$$= E_{q_\phi(\mathbf{z},n)}\left[\log \frac{q_\phi(\mathbf{z} \mid n)}{q_\phi(\mathbf{z})}\right] + E_{q_\phi(\mathbf{z})}\left[\log \frac{q_\phi(\mathbf{z})}{\prod_j q_\phi(\mathbf{z}_j)}\right] + E_{q_\phi(\mathbf{z})}\left[\sum_j \log \frac{q_\phi(\mathbf{z}_j)}{p(\mathbf{z}_j)}\right] \tag{2.41}$$

$$= E_{q_\phi(\mathbf{z},n)}\left[\log \frac{q_\phi(\mathbf{z}, n)}{q_\phi(\mathbf{z})\, p(n)}\right] + E_{q_\phi(\mathbf{z})}\left[\log \frac{q_\phi(\mathbf{z})}{\prod_j q_\phi(\mathbf{z}_j)}\right] + E_{q_\phi(\mathbf{z})}\left[\sum_j \log \frac{q_\phi(\mathbf{z}_j)}{p(\mathbf{z}_j)}\right] \tag{2.42}$$

$$= I_{q_\phi}(\mathbf{z}; n) + \mathrm{KL}(q_\phi(\mathbf{z}) \parallel \prod_j q_\phi(\mathbf{z}_j)) + \sum_j \mathrm{KL}\left(q_\phi(\mathbf{z}_j) \parallel p(\mathbf{z}_j)\right). \tag{2.43}$$

The first mutual information term I_{q_ϕ} is identical to the mutual information term in Eq. (2.31), which represents how much data information is stored in the latent code. The first KL term is the same as the added loss function in FactorVAE in Eq. (2.32), which is a total correlation term. Both FactorVAE and β-TCVAE argue that putting more weights on this term encourages independence among each variable of the latent code thus is more effective in achieving disentanglement. The second KL term is a dimension-wise KL which sums the KL divergence between each latent variable and its factor prior. This loss prevents individual latent variables' distributions from diverging away from the assumed prior distributions.

The β-TCVAE also has one difficulty of calculating the aggregated posterior distribution $q_\phi(\mathbf{z})$. A recent work takes another approach to calculate this term, namely *Minibatch-Weighted Sampling* [46]. Let $B_M = \{n_1, \cdots, n_M\}$ be a size-M minibatch. Then $p(B_M) = (1/N)^M$. Denoting $r(\mathbf{X}_M \mid n)$ as the probability of a minibatch sample that given a datapoint n, the rest are sampled from $p(n)$. The method aims to obtain the expectation of the aggregated posterior distribution by:

$$
\begin{aligned}
& E_{q_\phi(\mathbf{z})}\left[\log q_\phi(\mathbf{z})\right] && (2.44)\\
=& E_{q_\phi(\mathbf{z},n)}\left[\log E_{n'\sim p(n)}\left[q_\phi\left(q_\phi\left(\mathbf{z} \mid n'\right)\right)\right]\right] && (2.45)\\
=& E_{q_\phi(\mathbf{z},n)}\left[\log E_{p(B_M)}\left[\frac{1}{M}\sum_{m=1}^{M} q_\phi\left(\mathbf{z} \mid n_m\right)\right]\right] && (2.46)\\
\geq& E_{q_\phi(\mathbf{z})}\left[\log E_{r(B_M|n)}\left[\frac{p\left(B_M\right)}{r\left(B_M \mid n\right)}\frac{1}{M}\sum_{m=1}^{M} q_\phi\left(\mathbf{z}|n_m\right)\right]\right] && (2.47)\\
=& E_{q_\phi(\mathbf{z})}\left[\log E_{r(B_M|n)}\left[\frac{1}{NM}\sum_{m=1}^{M} q_\phi\left(\mathbf{z} \mid n_m\right)\right]\right] && (2.48)\\
=& \frac{1}{M}\sum_{i=1}^{M}\left[\log\sum_{j=1}^{M} q_\phi\left(\mathbf{z}\left(n_i\right) \mid n_j\right) - \log\left(NM\right)\right]. && (2.49)
\end{aligned}
$$

With this sampling-based estimator, all three terms can be calculated individually.

To sum up the relations between the two disentangled VAEs, both of them decompose the KL loss function into smaller pieces and weigh them differently. Both put a great emphasis on the total correlation term and argue that the term is the key to disentanglement. Our work in Chapter 5 is inspired by this decomposition and uses the mutual information term between the input and the latent code as an additional regulariser in order to disentangle face identity and facial expression. We also use the minibatch weighted sampling technique to estimate our regulariser.

2.1.6 VAE Information-based Methods

There are a number of works that implement information-theoretic ideas into their VAE-based architecture; for example, the Variational Information Bottleneck (VIB) proposed by Alemi *et al.* [52]. Starting from the information bottleneck idea firstly proposed by Tishby *et al.* [53], Alemi *et al.* propose to optimise the information bottleneck using deep neural networks. This results in a similar autoencoder architecture to the VAE.

The information bottleneck, an idea that has lots of connections with VAE, has an objective function formulated as follows:

$$J_{IB} = \underset{p(\mathbf{z}|\mathbf{X})}{\arg\min} \left(I\left(\mathbf{X} \mid \mathbf{z}\right) - \beta I\left(\mathbf{z} \mid \mathbf{X}\right) \right), \tag{2.50}$$

where the first and the second mutual information term aim to achieve better accuracy and better compression respectively. The scalar β is a Lagrangian multiplier that balances the two mutual information term. Minimising them jointly aims to find the perfect balance of accuracy and compression for the model that estimates $p(\mathbf{z} \mid \mathbf{X})$.

One of the main terms in information theory related to VAEs is the mutual information between the input and the latent code. However, analytically calculating this term requires a forward pass of the encoder network on the entire dataset for every backpropagation. This is undesirable since it can increase the training time prohibitively. InfoGAN [43] uses Monte Carlo simulation to estimate a lower bound for the mutual information term directly. Kim *et al.* [9] and Zhang *et al.* [54] use the density ratio trick [49, 50] by introducing a discriminator. InfoVAE [55] obtains unbiased samples of the latent code by running a forward pass of decoder and encoder [56], and proposes to use other divergences such as the Jensen-Shannon divergence. In our work, we use Minibatch Weighted Sampling (MWS), used in beta-TCVAE [46], to obtain a direct estimate of the aggregated prior without additional hyper-parameters or networks.

2.2 3D Morphable Model (3DMM)

The 3D Morphable Model (3DMM) was firstly proposed by Blanz and Vetter [12] to build a statistical model for 200 registered 3D face meshes and their textures. The idea of a linear 3DMM that is built with Principal Component Analysis (PCA) will be discussed in this section. A review of the face and eye 3DMMs (*e.g.* multilinear 3DMM [57]) will be discussed at the end of the section.

The idea of the 3DMM is to use significantly fewer parameters to represent the variation in human faces. The initial approach uses a PCA to perform dimension reduction on the registered 3D face vertices where all vertices of two 3D faces are paired. The reconstruction is then the inverse transform process of the PCA. Let $\mathbf{S} \in \mathbb{R}^{3N\times 1}$ be the 3D face point cloud with N vertices, and $\mathbf{T} \in \mathbb{R}^{3N\times 1}$ be the colour for each vertex in S. Denote the mean of the vertices and the colours as $\bar{\mathbf{S}}$ and $\bar{\mathbf{T}}$. The original paper presents the reconstruction as:

$$\mathbf{S} = \bar{\mathbf{S}} + \mathbf{U}_s\boldsymbol{\alpha}_s \tag{2.51}$$

$$\mathbf{T} = \bar{\mathbf{T}} + \mathbf{U}_t\boldsymbol{\alpha}_t, \tag{2.52}$$

where $\boldsymbol{\alpha}_s \in \mathbb{R}^{M_s\times 1}$ and $\boldsymbol{\alpha}_t \in \mathbb{R}^{M_t\times 1}$ are M_s and M_t parameters for shape and texture respectively. Also, $\mathbf{U}_s \in \mathbb{R}^{3N\times M}$ and $\mathbf{U}_t \in \mathbb{R}^{3N\times M}$ are the learned principal components generated by the PCA algorithm.

2.2.1 Face and Eyes 3D Morphable Model (3DMM)

Face 3DMMs were introduced more than two decades ago by Blanz and Vetter [12] and perhaps is the most widely employed technique in recent statistical 3D face modelling applications. Such 3DMMs model a linear or non-linear 3D facial space using a latent representation that can be constructed in a number of different ways. Examples include PCA [3, 58, 59, 60, 61], dictionary learning [62], wavelet decomposition [63], Gaussian mixture models [64] and neural nets [31]. Apart from general face 3DMMs, there are several approaches that bring more focus to eyeball modelling. Bérard *et al.* were the first to build a parametric model of eyeballs. The quality of this eye model is high, but the reconstruction process is semi-automatic. Wood *et al.* [65, 66] attempt to build an eye-region model of the single eye and use model fitting to estimate gaze. Ploumpis *et al.* [61] propose a method for building a complete head morphable model that includes eyeballs. The eye-region modelling is similar to the approach of Wood *et al.* and is blended into the head model.

Amongst the publicly-available 3DMMs, we choose the FLAME [3] model to build our eye-region model since it has both eyeballs and can form a minimal eye-region model for both eyes. It is PCA-based, and it reconstructs faces via linear combinations. Thus, it does not require additional training and is both fast and stable in the training process. We use a masked FLAME 3DMM to form the model-based decoder to reconstruct 3D eye-region meshes.

2.3 End-to-end 3D Human-related Object Reconstruction

2.3.1 2D-to-2D Face Reconstruction

Numerous reconstruction methods have been proposed, and they generally fall into three categories: generative, regression and generative-regression hybrid. Generative methods focus on generating a 3D model to fit the target data [67, 68, 69, 70, 71]. The approaches proposed by Wood *et al.* [65] and Ploumpis *et al.* [61] both fall into this category. Regression methods, recently popular due to deep learning advances, focus on regressing the model parameters directly via deep networks [72, 73, 74, 75, 76, 77]. The third category was firstly proposed by Tewari *et al.* [6], and adopted by many other works [31, 21, 78, 32]. This approach usually trains a joint autoencoder model that encodes the model parameters via the regression method, decodes the regressed model parameters, and reconstructs the original input. Finally, the reconstructed output is compared with the input data to form an autoencoder end-to-end learning model. Specifically, if the input data are in the format of an image, a differentiable renderer is employed to render images from the reconstructed 3D meshes while maintaining an end-to-end trainable network. All of the mentioned face autoencoders focus on full-face reconstruction and use only a mesh surface to model the eyeball, and the appearance of different gaze directions is not present or modelled via texture.

2.3.2 In-the-wild Ear Image Dataset

Numerous in-the-wild ear image datasets are built for various purposes. Here we focus on Collection A from the *In-the-wild Ear Database* (ITWE-A) since it has 55 manually-marked landmarks. All the landmarks have semantic meaning, as shown in Figure 3.1 (1). This dataset contains 500 images in its training set and 105 images in its test set, where each image is captured in the wild and contains a clear ear. The dataset has a large variation in-ear colours, as is the nature of in-the-wild images, and it even contains several grayscale images. Traditional 3DMM colour models, such as that of the Basel Face Model 09 (BFM09) [12], often fail to generate a highly-similar appearance to the input. However, the in-the-wild ear colour model proposed here can cover such colour variance since it models directly from the in-the-wild images themselves.

2.3.3 Parametric Ear Models

An Active Appearance Model (AAM), a parametric ear model built by Zhou and Zaferiou, is a linear model that aims to model the 2D ear's shape and colour simultaneously [79]. A 3D Morphable Model (3DMM) is a closely-related model that models objects' shapes and colours in 3D instead of 2D. Blanz and Vetter first propose a 3D Morphable Model (3DMM) for human faces [12], which builds a linear system that allows different 3D face meshes to be described by 199 shape parameters. Similarly, Dai *et al.* [80] propose a

3D morphable model for the human ear named the York Ear Model (YEM), also based on a linear system but with 499 parameters. Here, we utilise this ear 3DMM for its strong 3D ear shape prior. Meanwhile, the reduced dimension of the parameters allows the neural network to perform a much easier regression task using 499 shape parameters rather than 21333 raw vertex parameters.

2.3.4 2D Ear Detection

Ear detection or localisation in 2D images is a task to find the region of interest bounding the ear from images of the human head that contain ears, for example, profile-view portraits. It is a vital preprocessing step in the 3D ear reconstruction pipeline. Object detection has been studied for decades, and there exists a number of algorithms that specifically perform the 2D ear detection task. Zhou and Zaferiou [1] use the histogram of oriented gradients with a support vector machine (HoG+SVM) to predict a rectangular region of interest. Emeršič *et al.* [81] and Bizjak *et al.* [82] propose deep learning methods to tackle the 2D ear detection task by predicting a pixel-level segmentation of the 2D ear image directly.

2.3.5 2D Ear Landmark Localisation

2D ear landmark localisation aims to find specific key points on 2D ear images. It is an intuitive method of quantitative evaluation of this work where the shape and alignment of the reconstructed 3D ear mesh can be evaluated precisely. In 2D face landmark localisation, numerous approaches obtain 2D landmarks by reconstructing 3D models first [83, 84, 85]. Being able to achieve competitive results against a specialised 2D landmark predictor is necessary for the success of a 3D dense ear reconstruction algorithm. Zhou and Zaferiou's approach comes with the ITWE-A dataset and is considered as a baseline. They use Scale Invariant Feature Transform (SIFT) features and an AAM model to predict 2D landmarks [1]. Hansley and Segundo [86] propose a CNN-based approach to regress 2D landmarks directly, and they also evaluate the ITWE-A dataset. Their approach proposes two CNNs that both predict the same set of landmarks but with different strengths. The first CNN has a better generalisation ability for different ear poses. The resulting landmarks of the first CNN are used to normalise the ear image. The second CNN predicts improved normalised ear images based on the results of the first CNN.

2.3.6 3D-to-3D Face Reconstruction

A 3D-to-3D autoencoder based method means that it uses an encoder to extract the latent representation from the input 3D face and a decoder to reconstruct the original input 3D face. Most current 3D face datasets use 3D meshes to represent their 3D face scans. A 3D mesh comprises a point cloud and a mesh topology. Depending on whether mesh topology information is utilised, encoder networks fall into two categories: (i) networks that process unordered point cloud data (*e.g.* PointNet [16], PCT [17]), and (ii) Graph Convolutional Networks (GCN) [87], which process sets of points with a predefined mesh topology (*i.e.* meshes). Recent GCN-based methods [22, 21, 23, 24] can only train on registered single datasets, however with PointNet, Liu *et al.* [88] train on combined multiple datasets with different topologies without point correspondence, which is a significant step towards reducing the limitations in the type of input data employed. We choose to employ an intermediate solution, such as [89], which uses registered point clouds only on single datasets and achieves better reconstruction and expression disentanglement results without topology information. Noting recent successes of transformer networks [90] in computer vision tasks [91], we use an open-sourced approach that applies this architecture to point cloud data (the Point Cloud Transformer (PCT) [17]) as our encoder for the work presented in Chapter 5. The advantage of avoiding using the topology information is that it allows more flexibility in the form of the input data. With a topology, all the vertices connected by the topology are required to be in certain order, and the number of vertices has to be fixed. While using point clouds as input data, those constrains exist no more. However, in our approach, we still use a fixed amount of vertices and fixed order. Although the backbone network enables processing vertices of different orders and amounts.

2.3.7 Disentangled 3D Facial Expression Modelling

3DMMs initially focus on modelling the variance over different identities of people, *e.g.* Basel Face Model 2009 (BFM09) [58], but latterly have added additional expression models to better model real faces that possibly appear with expressions. A number of works [59, 92, 21] model expression by modelling datasets that contain faces with expression, resulting in a set of identity-expression mixture coefficients. On the other hand, a number of models use separate coefficients for identity and expression [93, 3, 88, 94]. However, all aforementioned methods in this subsection do not explicitly disentangle identity and expression and the two disentangling works that are most related to us, [95] and [54], both use the GCN (Graph Convolutional Network, [87]) as their encoder. Note that some of the methods build separate models for identity and expression, however they are not built to explicitly separate identity and expression features given an input that contains a face with facial expression.

To achieve facial expression disentanglement, the way in which identities and expressions are combined has to be defined. In the context of modelling identity and expression in two separate sets of coefficients, Egger *et al.* [96] classify the combination of identity model and expression model into three categories: additive, multiplicative and nonlinear models. Zhang *et al.* [54] use the additive assumption where expressions are represented in a blendshape that each vertex's coordinates can be directly added to the corresponding neutral face vertex's coordinates. Jiang *et al.* [95] use the nonlinear model way that feeds both identity and expression latent code to a deep neural network and synthesises the final expression faces directly, which is the approach that we follow in our proposed method.

Jiang *et al.* [95] employ two networks, one removing identity and one removing expression from the input face, thus expecting the synthesised face to be the average face. They also synthesise the original face by a fusion network that combines the results from the identity remover and the expression remover. Zhang *et al.* [54] achieved the previous state-of-the-art in 3D facial expression disentanglement results prior to our work [51]. They propose to add an objective that suggests independence between the identity latent code and the expression latent code by utilising a discriminator similar to that of Kim and Mnih [9].

2.4 Conclusion

In this section, we conclude the relevant literature reviewed in this chapter and address the gaps in the current literature. In this chapter, we firstly review the core technologies in machine learning and deep learning used in this thesis. We cover the basic concepts and theories of the learning-based methods that aim to improve a model by observing the data. Then we focused on a more specific subset of the machine learning area that focuses on building a pipeline to learn automatically from the data without manual annotations. Then we cover the 3D morphable model and other individual technologies that are used in implementing the algorithms covered by this thesis. Finally, we review the recent literature that is the most relevant to our proposed methods.

For the end-to-end 3D reconstruction tasks using autoencoders, there has been a debate as to whether the encoders are needed. For some generation tasks, designing an encoder is not necessary and can require a large capacity model that is potentially unimplementable in practice. DeepSDF [30] addresses this issue by proposing an encoder-less auto-decoder architecture. Generative Adversarial Networks (GANs) [56] propose to use a discriminator network instead of the encoder to train the generator/decoder. For example, for the face generation task, an enormous number of GAN-based methods are proposed, few representatives are included [97, 98, 99, 100, 101, 102].

For generating realistic samples, there are gaps between current methods, too. For generating 3D data like point clouds, a multilayer perceptron-based decoder is sufficient for reasonable scales of the data. However, for generating 2D data like images, there is a trade-off between using a model-based decoder to explicitly model the 2D image generation process and using GANs to generate implicitly and directly. While a GAN is able to generate photo-realistic photos directly, it loses the advantages of the model-based decoders where more controls are obtained over properties such as rendering properties. A model-based decoder can explicitly vary semantically meaningful properties, such as lighting directions, face poses, texture materials and more. However, model-based decoders generate images that are less realistic compared to those generated by GANs. We address the control issue in Chapter 5 by disentangling the implicit method's parameters.

Similar to the realistic 2D image generation, the gap between model-based methods and implicit methods has been addressed more and more in recent years. Model-based methods have the advantage of possessing manageable and explainable properties. However, under most circumstances, they fail to compete with the implicit methods implemented with deep learning technologies in terms of the prediction accuracy of specific tasks. We address this trade-off in Chapter 4 and attempt to build a hybrid method that has both advantages.

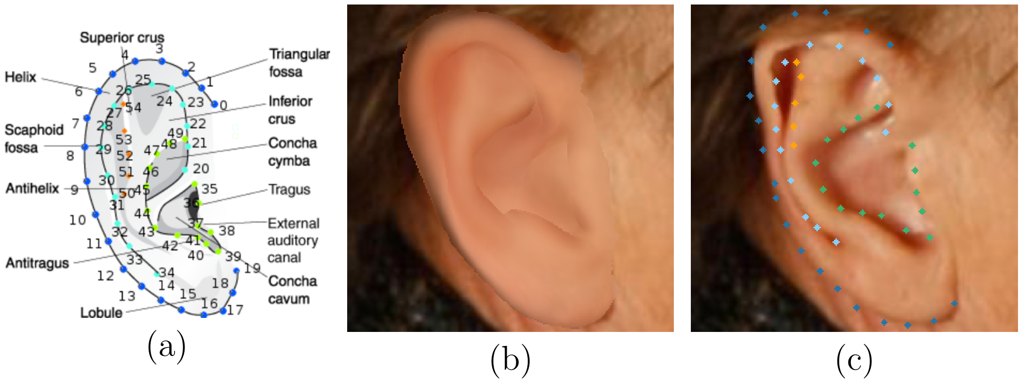
3 Human Ear Reconstruction Autoencoder

3.1 Introduction

In recent years, 3D face modelling and 3D face reconstruction from monocular images have drawn increasing attention. Especially with deep learning methods, 3D face reconstruction models are empowered to have more complexity and better feature extraction ability. However, as an important part of the human head, the human ear has received significantly less attention. Our 3D ear reconstruction approach establishes a dense correspondence between 2D ear input image pixels and 3D vertices of a 3D Morphable Model (3DMM) of the ear, thus enabling both 2D and 3D ear landmark localisation. Furthermore, 3D ear recognition is enabled [1, 103, 104] using the 3D shape encoding provided by the fitted 3DMM. In addition, we perform extensive experiments on the Headspace dataset [105, 106] by utilising our Human Ear Reconstruction Autoencoder, or HERA (pronounced 'hearer') system, as an initialisation to a 3D model fitting problem.

A detailed 3D ear reconstruction can be crucial to building a high-quality 3D model of the human head [107, 10, 61, 106]. In this context, it is desirable to model the ears as separate entities and then fuse them to the head. The reason is that it is difficult to control the spatially high-frequency aspects of the ear (such as the skin folds) with parameters that simultaneously control the whole head shape in a global optimisation. Such a 3DMM head fitting optimisation is better at capturing the low-frequency shape variations (*i.e.* relatively large components with relatively small local variations, *e.g.* face shape and cranial shape) across an aligned dataset of some shape class.

A number of applications are possible with the detailed ear shape modelled by a fitted ear 3DMM. This includes the design of ear wear (headphones, earphones, hearing aids), eye wear (since eye wear frames usually require ear support) and other head wear used in virtual and augmented reality applications.



(a) (b) (c)

Figure 3.1. (a) 55 landmarks and their semantics from ITWE-A dataset [1] (b) Rendered densely corresponded coloured 3D ear mesh projected onto the original image (c) Original image marked with predicted landmarks.

Most modern approaches for 3D face or 3D ear reconstruction from monocular images fall into three categories: generation based, regression based and the combination of both [6]. Generation-based methods require a parametric model for the 3D object and 3D landmarks to optimise a set of parameters for optimal alignment between projected 3D models and 2D landmarks. For 3D ear reconstructions, two approaches can be found in literature [80, 1]. Both are traditional generation-based methods that utilises model-fitting or

Active Appearance Model (AAM) to fit either a 3D or a 2D ear model to the ear images to localise 55 ear landmarks. Regression-based methods usually utilise neural networks to regress a parametric model's parameters directly, as proposed by [108, 109] for 3D face reconstruction. Generation-based methods are often more computationally costly, due to their non-convex optimisation criteria and the requirement for landmarks. Regression-based methods require ground truth parameters to be provided, which is only accessible when using synthetic data [108]. Otherwise other 3D reconstruction algorithms are required to obtain ground truth parameters beforehand [83]. Therefore, Tewari *et al.* proposed an unsupervised 3D face reconstruction method named *Model-based Face Autoencoder* (MoFA) that combines both generation and regression based methods. This aims to mitigate the negative aspects of the two categories of method, by using an autoencoder composed of a regression-based encoder and a generation-based decoder [6]. However, there are no regression-based or autoencoder structured approaches for 3D ear reconstruction in the literature. Whether this unsupervised autoencoder approach can tackle the complexity of the ear structure remains an open question that we address here.

The core idea of the unsupervised learning approach is to synthesise similar colour images from original colour input images in a differentiable manner. For such an approach, a parametric ear model is needed. Dai *et al.* propose a 3D Morphable Model (3DMM) of the ear, named the York Ear Model (YEM, [80, 10]). Its 3D ear mesh has 7111 vertex coordinates, so 21333 vertex parameters, reduced to 499 shape parameters using PCA. However, to enable unsupervised learning, the 3D ear meshes require colour/texture, which is not included in the YEM model. Furthermore, we perform 3D model fitting of the ear model to the raw 3D head scans to enable further understanding of such a small portion of the overall 3D head mesh. Such 3D model fitting is a challenging task since a small model is fitted to a large object. However, we mitigate this problem by employing the HERA system to provide a strong initialisation of the ear model, thereby making the whole model fitting both more robust and efficient.

In this context, we present a *Human Ear Reconstruction Autoencoder* (HERA) system, with the following contributions:

- A 3D ear reconstruction method that is trained in a completely end-to-end way, using in-the-wild monocular colour 2D images of the ear, and can potentially be trained unsupervised.
- An in-the-wild ear color model that colors the 3D ear mesh to minimise its difference from the associated 2D ear image in appearance.
- Evaluations that demonstrate that our proposed model is able to predict a colored 3D ear mesh in dense correspondence with other fitted models *e.g.* Fig. 3.1 (b), and 2D landmarks *e.g.* Fig. 3.1 (c).
- A set of 55 3D ear landmarks for the 3D head meshes in the Headspace dataset [106], which is generated by fitting the HERA-initialised York Ear Model to the raw 3D head meshes and transferring landmarks from the YEM model to the Headspace data. This has great utility when fitting full head models to Headspace data.

3.2 The HERA system

Our HERA system (Human Ear Reconstruction Autoencoder) employs an autoencoder structure that takes right ear images as input and generates synthetic images. Where left ears are mirrored to right ears in the first stage

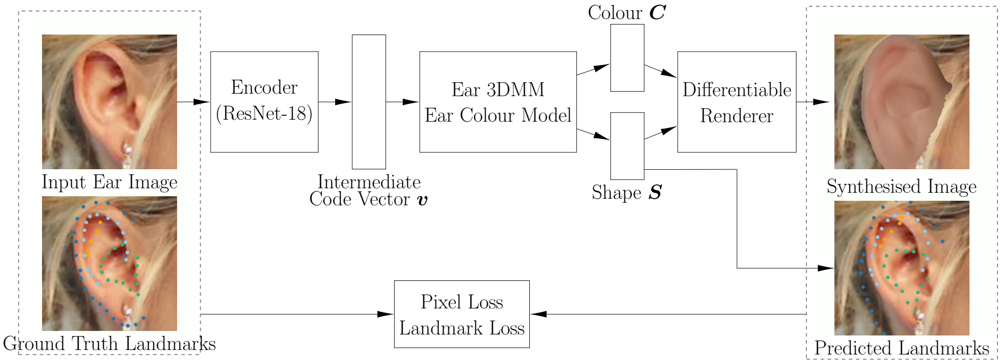


Figure 3.2. Overview of the HERA autoencoder architecture. The encoder is the ResNet-18 CNN predicting intermediate code vectors that are then fed to the decoder. The decoder is comprised of: *(1)* the YEM ear shape model and our in-the-wild ear colour model; *(2)* PyTorch3D [2] that renders images with ear shapes and colours in a differentiable way. We use a photometric (pixel) loss with an optional additional landmark loss for faster convergence and better accuracy.

of preprocessing. The autoencoder is trained by minimising the difference between input images and the final synthesised images. An illustration of our end-to-end architecture is shown in Fig. 3.2. The encoder is a CNN predicting intermediate code vectors that are then fed to the decoder, where coloured 3D ear meshes are reconstructed and rendered into 2D images. The decoder is comprised of: (i) the YEM ear shape model and our in-the-wild ear colour model that reconstructs ear shapes and ear colors respectively; (ii) PyTorch3D [2] that renders images with ear shapes and colours in a differentiable way. The comparison of the input and synthesised images is implemented by a combination of loss functions and regularisers. The essential loss function is a photometric loss where mean square error is calculated for every pair of corresponded pixels from both input images and synthesised images, with an additional landmark loss that can be included for both faster convergence time and better accuracy. The whole autoencoder structure is designed to be differentiable, so it can be trained in an end-to-end manner. Each part of the architecture (*i.e.* encoder CNN, ear 3DMM, scaled orthogonal projection and loss functions) is differentiable by default, thereby using a differentiable renderer to render 3D meshes with textures to 2D images for making the whole architecture differentiable. The core part of the decoder is described in Section 3.2.1. The whole end-to-end trainable architecture and the necessary training methods are then described in Section 3.2.5.

3.2.1 3D Morphable Model of the Ear

The decoder comprises an ear shape model derived from the York Ear Model (YEM), an ear colour model, and a 3D-to-2D projection model. The YEM shape parameters $\boldsymbol{\alpha}_s$ can be reconstructed to an 3D ear vertex coordinate vector $\mathbf{S} \in \mathbb{R}^{N \times 3}$ where N is the number of vertices in a single 3D ear mesh. The colour parameters $\boldsymbol{\alpha}_c$ are then reconstructed to a vertex colour vector $\mathbf{C} \in \mathbb{R}^{N \times 3}$ to colour each vertex. The pose parameters $\mathbf{p}$ are used in the projection model that aligns 3D ear meshes with 2D ears' pixels. Note that we assume the shape and the in-the-wild colour are independent, thus the shape and the colour models are built separately. Even they have potential correlations, the two separate models can still capture the correlations individually and does not affect the correctness of the finally synthesised images.

Ear Shape Model

For the ear geometric information modelling, we employ the YEM model [80] to perform reconstruction. It is constructed from 500 3D ear meshes and thus provides a strong statistical prior. The 3D ear vertex coordinate vector (*i.e.* 3D ear shape) $\mathbf{S}$ is reconstructed from shape parameter vector $\boldsymbol{\beta}_S$ by:

$$\mathbf{S} = \hat{S}\left(\boldsymbol{\beta}_s\right) = \mathrm{vec}^{-1}(\bar{\mathbf{S}} + \mathbf{U}_s\boldsymbol{\beta}_s), \tag{3.1}$$

where $\bar{\mathbf{S}} \in \mathbb{R}^{3N}$ is the mean ear shape, $\mathbf{U}_s \in \mathbb{R}^{3N \times 499}$ are the principal components of ear shape variation and the resulting $3N$-vector is reshaped into a $N \times 3$ matrix by the operator $\mathrm{vec}^{-1}(.)$ such that each row of $\mathbf{S}$ represents a vertex coordinate in 3D space.

The 3D-to-2D projection model that we used is the *Scaled Orthogonal Projection* (*SOP*). Note that we make this *SOP* assumption because no camera parameters are available to the in-the-wild ear images, and most ears in the dataset images are relatively far from the camera, further minimise the effect of using the scaled orthogonal projection against the full perspective projection. Given the 3D ear shape $\mathbf{S}$ from Eq. (3.1) and similarity transform parameters $\mathbf{p} = (\mathbf{r}, \mathbf{T}, f)^T$ comprising rotation, translation and scale respectively, this projection function, $\mathbf{V}$, is defined as:

$$\mathbf{V} = \hat{V}(\mathbf{S}, \mathbf{p}) = f \mathbf{P}_o \hat{R}(\mathbf{r}) \mathbf{S} + \mathbf{T}, \quad \mathbf{P}_o = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \tag{3.2}$$

where $\mathbf{V} \in \mathbb{R}^{N \times 2}$ are the projected 2D ear vertices, $\mathbf{P}_o$ is the orthogonal projection matrix and $\hat{\mathbf{R}}(\mathbf{r})$ is a function that returns the rotation matrix from $\mathbf{r} \in \mathbb{R}^3$, axis rotation angles in x, y and z axis respectively (*i.e.* azimuth, elevation and roll). Since scaled-orthogonal projection is used, $\mathbf{V}$ provides sufficient geometric information for the differentiable renderer and no additional camera parameters are needed.

In addition, 2D landmarks can be extracted from the projected vertices $\mathbf{V}$ by manually selecting 55 semantically corresponding vertices. Thus we can define a vector of 2D landmarks of a projected ear shape $\mathbf{V}$ as:

$$\mathbf{X}_i = \mathbf{V}(\mathbf{L}), \tag{3.3}$$

where $\mathbf{X}_i \in \mathbb{R}^{55\times 2}$ are the landmark's image coordinates indexed by $\mathbf{L}$ in the projected ear vertices $\mathbf{V}$.

In-the-wild Ear Colour Model

The decoder in our architecture requires the 3D ear meshes to be coloured to generate plausible synthetic ear images. However, the YEM model contains an ear shape model only. To solve this problem, we build an in-the-wild ear colour model using PCA whitening.

Firstly, for each ear image of the 500 images from the training set of the ITWE-A dataset, a set of whitened ear shape model $\boldsymbol{\alpha}_s$ and ear pose parameters $\mathbf{p}$ is fitted using a non-linear optimiser (*L-BFGS-B* implemented by SciPy package [15]) to minimise 2D landmark distances. Using the reconstruction Equations 3.1 $\sim$ 3.3, the optimisation criterion E_0 can be formed as follows:

$$\hat{X}\left(\boldsymbol{\alpha}_s, \mathbf{p}\right) = \hat{V}\left(\hat{S}\left(\hat{\alpha}\left(\boldsymbol{\alpha}_s\right)\right), \mathbf{p}\right), \tag{3.4}$$

$$E_0\left(\boldsymbol{\alpha}_s, \mathbf{p}, \mathbf{X}_{gt}\right) = \frac{1}{N_L}\left\|\left(\hat{X}\left(\boldsymbol{\alpha}_s, \mathbf{p}\right)\right)(\mathbf{L}) - \mathbf{X}_{gt}\right\|_2, \tag{3.5}$$

where $\hat{X}$ is the whole reconstruction and projection function and $\hat{\alpha}\left(\boldsymbol{\alpha}_s\right) = \boldsymbol{\beta}_s$ generates shape parameters from whitened parameters $\boldsymbol{\alpha}_s$ (described in Section 3.2.3). Also, $N_L = 55$ is a constant representing the number of landmarks and $\mathbf{X}_{gt} \in \mathbb{R}^{55\times 2}$ are the ground truth 2D landmarks provided by the ITWE-A dataset.An illustration of the definitions of all 55 ear landmarks can be found in Fig. 3.1 (a).

After shape fitting, the colour for each vertex is obtained by selecting the corresponding 2D pixel colour. This process generates 500 vertex colour vectors, which can then be used to build the in-the-wild ear colour model

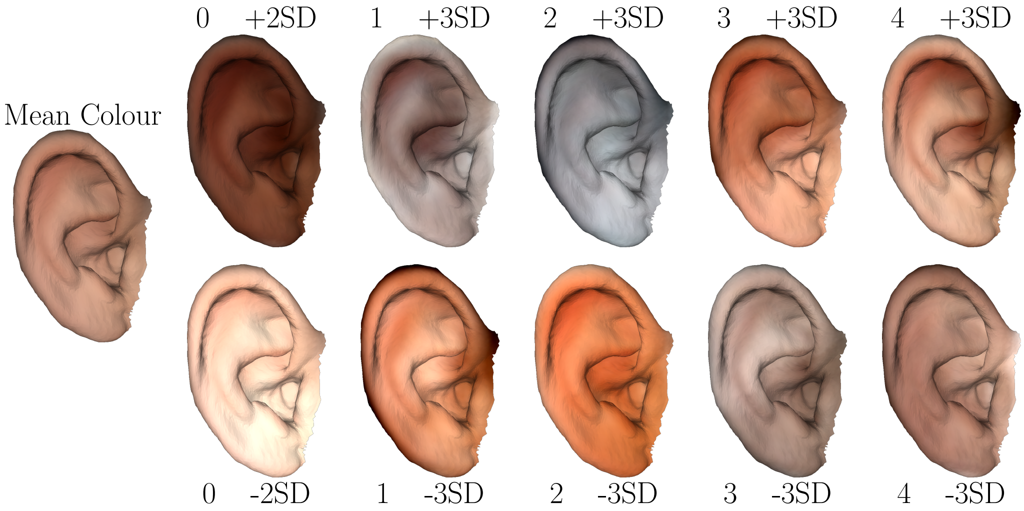


Figure 3.3. The HERA in-the-wild ear colour model. The mean colour and first 5 parameters ± standard deviations (SD) are shown. The mean 3D ear mesh is used.

using PCA whitening. The colour vector length is 40 which covers 86.6% of the colour variation. This is set at a moderate value to allow our system to implicitly ignore some occlusions (*e.g.* hair and ear piercings). The HERA colour model is shown in Fig. 3.3.

The reconstruction of the vertex colour vector **C** is:

$$\mathbf{C} = \hat{C}\left(\boldsymbol{\alpha}_c\right) = \bar{\mathbf{C}} + \mathbf{U}_c\boldsymbol{\alpha}_c, \tag{3.6}$$

where $\boldsymbol{\alpha}_c \in \mathbb{R}^{40\times 1}$ is the colour parameter vector. $\bar{\mathbf{C}}$ is average vertex colour vector, $\mathbf{U}_c$ is vertex colour variance component matrix and both are calculated by the PCA whitening algorithm.

3.2.2 Intermediate Code Vector

The intermediate code vector

$$\mathbf{v} = (\mathbf{p}, \boldsymbol{\alpha}_s, \boldsymbol{\alpha}_c)^T \tag{3.7}$$

connects the encoder and the decoder and has semantic meaning, where $\mathbf{p} = (\mathbf{r}, \mathbf{T}, f)^T$ defines the similarity transform (pose plus scale) of the 3D ear mesh. The vector $\mathbf{r} \in \mathbb{R}^3$ is the azimuth, elevation and roll that maps to the rotation matrix through the function $\hat{R}(\mathbf{r}) : \mathbb{R}^3 \rightarrow \mathbb{R}^{3\times3}$. $\mathbf{T} \in \mathbb{R}^{2\times1}$ defines the translation in x-axis and y-axis. Translation in the z-axis is not necessary, since scaled orthogonal projection is used. Note that f defines the 3D mesh's scale and $\boldsymbol{\alpha}_s \in \mathbb{R}^{40\times1}$ are the PCA-whitened shape parameters that will generate the shape parameters $\boldsymbol{\beta}_s \in \mathbb{R}^{499\times1}$ employed by the YEM 3DMM. $\boldsymbol{\alpha}_c \in \mathbb{R}^{40\times1}$ are the colour parameters for our in-the-wild ear colour model.

3.2.3 PCA Whitening

To ease the backpropagation process in training, we use PCA whitening to transfer the YEM ear model parameters into a format that is more favourable for deep learning frameworks. Firstly, the variances of the parameters can differ in a very large scale from 8×10^3 for the most significant parameter to 5×10^{-7} for the least important parameter. It is difficult to train a neural network to effectively regress such large variance data. Secondly, the large number of the parameters increases the neural networks' training time and is detrimental to the optimisation process. This could be mitigated by removing the least important parameters. However, this may lose shape and color information. Therefore, we perform PCA whitening [110] over the full set of parameters. PCA whitening aims to generate zero-mean parameters with reduced dimensions in unit-variance. In our experiment, YEM's original parameters $\boldsymbol{\beta}_s$ of 499 dimensions are transformed to $\boldsymbol{\alpha}_s$ of 40 dimensions while covering 98.1% of the variance associated with the original parameters. With such method, the ear shape and colour model has an equal number of parameters. The use of PCA whitening is further justified in the ablation study section that the training time is greatly reduced. Each original parameter vector $\boldsymbol{\beta}_s$ can be recovered from $\boldsymbol{\alpha}_s$ by:

$$\boldsymbol{\beta}_s = \hat{\alpha}\left(\boldsymbol{\alpha}_s\right) = \mathbf{U}_w \boldsymbol{\alpha}_s, \tag{3.8}$$

where $\mathbf{U}_w \in \mathbb{R}^{499 \times 40}$ is a constant matrix. The original parameters' mean is not added since they are already zero-mean.

3.2.4 Differentiable Renderer

A differentiable renderer is used for all end-to-end 2D-to-2D pipelines in this thesis. A renderer in the context of this thesis is a function that takes a 3D object and various scene settings and generates a 2D image. The scene settings can include multiple parameters such as camera parameters, lighting parameters and surface materials [2]. The term *differentiable* means the rendering process is differentiable, and the render function has gradients. Therefore, we can put the differentiable renderer in our end-to-end model and train all the components as a whole system. In our experiments, we use the PyTorch3D [2] package to perform differentiable rendering and use a weak or full perspective camera projection system with an ambient lighting model.

3.2.5 Ear Autoencoder

We now discuss the architecture, loss function components and data augmentation employed for the end-to-end training of the HERA system. Note that our method assumes the ear detector places the ear perfectly in the centre of the image, thus ignore the scale and translation invariance feature. This can be potentially added via data augmentation. The existing data augmentation only covers the variance in ear rotations. Fig. 3.2, shows our architecture that consists of an encoder, an intermediate code vector, the decoder components, the differentiable renderer and the loss for back-propagation. The encoder is a pre-trained 18-layer residual network (ResNet-18) which is a CNN that performs well on regression from image data [111]. The adoption of the ResNet-18 is that ResNet is the most popular backbone network at the time of experiment, and training data size is limited so the 18 layer version is empirically more suitable than the mostly adopted 50 layer version. The practical results show promising performance by the ResNet-18. We use PyTorch3D [2] as a differentiable image renderer developed using PyTorch [112]. It is a differentiable function that maps a set of vertex coordinate vector and vertex colour vector to a 2D image. The encoder Q and decoder W can be formed as follows:

$$\mathbf{v}_{pred} = Q\left(\mathbf{I}_{in}, \boldsymbol{\theta}\right), \tag{3.9}$$

$$\mathbf{S}^{T}_{pred}, \mathbf{C}_{pred} = W\left(\mathbf{v}_{pred}\right), \tag{3.10}$$

$$\mathbf{I}_{pred} = R\left(\mathbf{S}^{T}_{pred}, \mathbf{C}_{pred}\right), \tag{3.11}$$

$$\mathbf{X}_{pred} = \mathbf{S}^{T}_{pred}\left(\mathbf{L}\right), \tag{3.12}$$

where $\mathbf{I}_{in}$ is the input image and $\boldsymbol{\theta}$ are the weights of the encoder network Q. In the decoder W, the predicted 3D mesh (*i.e.* shape with pose $\mathbf{S}^{T}_{pred}$

and colour $\mathbf{C}_{pred}$) are reconstructed from the predicted intermediate code vector $\mathbf{v}_{pred}$. The reconstructed 3D mesh is then fed to the differential image renderer, $R(.)$, to generate the rendered image $\mathbf{I}_{pred}$. Note that $\mathbf{L}$ are the indices of the 55 ear landmark vertices in the ear shape $\mathbf{S}$ and $\mathbf{X}_{pred} \in \mathbb{R}^{55\times 2}$ are the predicted landmarks positions. The ResNet-18 encoder is initialised using the weights from pretraining on ImageNet [113]. The trained encoder network can be used for shape and color parameter regression.

Loss Function

We follow established loss function components for unsupervised 3D reconstruction approaches [6] and employ a combination of four weighted losses as:

$$\begin{aligned} E_{loss} = {} & \lambda_{pix} E_{pix}\left(\mathbf{I}_{in}\right) + \lambda_{lm} E_{lm}\left(\mathbf{I}_{in}, \mathbf{X}_{gt}\right) \\ & + \lambda_{reg1} E_{reg1}\left(\mathbf{I}_{in}\right) + \lambda_{reg2} E_{reg2}\left(\mathbf{I}_{in}\right), \end{aligned} \tag{3.13}$$

where λ_i are the weights for the losses E_i.

Pixel Loss Synthesising an output image and comparing it to the associated input images is the core idea of the autoencoder architecture. To form such comparison, the Mean Square Error (MSE) is used on all pixels:

$$E_{pix}\left(\mathbf{I}_{in}\right) = L_{MSE}\left(R\left(W\left(Q\left(\mathbf{I}_{in}, \boldsymbol{\theta}\right)\right)\right), \mathbf{I}_{in}\right), \tag{3.14}$$

where L_{MSE} is a function that calculates the mean square error. A pixel mask is used to compare the rendered ear region only, since the rendered ear images have no background.

Landmark Loss The optional landmark loss is used to speed up the training process and help the network learn to generate 3D ears with better accuracy. Zhou and Zaferiou [1] propose the mean normalised landmark distance error as their shape model evaluation metric. Note that the ground truth 55 landmarks for each ear image are provided along with their paper. Here, we employ it as a part of the loss function. It can be formed as:

$$E_{lm}\left(\mathbf{I}_{in}, \mathbf{X}_{gt}\right) = \frac{\left\|\left(W\left(Q\left(\mathbf{I}_{in}, \boldsymbol{\theta}\right)\right)\right)\left(\mathbf{L}\right) - \mathbf{X}_{gt}\right\|_2}{D_N\left(\mathbf{X}_{gt}\right) N_L}, \tag{3.15}$$

where $\mathbf{X}_{gt}$ is the ground truth landmarks and $D_N\left(\mathbf{X}_{gt}\right)$ is a function that returns the diagonal pixel length of the ground truth landmarks' bounding box. Since this loss is optional, setting $\lambda_{lm} = 0$ can enable the whole model to be trained on 2D image data $\mathbf{I}_{in}$ only, making the use of very large-scale unlabelled training data possible.

Regularisers We constrain the learning process with two weighted regularisers. The first regulariser is a statistical plausibility regulariser. This follows the basic assumption of the PCA whitening algorithm that each parameter has zero mean, and setting the weight of the regularisers to a small number can encourage the prediction to stay within the model space while not over-penalise the prediction to be all zeros.

This regulariser is formed by:

$$E_{reg1}\left(\mathbf{I}_{in}\right) = \sum_{j=1}^{40} \boldsymbol{\alpha}_{sj} + \sum_{j=1}^{40} \boldsymbol{\alpha}_{cj}, \tag{3.16}$$

where $\boldsymbol{\alpha}_s$ and $\boldsymbol{\alpha}_c$ are ear shape and colour parameters predicted by the encoder network. Therefore, this penalises the Mahalanobis distance from the mean shape and colour.

An additional restriction on the scale parameter f has to be applied for the model to be successfully trained without landmarks in practice. The restriction is formed by:

$$E_{reg2}\left(\mathbf{I}_{in}\right) = \begin{cases} \left(0.5 - f\right)^2 & \text{if } f < 0.5 \\ \left(f - 1.5\right)^2 & \text{if } f > 1.5 \\ 0 & \text{otherwise} \end{cases}, \tag{3.17}$$

Tuning the hyperparameters are non-trivial for such small dataset. We use a cross validation set to determine an empirically working set of parameters and use it to train the whole training set. We employ two sets of weights, λ, depending on whether or not landmark loss is used when training.

- Training with landmarks: $\lambda_{pix} = 10$, $\lambda_{lm} = 1$, $\lambda_{reg1} = 5 \times 10^{-}2$ and $\lambda_{reg2} = 0$
- Training without landmarks: $\lambda_{pix} = 2$, $\lambda_{lm} = 0$, $\lambda_{reg1} = 5 \times 10^{-}2$ and $\lambda_{reg2} = 100$

Dataset Augmentation

We perform data augmentation on the ITWE-A dataset, since it contains only 500 landmarked ear images, with limited variability of ear rotation. An ear direction of a 2D ear image is defined by a 2D vector from one of the ear lobe landmark points to one of the ear helix landmark points. For each 2D ear image, 12 random rotations around its central point are applied such that the angles between their ear directions and the Y-axis of the original image are uniformly distributed between $-60°$ and $60°$. The augmented ear image dataset contains $6,000$ images in total. With this augmentation, we find that the test set landmark error drops significantly.

3.3 Results

Both quantitative and qualitative evaluation results are discussed in this section. Quantitative evaluation focuses on comparing landmark fitting accuracy with different approaches, while the qualitative evaluation focuses on evaluating the visual results of this 3D ear reconstruction algorithm. Furthermore, an ablation study is conducted to analyse the improvements that various optimisations of this work have proposed, including the PCA whitening on the YEM model parameters, the statistical plausibility regulariser and the dataset augmentation. The abbreviation HERA (Human Ear Reconstruction Autoencoder) represents the *final* version of this work.

3.3.1 Quantitative Evaluation

The mean normalised landmark distance error proposed by [1] is the main quantitative evaluation method we applied. It is formed in Eq. (3.15), which also forms the landmark loss that trains our system. Projecting the 3D ear meshes' key points to 2D and comparing them with the ground truth can assess the accuracy of the 3D reconstruction. There are two approaches that predict the same set of landmarks using the same dataset in the literature, therefore comparisons can be formed. Zhou & Zaferiou's work [1] is considered as a baseline solution and Hansley & Segundo's work [86] is a specifically-designed 2D landmark localisation algorithm that has the lowest landmark error in the literature. To interpret the landmark error, it is suggested that, for an acceptable prediction of landmarks, the mean normalised landmark distance error has to be below 0.1 [1]. This is a dimensionless metric that is the ratio of the mean Euclidean pixel error to the diagonal length of the ear bounding box. As stated in Section 3.2.5, HERA can be trained

Table 3.1. Normalised landmark distance error statistics on ITWE-A.

Method	mean ± std	median	≤ 0.1	≤ 0.06
Zhou & Zaferiou	0.0522 ± 0.024	0.0453	95%	78%
Hansley & Segundo	$\mathbf{0.0393} \pm 0.0169^*$	0.0399^*	100%	93%
HERA	$0.0398 \pm \mathbf{0.009}$	**0.0391**	**100%**	**96.2%**
HERA-W/O-AUG-LM	0.0591 ± 0.014	0.0567	99%	64.7%

* Estimated from cumulative error distribution curve.

without landmarks or data augmentation in an unsupervised manner. The HERA version that uses no landmark loss during training and trains on the original 500 ear images is named HERA-W/O-AUG-LM.

The HERA system is now compared with Zhou & Zaferiou's and Hansley & Segundo's work regarding the normalised landmark error's mean, standard deviation, median and cumulative error distribution (CED) curve evaluated on the test set of ITWE-A which contains 105 ear images. The numerical results are shown in Tab. 3.1 and the CED curve is shown in Fig. 3.4. Additionally, the percentage of predictions that have error less than 0.1 and 0.6 are given in Tab. 3.1.

As shown in Tab. 3.1 and Fig. 3.4, HERA outperforms Zhou & Zaferiou's work by a large margin in terms of 2D landmark localisation task. When compared with Hansley & Segundo's 2D landmark localisation work, similar results are shown. This is considered acceptable when comparing a 3D reconstruction algorithm with a 2D landmark localisation algorithm. Hansley & Segundo's landmark localiser is comprised of two specifically designed CNNs for landmark regressions while HERA uses only one CNN to regress a richer set of information (*i.e.* pose, 3D model parameters and colour parameters). Regarding the threshold of 0.1 proposed by [1], both HERA and Hansley & Segundo's work are 100% below 0.1, and HERA trained without

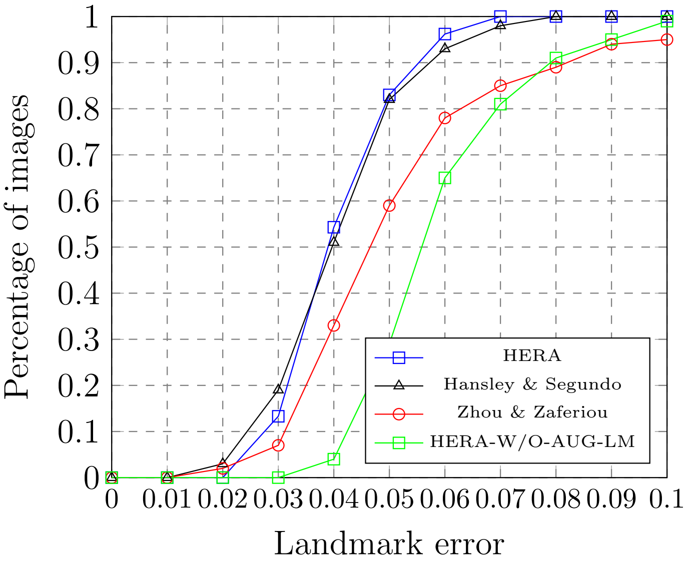


Figure 3.4. Cumulative error distribution curve comparison among different landmark detection algorithms and our work

landmarks achieves 99% below 0.1. The CED curves show that, although HERA-W/O-AUG-LM performs worse than Zhou & Zaferiou's work in the error region below around 0.077, our performance is better at the 0.1 error point. In other words, HERA-W/O-AUG-LM can predict landmarks with less than 0.1 error more consistently than the baseline.

3.3.2 Qualitative Evaluations

We visually show the 3D reconstruction results on ITWE-A's test set as qualitative evaluations. In Fig. 3.5, three images with large colour variation are predicted, the top row shows the 2D landmark predictions look reasonable. The comparison between the top row and the bottom row shows that the quality of the reconstructed 3D meshes are reasonable in geometric aspect, while the in-the-wild colour model can reconstruct a large variation of in-the-wild ear colours even from grayscale images. Readers are encouraged to

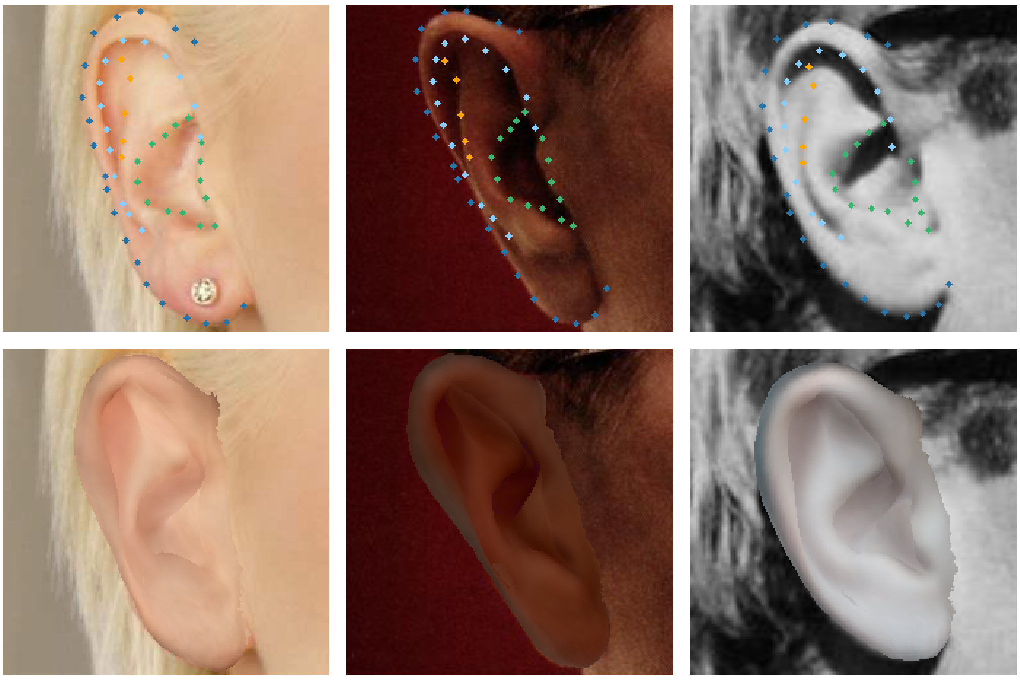


Figure 3.5. Test set prediction results with different ear colours. Top row: original ear images marked with predicted 2D landmarks. Bottom row: predicted 3D ear meshes projected onto original ear images.

examine the original papers for a visual comparison [10, 1]. Note that the reconstruction can still be visially different from the input.

As illustrated in Fig. 3.6, two images with different head poses are selected for 3D ear reconstruction. The top row shows the results from a near-ideal head pose (*i.e.* near-profile face) and the bottom row shows the results from a large head pose deviation from the ideal (*i.e.* front facing, tilted head). The figure shows that HERA works well with different head poses. For the front facing images, the model predicts the correct horizontal rotation rather than narrowing the 3D ear mesh's width to match the 2D image.

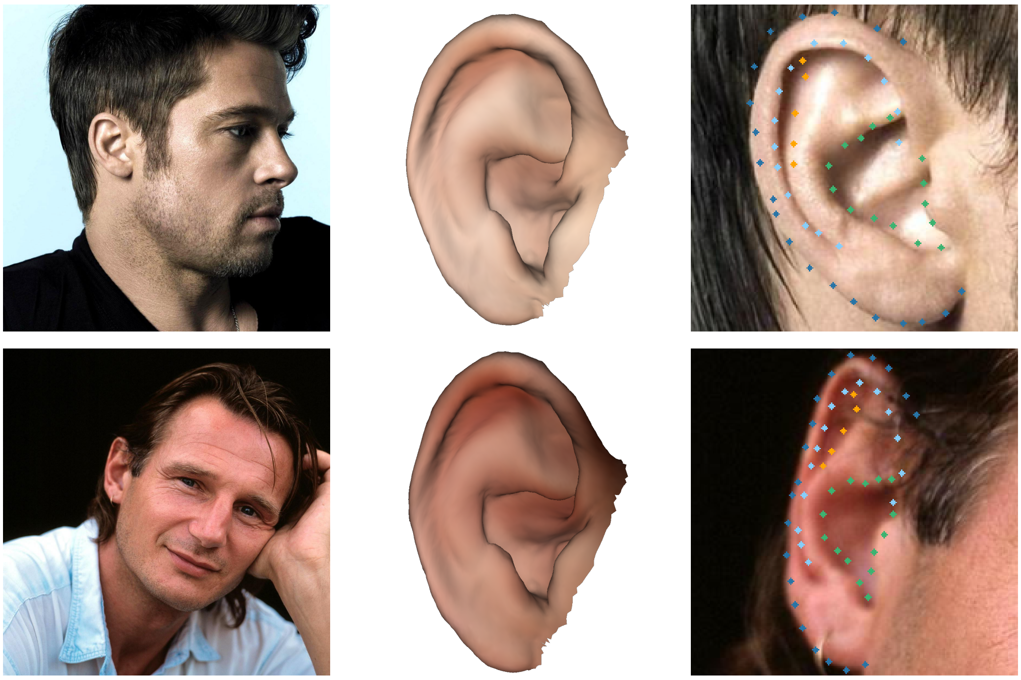


Figure 3.6. Test set prediction results with different head poses. Each row represents a distinct subject. 1st column: Original uncropped images. 2nd column: Predicted 3D ear meshes. 3rd column: Predicted 2D landmarks. Ear pose is successfully predicted when difficult head pose involves.

Table 3.2. Normalised landmark distance error statistics on ITWE-A for ablation study.

Method	mean ± std	median	≤ 0.1	≤ 0.06
HERA	0.0398 ± 0.009	0.0391	100%	96.2%
HERA-W/O-WTN	0.0401 ± 0.009	0.0384	100%	96.2%
HERA-W/O-PIX	0.0392 ± 0.009	0.0387	100%	96.2%
HERA-W/O-AUG	0.0446 ± 0.011	0.0437	100%	92.4%
HERA-W/O-AUG-LM	0.0591 ± 0.014	0.0567	99%	64.7%

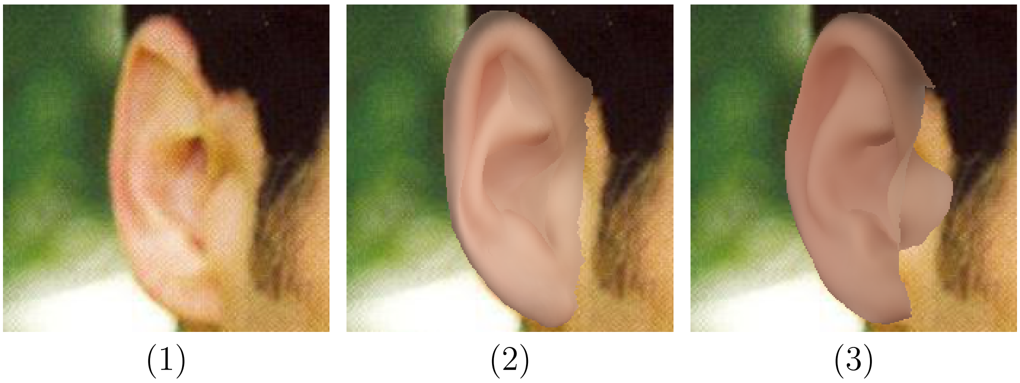


Figure 3.7. Appearance comparison between the reconstructed 3D ear meshes of (1) Ground truth input image, (2) HERA and (3) HERA-W/O-PIX (without using the pixel error). Although the landmark errors are similar, not using pixel error results in a rendered image with more appearance difference.

3.3.3 Ablation Study

We now study each component's effect on HERA's performance and we evaluate on several system variations including HERA-W/O-WTN (without PCA whitening on 3D ear shape parameters, $\boldsymbol{\beta}_s$), HERA-W/O-PIX (without pixel loss), HERA-W/O-AUG (without data augmentation) and HERA-W/O-AUG-LM (without landmark loss).

The statistics for all variations of ablated HERA are shown, along with (full) HERA, in Tab. 3.2. When training without PCA whitening on 3D ear shape parameters and without pixel loss, performance on 2D landmark localisation is similar to the final proposed method. However, using PCA whitening balances the parameters for the neural network to predict and therefore acts as a better underlying design choice. The major contribution of applying PCA whitening in this work is that it speeds up the training process by more than 30% per epoch on a GPU. In the meantime, a balanced design of intermediate code vector with similar variance for each parameter can benefit the performance of the neural network. The proposed HERA

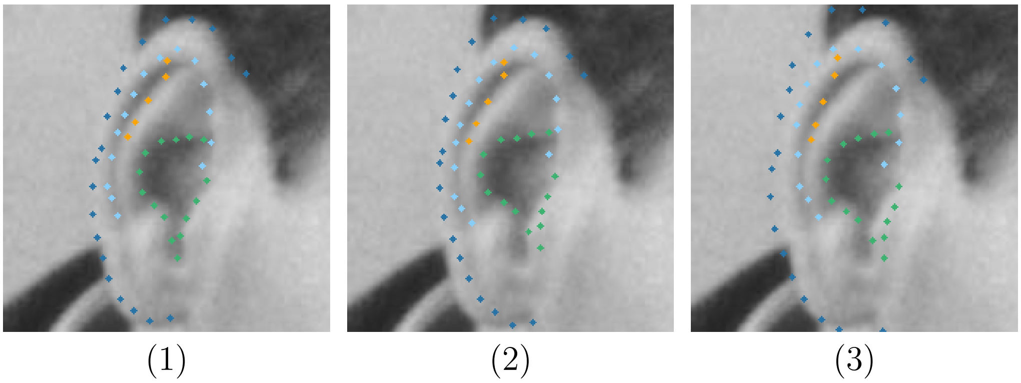


Figure 3.8. 2D landmark localisation comparison between the prediction results of (1) HERA, (2) HERA-W/O-AUG (without data augmentation) and (3) HERA-W/O-AUG-LM (without data augmentation or landmark error). Data augmentation enables better ear rotation prediction and landmark loss is vital to accurate alignment especially for the ear contour part.

system then takes $\sim$ 70 seconds to train one epoch on an NVIDIA RTX 2080 and takes $\sim$ 350 epochs to train the whole network. After training, the network predicts a single image in 6 ms.

When training without pixel loss, as illustrated in Fig. 3.7, the overall appearance of the rendered ear image differs from the input ear image especially for the helix part. Training without pixel loss makes the model focus on lowering the landmark alignment error regardless of the overall appearance of the ear. Therefore it is necessary to utilise the pixel loss. This set of figures also illustrates the pose ambiguity of this system caused by orthogonal projection. For a distinct set of ear parameters, there exists two different rotations that result in the *same* projected 2D landmarks. In one case, such as Fig. 3.7 (1), the external auditory canal part of the ear is visible and in the other case, such as the other rendered images in this chapter, the external auditory canal is covered by itself. This ambiguity may affect further applications that relate the reconstructed 3D ear and other 3D objects, such as the 3D head, but a simple 3D registration task can be carried out to solve the rotational ambiguity, if required. Restrictions on the rotations during the training phase can be applied to allow the results to fall into the desired range.

For training without data augmentation, the 2D landmark localisation performance drops by a small amount mainly due to its lack of variety in ear rotation, shown in Fig. 3.8. When training without landmark loss, the predicted landmark positions are not accurate enough, as shown in Fig. 3.8. As a result, the reconstructed 3D ears are not accurately aligned with the 2D ears, especially for the outer ear contours.

3.4 3D Ear Landmarking on Headspace

We apply the HERA system on the *Headspace* dataset [106] of 3D human head images and thereby equip that dataset with a set of 55 landmarks per ear and associated confidence values. Thus we demonstrate the high utility of the HERA system which, due to its inherent 3D reconstruction property, is able to operate on both 2D and 3D datasets. To localise these 3D ear landmarks for raw scans of complete human heads, the HERA system is used to generate an initial set of 3DMM shape and pose parameters, which are then refined in an optimisation stage. In this section, the *Headspace* dataset will be introduced first, followed by the methods applied to obtain 3D ear landmarks on raw data. Finally, we evaluate our resulting 3D ear landmarks. Our automatically generated 3D ear landmark set for Headspace has great utility in full head 3DMM fitting and will be made publicly available, along with the HERA code repository.

3.4.1 Object Detection

Object detection is a task that detects a bounding box for specific objects in an image [114]. We summarise this task in this section because we use it to extract ear images from full head images. It forms a part of our pipeline for ear image analysis and is essential to reduce the complexity of the problem by enabling us to train the system on ear images only. We use YOLO-v3 [115] as our object detector. It generates ear bounding boxes on full head images, and the ear images can be fed into the next stage of the pipeline.

3.4.2 Obtaining Correspondence Between Meshes

Obtaining dense point-to-point correspondence is a frequently employed task to organise raw scans in the same vector space [88]. This task is important for transforming unordered, variable size point clouds so that they can be organised and analysed as a whole. The 3D ear model fitting task in Sec. 3.4 is a special case of the correspondence establishment task. Normally the dataset over which to establish correspondence contains only one type of object. While in our task, we are trying to find the correspondence between a 3D ear model and a 3D full head mesh. While numerous methods focus on finding correspondences between different types of surfaces using traditional or deep learning techniques [116, 117, 88, 118, 119], we choose to root our correspondence finding method with a traditional iterative approach, Iterative Closest Points (ICP) [120]. The core idea behind this method is to iteratively calculate correspondences and solve rigid transformations, yielding better correspondences and more accurate transformations over iterations. We extend this method to jointly solve rigid transformations and model fitting at the same time using a general-purpose nonlinear optimiser named the

Nelder-Mead method [121].

Our task is highly related to part-to-whole registration task where different methods are proposed [5, 4, 122]. Tan *et al.* [4] propose the first CNN to tackle local shape deformation problem, therefore achieve the goal of the part-to-whole registration task. Yang *et al.* [5] propose a multi-scale method that can perform mesh deformation in a coarse-to-fine manner, allow registration of sub-parts. The approach proposed by us is more traditional where an optimisation pipeline is used but a good initialisation is provided by our HERA system.

3.4.3 Headspace Dataset

The Headspace dataset [106] is a set of 3D images of full human heads captured by the *3dMDhead* system, which has five 3D cameras for full head coverage. The data are collected by taking five 3D images of the subject, which are stitched together into a single 3D image. The subject's cranial shape is revealed, as they all wear a close-fitting latex cap. The dataset contains 1519 subjects in total, and 3D facial landmarks are available for 1212 of them. We apply our method to the 1212 subjects where 3D facial landmarks are already available. Also, since we render the 3D meshes into 2D images during our proposed method, we further filter the subjects where a texture is not publicly available, which results in 1002 subjects. Finally, since we use a 2D ear detection algorithm to locate the ears in the rendered 2D images, a further 25 ears are discarded, since no ear was detected. A typical case for this failure is when the ear region is not imaged and reconstructed correctly, typically due to occlusion by hair. This can result in missing ear parts in the reconstructed 3D mesh, causing the 2D ear detection algorithm fail.

3.4.4 Head Pose Normalisation and 2D Image Rendering

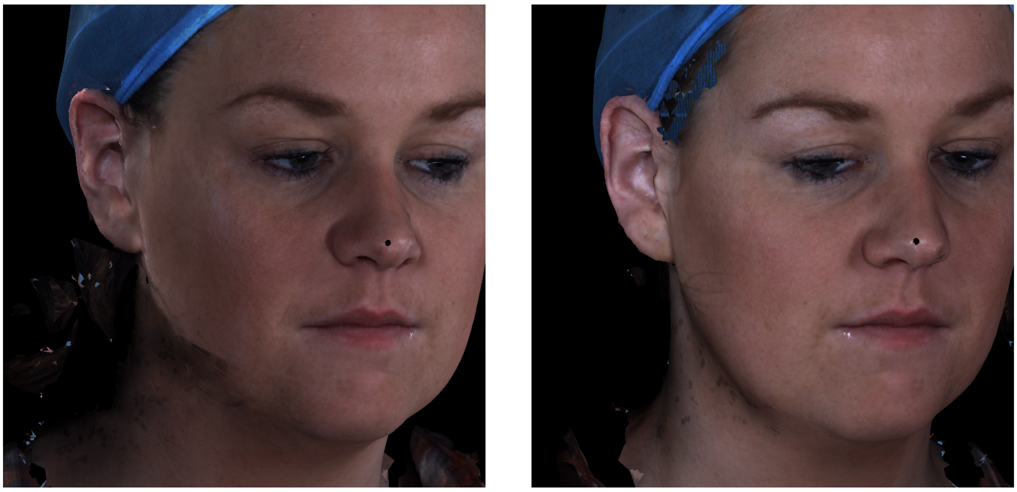
The 3D head meshes from the Headspace dataset are in non-standard and varied poses. In order to consistently produce 2D head images, so that there is one ear clearly visible and well-posed in each 2D image, 3D pose normalisation is the first step. This pose normalisation has four steps. Firstly, the 3D head is rotated such that the face is parallel to the z-plane. To achieve this, we first define two vectors that form the face plane. The two vectors are both originating from the lip centre, pointing to left eye corner and right eye corner respectively. These key points are defined by the provided 52 3D face landmarks with each Headspace scan. Then, we find an estimate of the face plane normal using the cross product these two vectors, and finally we find the rotation matrix $\mathbf{R}_z$ between the face plane normal and the z-plane normal. The second step is to find another rotation matrix to make the face *upright* (where the y-axis direction is the upright direction). For the 3D head, again we use the provided 3D landmarks to get a vector originating from the chin tip and pointing to the top most point on nose ridge. With the two vectors, another rotation matrix $\mathbf{R}_u$ can be computed to finalise the head orientation. The third step is to find a translation, $\mathbf{T}$. We define the face centre as the mean of all 52 face landmarks, the translation $\mathbf{T}$ is then computed to move the face centre to the origin. Finally, we work out a scale $s = \frac{1}{2L_f}$ to normalise the size of all faces, where L_f is the width of the face calculated using the face landmarks.

Denoting a 3D head mesh's vertices from the dataset as $\mathbf{X}' \in \mathbb{R}^{N\times3}$ where N is the number of vertices, we summarise the above steps as applying a similarity transformation to get a pose-normalised profile-view 3D head mesh

$\mathbf{X}'_{norm}$ via:

$$\mathbf{X}'_{norm} = s\left(\mathbf{X}'\mathbf{R}_z^T\mathbf{R}_u^T + \mathbf{1}_N\mathbf{T}^T\right) \tag{3.18}$$

where $\mathbf{1}_N$ is an N-vector of ones. The resulting well-posed 3D head is then imaged by an orthographic camera, placed at $(0, 0, -1)^T$. The projection model is the same as used in the HERA system for its simplicity and to align with the HERA assumption, $\mathbf{P}_o$. Then the head is rotated along y-axis by 0.37 radian (21 degrees) to reveal the right ear in a pose that is consistent with the pose that the HERA system is primarily trained with (*i.e.* most of the in-the-wild ear training images have head poses that are close to 20 degrees). Since the HERA system is trained to process right ear images only, we produce the left ear by y-plane reflection of the 3D head before the rotation. Finally, we render two images each with 1024 × 1024 pixels using the orthographic camera model for both ears. An example of the right ear image and the reflected left image of subject number 3 is shown in Fig. 3.9.



(a) Generated 2D image of the left ear (after reflection).

(b) Generated 2D image of the right ear.

Figure 3.9. Rendered 2D images from pose normalised 3D head images with both left and right ears of subject 3

3.4.5 YOLO ear detection and cropping

Since the HERA system processes 2D ear images, an ear detection is employed to generate a region-of-interest (ROI) bounding boxes on the rendered head images. We employ YOLOv3 [115], a general object detection algorithm for this task. We train the YOLOv3 net using the 500 ear images from the ITWE-A dataset, the ear detection results show that the model detects right ears sufficiently well. It only fails to detect 25 images out of 2004 2D head images (two per Headspace subject). Fig. 3.10a shows the detected ear region of subject number 3. The major failure reason is because of missing ear data in the reconstructed 3D head, which is typically due to occluding hair.

3.4.6 Landmark initialisation using the HERA system

The cropped ear images are fed to a trained HERA system, which generates a set of York Ear Model (YEM) latent parameters to reconstruct the 3D ears and their pose. For each reconstructed ear and ear pose, we get 55 3D ear landmarks. Note we use head vertices instead of any points on head mesh surface to represent 3D ear landmarks, this is acceptable because the head vertices are relatively dense. There are two problems to solve; i) since these are vertices on the reconstructed ears instead of the 3D head, we have to map them to 3D head vertices; ii) the orthographic camera that HERA employs induces an ambiguity in the distance from the ear to the camera. Thus we can only use the (x, y) coordinates as reliable landmark locations. We solve both of these problems with a single solution. Firstly, both the vertices from the 3D head and the points from the initial 3D ear landmarks are projected to a 2D plane using the camera projection matrix $\mathbf{P}_o$. We denote the 3D head vertex indices as $\mathbf{i}_X \in \mathbb{N}^N$, the 3D head vertices as $\mathbf{X}'$

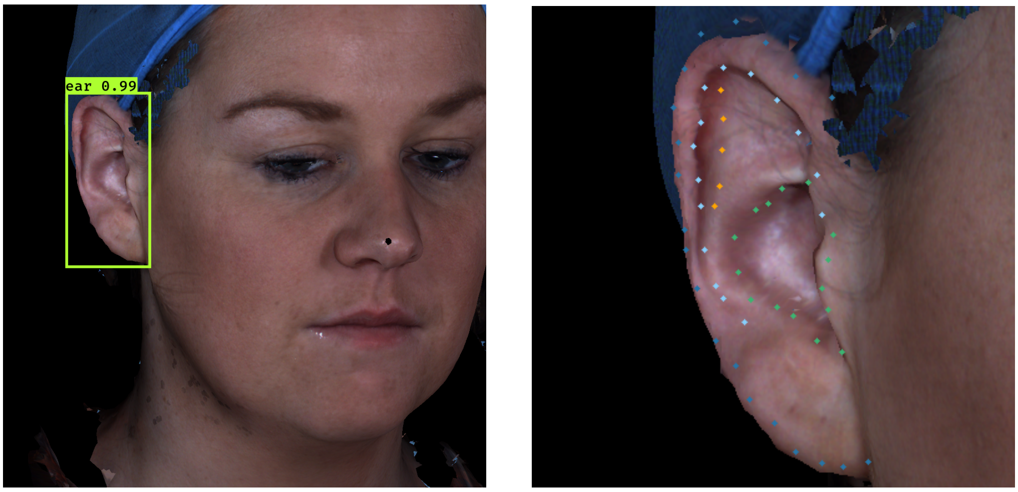
and the projected initial 3D ear landmarks as $\mathbf{L}_P = \begin{bmatrix}\mathbf{l}_0 & \dots & \mathbf{l}_{54}\end{bmatrix}$. Then, for each projected landmark $\mathbf{l}_t \in \mathbf{L}_P$, we find k nearest-neighbour vertices from the 3D head and denote their indices as $\mathbf{i}^t = i_1^t \dots i_k^t$. Here we essentially find first k vertices with closest euclidean distance to each projected landmark. Note that the Euclidean distances are calculated in 2D image plane. Finally, from that set, we select the index i^* whose corresponding vertex is the closest to the camera plane, *i.e.* has the minimum z-coordinate value:

$$i^* = \arg\min_i \mathbf{X}'(i) \cdot \begin{bmatrix}0 & 0 & 1\end{bmatrix}^T. \tag{3.19}$$

With such a procedure, we effectively select vertices that are closest to the camera (more likely to be vertices on ears) and ignore vertices that are on the back of the head. Thus practically reliable initial ear landmarks that are on ears can be selected. We find that searching for the $k = 10$ nearest neighbours works consistently in our Headspace experiments. For datasets other than Headspace, increasing the value of k can mitigate situation where the initial landmark predictions are of lower accuracy. Fig. 3.10b shows the generated initial 3D landmarks on a Headspace subject.

3.4.7 Ear Model Fit Refinement with Iterative Optimisation

With the initial ear landmarks on the 3D head, using the initial YEM latent code and the initial ear pose, we adopt an iterative approach to refine the fitting of the ear model. Since the 3D data are available, we can perform fine adjustment of the initialised 3D ear model to the 3D head data directly. We adopt an iterative 3D model fitting procedure where model parameters, pose, scale and dense 3D correspondences between the ear model and the



(a) Detected right ear

(b) HERA 3D landmark initialisation

Figure 3.10. Ear detection result and HERA initialisation of subject number 3.

Headspace scan ears are iteratively refined. The algorithm is outlined in Algorithm 1. The algorithm essentially optimises the ear model and the correspondence between the ear model and the ear of the head scan in turn. The algorithm's iteration consists of three steps: i) reconstruct the ear shape and compute the required similarity transform using initial parameters or optimised parameters from the previous iteration; ii) find the correspondences

Algorithm 1 Iterative optimisation of the ear model

```
Ensure: $\mathbf{R} \in \mathrm{SO}(3)$
  $t \leftarrow 0$
  $\mathbf{R}^t \leftarrow \mathbf{1}$                                  ▷ Identity matrix.
  $\mathbf{T}^t \leftarrow \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T$
  $\boldsymbol{\alpha}_s^t$                     ▷ HERA initialised YEM shape parameters.
  $f^t \leftarrow 0.0$
  while $t < T$ do
      $\mathbf{X}_e^t \leftarrow f^t \mathbf{R}^t \mathrm{YEM}\left(\boldsymbol{\alpha}_s^t\right) + \mathbf{T}^t$
      $\mathbf{X}_h^t \leftarrow \mathrm{MutualNN}\left(\mathbf{X}_e^t, \mathbf{X}_h\right)$          ▷ Find correspondences.
      $\mathbf{R}^{t+1}, \mathbf{T}^{t+1}, f^{t+1}, \boldsymbol{\alpha}_s^{t+1} = \min \left\|\mathbf{X}_e^t - \mathbf{X}_h^t\right\|_2^2$
      $t \leftarrow t + 1$
  end while
```

(*i.e.* mutual nearest neighbours) between the reconstructed and transformed ear shape and the raw head shape by the mutual nearest neighbour method; iii) run the nonlinear optimiser to optimise a new set of model and similarity transform parameters, given the correspondences.The function 'MutualNN' takes two point clouds and returns a set of paired vertices such that each paired vertices $A \in \mathbf{X}_e^t, B \in \mathbf{X}_h$ satisfies following conditions:

$$B \equiv \min_{\hat{B} \in \mathbf{X}_h} \|\hat{B} - A\| \tag{3.20}$$

$$A \equiv \min_{\hat{A} \in \mathbf{X}_e^t} \|\hat{A} - B\|. \tag{3.21}$$

We also expand our method from processing 3D vertices to processing 6D vertices, by appending a weighted normal to each vertex. The weight is choosen empirically to balance the numerical values of vertex coordinates and unit vectors. In our experiments, we empirically choose to iterate the refinement loop five times for it to finish running in a practically reasonable time after the algorithm gradually converges (*i.e.* parameter update becomes insignificant)..

3.4.8 Evaluation on Headspace data

In this section, we quantitatively evaluate our Headspace results using both 3D model fitting error and reliability of fitted 3D landmarks. Also, we show a number of qualitative results to illustrate both typical results and worst-case results.

3D model fitting We evaluate our HERA initialised 3D model fitting using a modified Chamfer distance between the fitted 3D ear vertices and the raw 3D head vertices. The key modification is to ignore the vertices from the raw 3D head that are far away from the region of interest. This is implemented by rejecting *head-to-ear* distances that are larger than the longest *ear-to-head* distances. Where *head-to-ear* distances is defined as: for all vertices in the head mesh, calculate their shortest distances to the ear mesh. And vice versa for *ear-to-head* distances. Details are explained as follows. We denote fitted 3D ear vertices as $\hat{\mathbf{X}}_e \in \mathbb{R}^{N_e \times 3}$ and raw 3D head vertices as $\mathbf{X} \in \mathbb{R}^{N \times 3}$, where N_e is the number of vertices of the YEM template and define the distance function from ear to head $D_h : \hat{\mathbf{X}}_e \mapsto \mathbb{R}$ following the usual Euclidean distance as:

$$D_h\left(\mathbf{P}_e\right) = \min_{\mathbf{p}\in\mathbf{X}} \left\|\mathbf{P}_e - \mathbf{p}\right\|, \tag{3.22}$$

where $\mathbf{P}_e$ is a vertex from ear vertices $\hat{\mathbf{X}}_e$. Then we define the modified distance function from head to ear $D_e : \mathbf{X} \mapsto \mathbb{R}$ to reject irrelevant head vertices (*e.g.* neck, eye, nose, the other ear, etc.):

$$D_e\left(\mathbf{P}_h\right) = \begin{cases} \min_{\mathbf{p}\in\hat{\mathbf{X}}_e} \left\|\mathbf{P}_h - \mathbf{p}\right\|_2 & \text{if } \min_{\mathbf{p}\in\hat{\mathbf{X}}_e} \left\|\mathbf{P}_h - \mathbf{p}\right\|_2 \\ & \quad \leq \max_{\mathbf{q}\in\hat{\mathbf{X}}_e} D_h\left(\mathbf{q}\right) \\ 0 & \text{otherwise,} \end{cases} \tag{3.23}$$

where $\mathbf{P}_h$ is a vertex from head vertices $\mathbf{X}$. Finally, the modified Chamfer distance E_c between $\hat{\mathbf{X}}_e$ and $\mathbf{X}$ is:

$$E_c = \frac{1}{N_e} \sum_{\mathbf{p}\in\hat{\mathbf{X}}_e} D_h\left(\mathbf{p}\right) + \frac{1}{N'} \sum_{\mathbf{p}\in\mathbf{X}} D_e\left(\mathbf{p}\right), \tag{3.24}$$

where N' is the number of head vertices below the threshold in Eq. (3.23). We use the average E_c for all fitted ears as our evaluation metric for 3D ear model fitting results. To the best of our knowledge, this is the first work on

	HERA Initialisation		Refined	
	mean ± std	median	mean ± std	median
Left Ear	7.27 ± 2.02	6.84	$\mathbf{4.75 \pm 0.78}$	**4.66**
Right Ear	7.42 ± 1.99	7.10	$\mathbf{4.76 \pm 0.80}$	**4.66**
Total Ear	7.34 ± 2.00	6.94	$\mathbf{4.75 \pm 0.79}$	**4.66**

Table 3.3. Chamfer distance result (mm) for left and right ear and both combined. Comparison between HERA initialisation and iteratively refined results are shown, too.

ear model fitting to the Headspace dataset. Therefore, in Tab. 3.3, we report the evaluation results on the HERA initialised model alongside those of the final fitted model after the 3D refinement optimisation. It is also possible to use one-way distance to avoid using the modified Chamfer distance, but at a cost of losing certain amount of information in evaluations.

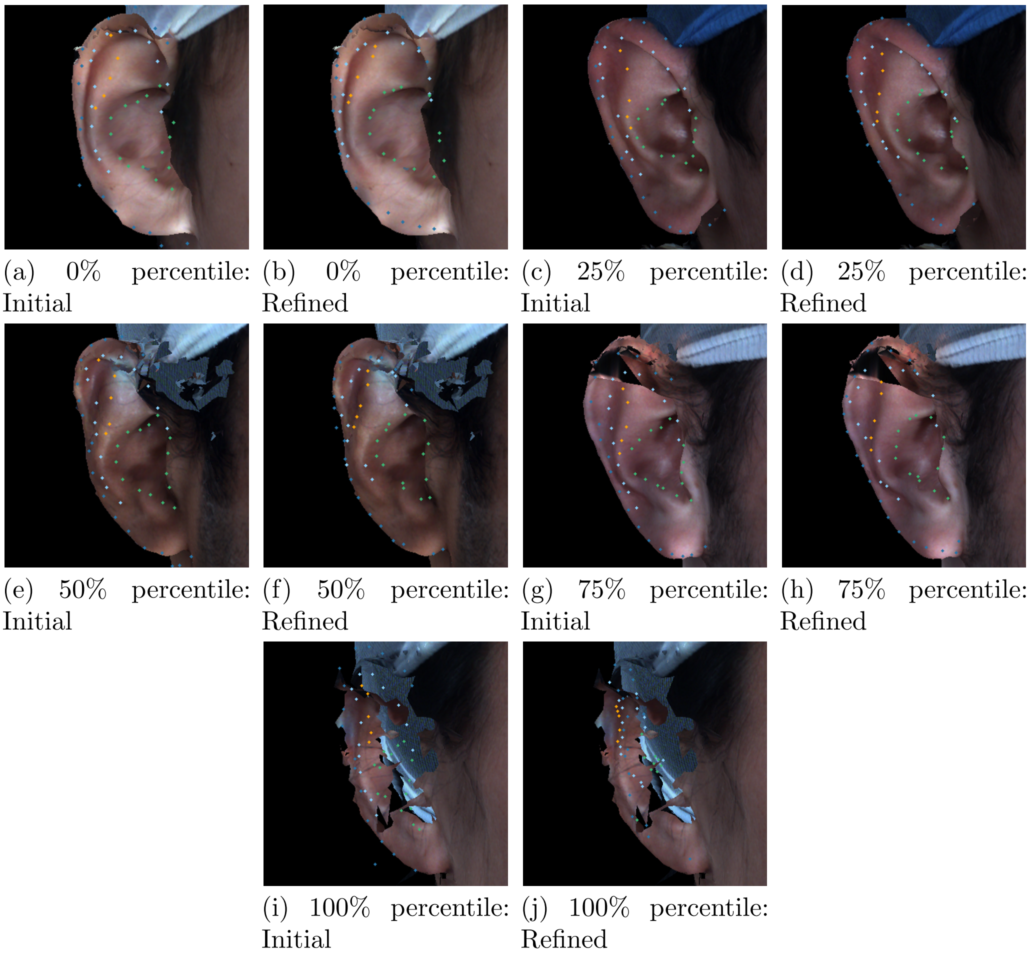
3D ear landmarks The HERA system allows the Headspace data to be landmarked with 55 landmarks per ear. We augment these head scan landmark indices with their residual Euclidean distances from the fitted ear model, as defined by Eq. (3.22). This effectively provides a reliability measurement for each fitted 3D ear landmark on each fitted ear, enabling further applications to employ landmark confidence measures. For example, in a full head model fitting application, ear landmarks may be weighted using a Gaussian weighting function based on their respective distance errors. Using Eq. (3.22), but further restricting the function's domain to the 55 landmarks on the ear vertices, we report an average landmark Euclidean distance of $1.731\,mm$.

Qualitative evaluation We sort the refined Chamfer distance in ascending order and select the nearest sample from the ranking at the following percentiles: 0% (best result), 25%, 50% (median result), 75%, 100% (worst result), and show the examples in Fig. 3.11.

We show one successful example and one failed sample of our iterative fine fitting step in Fig. 3.12. As shown by the successful example, the initial HERA prediction is shown on the top-left image and the final fine fitted 3D landmarks are shown on the top-left image. By comparing the two images, one can find that although there is a small compromise on the top part of the ear, the ear contour after fine fitting is much improved compared to the initial one, especially for the ear lobe area. For the failed example where the raw data has a significant portion of missing data, the fine-fitting fails to generate reasonable 3D ear landmarks. Being able to identify such incomplete data and failed example automatically can be an important future work.

3.5 Conclusion

We have built an end-to-end deep 3D ear reconstruction autoencoder system that can successfully fit a 3D ear model to a single 2D image, and can potentially be trained unsupervised. Our model reconstructs the 3D ear mesh with a plausible appearance and accurate dense alignment, as witnessed by the accurate alignment compared to ground truth landmarks. A comprehensive evaluation shows that our method achieves state-of-the-art performance in 3D ear reconstruction and alignment. We have shown how this system can be employed to initialise a model fitting of ears to raw 3D head images and thereby apply automatic 3D landmarking to those 3D images. We also generate predicted 3D ear landmarks for almost 2K ears over almost 1K subjects in the Headspace dataset, with residual fitting errors for confidence weighting estimation, which can support further applications on this dataset, such as landmark-guided full head 3DMM fitting. For future work, it is worthwhile to compare the HERA initialised 3D model fitting with part-to-whole methods to further analyse the approach.



(a) 0% percentile: Initial

(b) 0% percentile: Refined

(c) 25% percentile: Initial

(d) 25% percentile: Refined

(e) 50% percentile: Initial

(f) 50% percentile: Refined

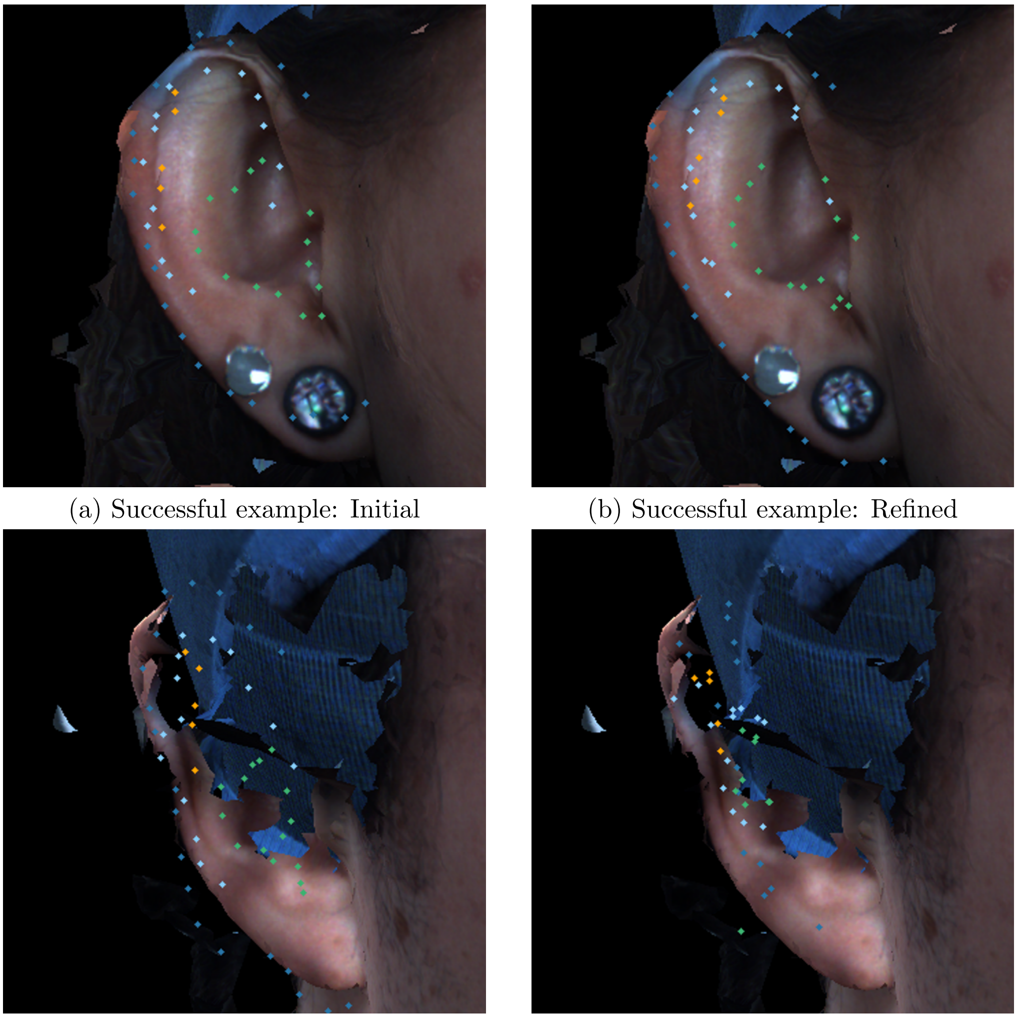
(g) 75% percentile: Initial

(h) 75% percentile: Refined

(i) 100% percentile: Initial

(j) 100% percentile: Refined

Figure 3.11. Qualitative results at a variety of percentiles (please zoom in for detail). Note that for the 0%, 25% and 50% percentiles, the refinement stages drags the landmarks that are not on the ear onto the ear. Also observe that in the cases of 25%, 50% and 75%, we see that the refinement makes part of the ear contour more accurate. At 75%, we see that the ear has a minor missing part, causing a performance drop, but within reasonable range. Finally at 100% the ear has a major missing part, causing significant degradation in performance and subsequently it has the largest Chamfer distance.



(a) Successful example: Initial

(b) Successful example: Refined

(c) Failed example: Initial

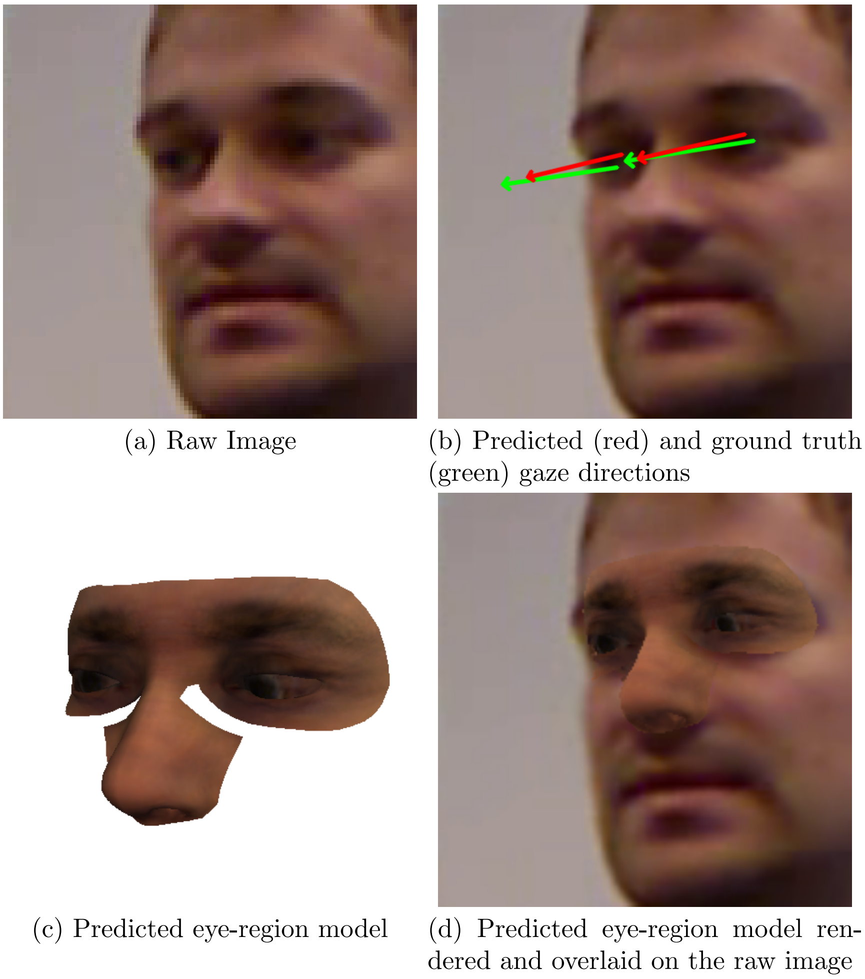
(d) Failed example: Refined

Figure 3.12. Rendered images from normalised Headspace 3D head with both left and right ears of subject number 3

4 Eye-region Reconstruction Autoencoder for Accurate Gaze Estimation

4.1 Introduction

Human eye gaze has a significant role in the visual understanding of human intention and has high utility in a variety of important applications; for example, in domains such as human-computer interaction [123] and virtual reality [124]. Several previous works have built an eye-region model [65], or a full head model [61], that can be fitted to given images, which thereby provides a gaze direction estimation. Also, appearance-based methods that regress gaze directions directly from RGB input images using deep neural networks without the use of 3D shape models have been increasingly popular [8]. We note that, compared to these appearance-based methods, model-based methods are less competitive in regard to gaze estimation accuracy. This is because of the deep neural network's feature extraction and nonlinear fitting ability. However, most appearance-based gaze estimation methods predict only a gaze direction (i.e. azimuth-elevation rotational orientation), but no other information about the 3D geometry of the gaze or the eye-region. Current literature has different gaze origin representations (*e.g.* eyeball centres or a point on the face), which requires additional effort to make performance comparisons [125].



(a) Raw Image

(b) Predicted (red) and ground truth (green) gaze directions

(c) Predicted eye-region model

(d) Predicted eye-region model rendered and overlaid on the raw image

Figure 4.1. Active-gaze 3DMM fitting example. (a) Raw input image, (b) predicted gaze directions (red) compared to ground truth (green), (c) predicted eye region model, (d) eye region model overlaid on raw input image.

We propose an end-to-end method that combines both appearance-based and model-based elements and is trained in an end-to-end manner. Our method reconstructs the 3D eye-nose region and thereby avoids the highly-variable mouth-jaw area, so that it can more accurately predict gaze direction. In order to achieve this, we employ an *eyes-and-nose* 3D morphable model (3DMM) and, crucially, we equip this with a geometric vergence model of gaze. We call this an '*active-gaze* 3DMM'. Specifically, this enables the combined rotation of the eyeballs for the expression of gaze under certain geometric constraints, such as coplanarity of the gaze vectors. This ensures both accurate gaze estimation and that the eyeball positions are consistent with both the face geometry and head pose. As a result, we can model the correlations between the face and the left and right eyeballs, without additional design of the neural network, and we only require face image inputs. An example of the input and outputs is shown in Fig. 4.1.

Most of the current image-to-image 3D reconstruction methods from monocular RGB images focus on faces [6, 31]. Typically, their 3D face models only model the eyeball surface area as part of the face, and the gaze directions are not explicitly modelled. Our method both takes advantage of the image-to-image architecture *and* models the specific eye-region area, designing gaze information into the model. Our results show that predicting both gaze direction and the eye-region model results in a significant improvement in gaze estimation accuracy. In summary, our main contributions are:

1. Development of an eye-region 3DMM fitting process that is trained end-to-end without additional manually-annotated ground truth labels.

2. Development of an active-gaze 3DMM, which equips the regular 3DMM with a geometric eye vergence model in order to regularise network training.

3. Demonstration that the active-gaze 3DMM increases gaze estimation accuracy and the method's versatility.

4. Demonstration of our method's adaptability when only ground truth 3D gaze targets are available, with no access to gaze origin information.

4.2 Proposed method

In this section, the overall architecture will be described first. Then we will elaborate each component following the order of the whole pipeline. At the end of each subsection, the outputs and loss function terms relating to the outputs are presented.

4.2.1 Architecture

As shown in Fig. 4.2, the raw image $\mathbf{I}$ is firstly fed to the encoder to regress eye-region reconstruction parameters $\mathbf{z}_M$ and eye rotation parameters $\mathbf{z}_E$. The eye-region reconstruction parameters are defined as follows:

$$\mathbf{z}_M = (\mathbf{z}_S, \mathbf{z}_A, \mathbf{r}, \mathbf{T}, f)^T$$

where $\mathbf{z}_S$ are shape parameters, $\mathbf{z}_A$ are texture parameters, $\mathbf{r}, \mathbf{T}$ are head pose parameters describing rotation and translation respectively and f is the scale factor of the imaging projection.

We use the Swin Transformer [126] as our encoder network, the details will be elaborated in Section 4.2.2. The eye-region reconstruction parameters $\mathbf{z}_M$ are used to reconstruct a textured eye-region 3D mesh, thus providing predicted 3D eyeball centres as gaze origins (eyeball vertex means), and a set of 2D projected landmarks for eye-region alignment. We discuss the

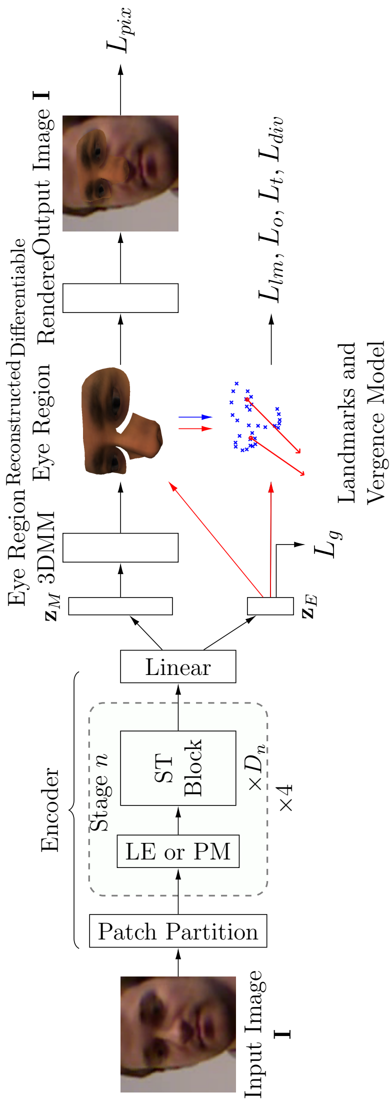


Figure 4.2. Overview of gaze estimation using our active-gaze 3DMM autoencoder. We employ the tiny version of Swin Transformer for our encoder, which has four stages. *LE* stands for Linear Embedding, which is used in stage one only, and *PM*, standing for Patch Merging, is used in stages $2-4$. *ST Block* stands for Swin Transformer block. Red points are in the 3D camera coordinate system, while blue points are on the 2D image plane. The 'L' terms show where the various loss function components are generated.

eye-region reconstruction in Section 4.2.3. The eye rotation parameters $\mathbf{z}_E$ predict the gaze vectors for both eyes. Using the gaze origins and gaze vectors, we employ a geometric vergence model to constrain the gaze directions of both eyes jointly. This is detailed in Section 4.2.4. Additionally, we use a differentiable renderer to create a rendered image for autoencoder-based pixel-wise comparison. The differentiable renderer is introduced earlier in the Section 3.2.4

4.2.2 Encoder

We employ the state-of-the-art Swin Transformer [126] as our encoder to regress eye-region features. The input RGB image is divided into non-overlapping patches by the patch partition module, where each patch is considered as a *token*. This is followed by four stages of modified self-attention computation (*i.e.* Swin transformer blocks) and we define D_i as the number of repetitive Swin transformer blocks at stage i. We use the *Tiny* network structure provided by the authors, whose $D_{1...4} = (2, 2, 6, 2)$. For the first stage, the linear embedding module is applied before the transformer blocks, and for the other three stages, a patch merging module is applied before each set of transformer blocks to reduce the output dimensionality. These four stages jointly produce a feature map that is appended by a linear layer to regress a semantically-meaningful feature vector.

The regressed feature vector is then divided into two parts: eye-region reconstruction parameters $\mathbf{z}_M$ and both eyes' gaze directions $\mathbf{z}_E$ defined by azimuth and elevation. The first part of the features is elaborated in Section 4.2.3, and the second part, which is used in constructing the geometric vergence constraints, will be described in Section 4.2.4.

4.2.3 Eye-region 3D morphable model (3DMM)

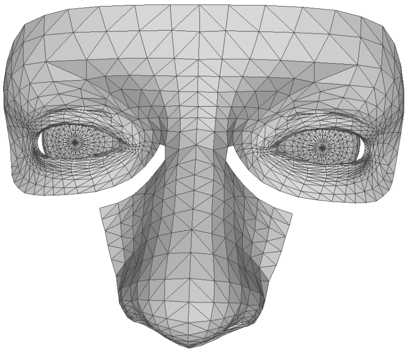


Figure 4.3. The mean eye-region mesh, extracted from the FLAME model [3] and incorporated into our active-gaze 3DMM fitting system, which has rotatable eyeballs.

In this section, the process of reconstructing and rendering the 3D eye-region from the eye-region reconstruction parameters $\mathbf{z}_M$ will be discussed. The eye-region 3DMM is constructed by selecting the relevant vertices and their topology from the full head FLAME [3] model. As shown in Fig. 4.3, both eyeballs, the eye-region and the nose are selected. Eyeballs are used to model gaze directions, eyeball sizes, and inter-ocular distances. The eye-region contains 22 landmarks on eyebrows and eye contours, which is used to model eyeball positions and head poses. A visualisation of the 22 landmarks can be found in Fig. 4.2 and in Fig. 4.5. We omit the remaining parts of the FLAME head model, firstly to enable a more compact and efficient learning process, and secondly since they have much higher variance in features (*e.g.* mouth/jaw variations due to speech and/or facial expressions) that are not relevant to gaze modelling, and may indeed introduce confounding factors. Notably, the largely rigid nose area, which contains nine landmarks on the nose ridge and the philtrum area, is added to strengthen the head pose prediction. Note that the gap between the eyeball and the eye contour does not affect the model learning process. We also use the texture model presented by [127, 128] to enable differentiable rendering of the eye-region model.

We follow the common procedure to reconstruct the 3DMM's shape $\mathbf{S} \in \mathbb{R}^{N\times 3}$ from shape parameters $\mathbf{z}_S$ and the texture $\mathbf{A} \in \mathbb{R}^{512\times 512\times 3}$ from texture parameters $\mathbf{z}_A$ as follows:

$$\mathbf{S} = \boldsymbol{\mu}_S + \mathbf{U}_S\mathbf{z}_S \tag{4.1}$$

$$\mathbf{A} = \boldsymbol{\mu}_A + \mathbf{U}_A\mathbf{z}_A, \tag{4.2}$$

where N is the number of vertices in the eye-region shape model, $\boldsymbol{\mu}_{\{S,A\}}$ and $\mathbf{U}_{\{S,A\}}$ are the mean and principal components provided by the shape and texture 3DMMs respectively. Then the eye-region shape $\mathbf{S}$ is transformed with rotation $\mathbf{R}$, translation $\mathbf{T}$ and scale f to the camera coordination system by:

$$\mathbf{S}' = f\mathbf{S}\mathbf{R}^T + \mathbf{1}\mathbf{T}, \tag{4.3}$$

where $\mathbf{R} \in \mathrm{SO}\,(3)$ is the rotation matrix derived from the rotation $\mathbf{r}$ by Rodrigues' rotation formula and $\mathbf{1} \in \mathbb{R}^{N\times 1}$ is the vector of all ones. Finally, given the camera calibrations are available, we construct a full perspective projection $\boldsymbol{\Pi} \in \mathbb{R}^3 \rightarrow \mathbb{R}^2$ to project the eye-region shape in 3D camera space $\mathbf{S}'$ to image plane, thus obtaining the predicted 2D landmarks $\hat{\mathcal{L}}$ on image plane. We use a differentiable renderer DR implemented by PyTorch3D [2], with the same projection model, to form a rendered image $\hat{\mathbf{I}}$ as:

$$\hat{\mathbf{I}} = \mathrm{DR}\left(\mathbf{S}', \mathbf{A}, \boldsymbol{\Pi}\right). \tag{4.4}$$

Note that all previous works of 3D face reconstruction that involve differentiable rendering assume a Lambertian surface, which is not well-suited to the eyeball surface due to it's inherent moisture, which causes specularities. Our further experiments show that the geometric vergence constraints contribute the most to gaze estimation accuracy, thus we choose the ambient Phong lighting model and leave the discussion of more refined eye-region modelling in our *limitations* section.

With the reconstructed 3D eye-region, we form the 3D gaze origin loss function L_o as

$$L_o = \|\hat{\mathbf{o}} - \mathbf{o}\|_1^1, \tag{4.5}$$

where $\mathbf{o}$ is a 3D ground truth gaze origin provided by the dataset and $\hat{\mathbf{o}}$ is some point derived by the eye-region shape. For example, a predicted eyeball centre is obtained by averaging all eyeball vertices. With such a design, our method becomes universally applicable to any gaze origin definition, as provided by the dataset; for example, both eyeball-centered and face-centered have been used in the literature. This obviates the conversion step described by Chen *et al.* [125] that converts gaze ground truth between datasets using different gaze representations.

With the projection model, 2D projected landmarks can be obtained, thus forming the 2D landmark loss function L_{lm} as:

$$L_{lm} = \|\hat{\mathcal{L}} - \mathcal{L}\|_2^2, \tag{4.6}$$

where $\mathcal{L}$, the ground truth 2D landmarks, are either provided by the dataset or generated before training using PyTorch Face Landmark [129] with a pre-trained MobileNetV2 [130] as the backbone network. The predicted 2D landmarks $\hat{\mathcal{L}}$ are obtained by projecting selected vertices in the eye-region shape

$\mathbf{S}'$ onto the image plane via the perspective projection, $\mathbf{\Pi}$. We employ the Multi-PIE [131] definition of 68 facial landmarks and select 31 corresponding points on both the eye-region 3DMM and the input images. We use perspective projection in this work since the camera parameters are available.

Finally, the pixel loss L_{pix} for rendered eye-region images is formed as:

$$L_{pix} = \|\hat{\boldsymbol{I}} - \boldsymbol{I}\|_2^2. \tag{4.7}$$

4.2.4 Vergence model

The predicted gaze rotations $\mathbf{z}_E = (\mathbf{r}_l, \mathbf{r}_r)^T$ by the encoder are azimuths and elevations for both eyes (*e.g.* $\mathbf{r}_l = (r_{le}, r_{la})^T$). Two rotation matrices $\mathbf{R}_{\{l,r\}}$ are derived from the rotation angles by Rodrigues' rotation formula. We assume the gaze direction is a vector originating from the centre of the eyeball, and pointing towards the iris centre. Thus, the gaze vectors for both eyes are calculated by: $\mathbf{g}_i = \mathbf{R}_i \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T, i \in \{l, r\}$. They originate from both eyeballs' centre $\mathbf{o}_{\{l,r\}}$ respectively. The eyeball rotation matrix is also applied to the eyeball shapes of the reconstructed 3D eye-region shape to rotate the eyeballs to produce a plausible appearance.

As shown in Fig. 4.4, we equip our system with geometric constraints that capture both gazes in one system, such that both of the gazes are mutually constraining each other, and produce a gaze target $\hat{\mathbf{t}}$. Due to the nature of human gazes, there are three underlying constraints for this vergence model and which a single-eye model does not have: i) both gaze vectors are directed away from the head; ii) the gaze vectors are coplanar; iii) the gaze vectors intersect at the gaze target $\hat{\mathbf{t}}$, unless they are parallel. These three constraints can be satisfied during the process of calculating the gaze target $\hat{\mathbf{t}}$, which is defined as the closest point between the two gaze vectors. We define

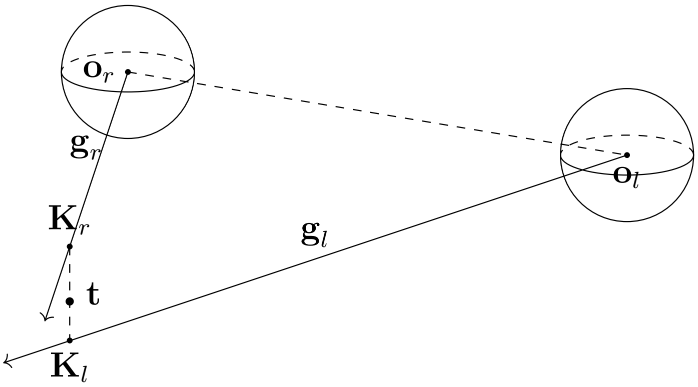


Figure 4.4. The vergence model of gaze for the active-gaze 3DMM, showing eyeball origins ($\mathbf{o}_{l,r}$), gaze directions ($\mathbf{g}_{l,r}$) and viewing target $\mathbf{t}$ in the global camera frame. In general, the regressed gaze directions are skew and the loss function penalises this lack of coplanarity.

$\mathbf{K}_i = \mathbf{o}_i + k_i\mathbf{g}_i, i \in \{l, r\}$ as the two end points of the shortest segment connecting left and right gazes. Therefore,

$$\hat{\mathbf{t}} = (\mathbf{K}_l + \mathbf{K}_r)/2. \tag{4.8}$$

Since the shortest segment must be perpendicular to both gaze vectors, we can derive the shortest distance d as:

$$d := \|\mathbf{K}_l - \mathbf{K}_r\| = k_{lr}(\mathbf{g}_r \times \mathbf{g}_l), \tag{4.9}$$

where k_l, k_r and k_{lr} can be solved by:

$$\begin{bmatrix} k_l & k_r & k_{lr} \end{bmatrix}^T = \begin{bmatrix} \mathbf{g}_l & -\mathbf{g}_r & \mathbf{g}_r \times \mathbf{g}_l \end{bmatrix}^{-1} (\mathbf{o}_r - \mathbf{o}_l). \tag{4.10}$$

We design three loss terms based on the underlying constraints of the geometric vergence model. Firstly, the gaze skew loss, $L_{skew} = d^2$, encourages the two gaze vectors to be coplanar. Secondly, the predicted gaze target, $\hat{\mathbf{t}}$, along with the 3D ground truth target, $\mathbf{t}$, forms a gaze target loss $L_t =$

$\left\|\hat{\mathbf{t}} - \mathbf{t}\right\|_1^1$. Finally, a gaze pose loss is given as $L_g = \|\mathbf{z}_E - \mathbf{r}_{gt}\|_1^1$, where $\mathbf{r}_{gt}$ is the ground truth eyeball rotation. All of these losses reduce gaze error, while additionally preventing the catastrophic case of the gaze being directed into the head.

4.2.5 Complete loss function

In addition to the previously stated loss function terms, we employ a regulariser on the 3D eye-region shape and texture latent code $\mathbf{z}_S$ and $\mathbf{z}_A$ to encourage the reconstructed eye-region shape and texture predictions to stay within the model space. The regulariser is defined as follow:

$$L_{reg} = \|\mathbf{z}_S\|_2^2 + \|\mathbf{z}_A\|_2^2 . \tag{4.11}$$

Finally, all the losses are combined linearly with a weight added to each loss to balance them in an appropriate trade-off. Thus, the complete loss function L is :

$$L = \lambda_1 L_{pix} + \lambda_2 L_{lm} + \lambda_3 L_o + \lambda_4 L_t + \lambda_5 L_{skew} + \lambda_6 L_g + \lambda_7 L_{reg}, \tag{4.12}$$

where $\lambda_1 \ldots \lambda_7$ are the hyperparameter weights required to balance each loss component.

The loss components can be divided into 3 categories, where each category reflects one task in the learning process. The three tasks are:

1. eye-region reconstruction with L_{pix}, L_{lm} and L_{reg}
2. appearance-based gaze estimation with L_g
3. geometric vergence constraints with L_t, L_o and L_{skew}

We will perform ablation studies to analyse the effectiveness of each task later in this chapter.

4.3 Evaluation

In this section, we introduce the two datasets used to evaluate our work, demonstrate the details of experiments and show both quantitative and qualitative evaluations of our method.

4.3.1 Datasets

Eyediap [132] is a dataset containing videos of 16 subjects looking at various targets. We use the floating ball target videos where 14 distinct subjects are participating. A static head pose session and a dynamic head pose session are recorded for each subject, resulting in 28 sessions of, on average 2701 frames per session. We use the *low-resolution* version (640×480) for our experiments. During training and testing, we utilise all validated frames except for those that are not detected by the face landmark localisation algorithm. We perform cross-subject evaluations on this dataset, using a leave-two-subjects-out strategy by using two subjects' both static and dynamic head pose sessions as the test set, and the remainder as the training set. We effectively train on $\sim$ 61k frames and test on $\sim$ 14k frames.

ETH-XGaze [133] is a large-scale dataset covering a large range of head poses. It collects over one million photos from 110 participants. The evaluation on the test set is performed on an online platform provided by the authors. We use 15 participants as the test set and the remainder as the training set. We also use the landmarks provided by this dataset to train our model.

4.3.2 Implementation details

For our Swin transformer encoder, we use the *tiny* configuration with the pretrained weights on ImageNet [134]. Note that it is potential for other compared methods to use Swin transformer to improve their results. However, our method only uses a low resolution face image on Eyediap dataset and that remains one advantage compared to those who uses high resolution face images or eye images. Re-implement some of the existing work using the state-of-the-art backbone network can be a potentially meaningful thing to do in the future. We use the Adam optimiser [135] with learning rate set to 5×10^{-5} and weight decay set to 1×10^{-4} to train our model for 70 epochs. The hyperparameters $\lambda_1, \ldots, \lambda_7$ to weight all loss function components are set to 1, 0.5, 1×10^3, 2.5×10^3, 5×10^2, 1 and 5×10^{-2} respectively for the Eyediap dataset.

4.3.3 Quantitative evaluation

In this section, we compare our results with some previous methods with the commonly-adopted angular error metric. This error metric measures the angle between the predicted gaze vector and the ground truth gaze vector. We show our Eyediap results in Tab. 4.1. We also include a baseline which uses the Swin transformer to regress gaze rotation only. Due to the difficulty of re-implementation of the appearance-based methods which employ different subjects in different video session types and over different resolutions, we cautiously show some other approaches' reported accuracy for reference. Note that no existing method uses Swin transformer as we do, their choices of backbone networks include AlexNet [123], ViT [91, 136], multimodal CNN [137], customised ResNet [138], bidirectional LSTM [139], multiple VGG-16 networks [140] and multiple dilated CNNs [141].

	Method	mean ± std	median
Appearance-based Methods	Zhang *et al.* [123]#	6.76	\
	Cheng *et al.* [136]	5.17	\
	Zhang *et al.* [137]	7.37	\
	Sinha *et al.* [138]	4.62 ± 2.93	\
	Gaze360 [139]#	5.58	\
	RT-Gene [140]#	6.30	\
	Dilated-Net [141]#	6.57	\
	Baseline	5.25 ± 3.58	4.45
Model-based Methods	PR-ALR [132]*	8.1	\
	Wood *et al.* [65]*	9.44	8.63
	Ploumpis *et al.* [61]	8.85	\
Combined Method	Ours	**4.55 ± 3.29**	**3.82**

* Evaluated on static head pose only. # Converted from face gaze by [125].

Table 4.1. Angle error (°) on gaze vectors originate from eyeballs compared with current literature on the Eyediap dataset.

For the more recent ETH-XGaze dataset, we also show our results on the XGaze dataset and compare with other appearance-based methods.

There are two types of task for gaze vector estimation: i) the gaze originates from eyes and ii) the gaze originates from faces [125]. While our method is successful on the eye gaze task, it does not have advantages on accurately predicting the face gaze. This is due to only one gaze vector being available and our model takes advantage of the correlations between both gaze vectors originating from the eyes. However, our method does not require explicit conversion between the eye gaze task and the face gaze task. Moreover, during training our method approaches the ground truth very quickly and we obtain our results with training for only 20 epochs on 10% of the training set (approx. 60,000 images) that is randomly sampled every batch.

Furthermore, we conduct partially-supervised experiments where only the

Method	mean	std
PureGaze [142]	6.79	\
Zhang *et al.* [143]	4.50	\
Gaze360 [139]	4.46	\
Zhang *et al.* [123]	7.38	\
Cai *et al.* [144]	**3.11**	\
Ours	5.80	4.95

Table 4.2. Angle error (°) compared with current literature on the ETH-XGaze dataset.

ground truth target point in 3D and camera calibrations are provided. This yields a scenario where no ground truth gaze vectors are available and the depth ambiguity inherent in 2D (RGB) images can significantly harm the performance. Our mean angular error on Eyediap is 15.38° and the same error on XGaze is 15.73°. Although the accuracy is significantly compromised compared to the full supervision scenario, the method is still able to learn the gaze directions by locating the 3D gaze targets correctly, while maintaining an accurate 2D eye-region landmark correspondence.

Lastly, we report our reconstructed model's quality. Our face patches on the Eyediap dataset have 96 × 96 pixels, our predicted face landmarks are filtered manually to remove extreme outliers. The average landmark error in pixels is 4.84 pixels per landmark. We further normalise pixel landmark errors by dividing the distance between the left eye's left corner and the right eye's right corner, which results in a proportion of 0.113.

4.3.4 Qualitative evaluation

In this section, we present some qualitative visual results of our method on the Eyediap dataset. Four different predictions are presented in Fig. 4.5. The first row shows a successful example of the static head pose. The second and the third rows demonstrate the successful examples when different head poses are present. The final row shows a failure case where an extreme expression is present. Such expression involves massive morphing of the eye contour region. Although we have chosen the most expression-invariant parts on the face to build the eye-region model, expressions involving eyes like this are still not modelled. Note that the samples on the second and the third rows are trained with a higher weight (5 times larger) on the pixel loss L_{pix}. This results in obtaining a more accurate face area texture. However, due to the nature of the albedo model we used, the eyeball's sclera region appears to be cloudy and in a wrong colour. We consider this to be a trade-off of the albedo model and gaze estimation accuracy is not adversely affected.

4.4 Ablation studies

We have proposed a sophisticated loss function with a linear combination of various different loss components. In this section, we perform ablation studies to determine the effectiveness of each of the three previously described tasks and analyse the advantages of using the state-of-the-art vision backbone network. All ablation study results are presented in Tab. 4.3 and we now detail each method. We used a randomly fixed subject's static and dynamic head pose sessions as the test set for all ablation experiments for a fair comparison. First, we construct a baseline model that comprises of only our vision backbone network (*i.e.* Swin transformer) which predicts two eyeball rotations. It is trained with only the gaze pose loss function L_g, thus it solves Task 2 only. We denote this experiment as *baseline* in the table. Then we construct our system with the vergence model only, *i.e.* the model solves

Task 3 only. Since the predicted gaze origins (*i.e.* eyeball centres) are not available if no 3D eye-region model is reconstructed, we predict the eyeball centres directly using the backbone network. This experiment is denoted as *Vergence model* in the table. Then we report our proposed method with all loss terms (*i.e.* aimed to solve all three tasks simultaneously), denoted as *Ours*. Lastly, we swap or remove a specific part in our proposed method to observe the impact. We remove the loss term L_o to let the model learn without information about ground truth eyeball positions. This experiment is denoted as *w/o* L_o. Finally, we evaluate the improvement in gaze estimation accuracy by employing state-of-the-art vision backbone network Swin transformer against a former popular vision backbone ResNet-18 [111]. We denote the experiment of replacing Swin transformer with an 18 layer ResNet as *Ours - ResNet18*.

Method	mean ± std	median
baseline	5.60 ± 3.28	5.01
Vergence model	4.80 ± 3.07	4.23
w/o L_o	6.64 ± 4.92	5.26
Ours - ResNet18	4.94 ± 3.16	4.34
Ours	$\mathbf{4.11 \pm 2.93}$	**3.42**

Table 4.3. Angle error (°) on subject 15 from Eyediap dataset for the ablation study.

From the results in Tab. 4.3, we can observe that our proposed method that utilises all the loss components and the Swin transformer performs the best among all experiments. A vanilla appearance-based method (*i.e. baseline*) cannot perform competitively when only low resolution face images are fed to the network. Our geometric vergence model that utilises gaze directions from both eyeballs and correlates them contributes hugely towards an accurate gaze estimation. However, combining all three tasks performs better compared to using only Task 2 or Task 3, thus showing the effectiveness of our proposed multi-task method.

Our experiments show that the gaze origin loss L_o is vital to the success of our proposed method. It provides the necessary guidance to both the gaze direction prediction and the 3D eye-region reconstruction, since the 3D eye gaze directions and 3D location of the reconstructed 3D eye-region directly depend on it. Since the ETH-XGaze dataset does not provide ground truth 3D eyeball centres, this partially explains why the results on ETH-XGaze dataset are not as good as the results on Eyediap dataset.

Finally, we justify our choice of Swin transformer as the backbone network over ResNet-18 [111]. ResNet introduced residual connections to the CNN architecture making it one of the most popular vision backbone networks. Only recently, attention-based networks ViT [91] applied to vision problems has led to an improvement over CNN-based performance. There is also literature that focuses on exploiting the attention mechanism provided by transformers to solve the gaze estimation problem [136]. The Swin transformer [126] has proven to be the state-of-the-art transformer-based vision backbone network to date. We also argue that the attention mechanism is essential to the gaze estimation task especially when only low resolution images are provided. Our results further justify our assumptions. Additionally, we observe similar training times when we employ the smallest architectures for both ResNet and the Swin transformer, although the Swin transformer has 28 million parameters while ResNet18 has 11 million parameters.

4.5 Limitations and potential societal impact

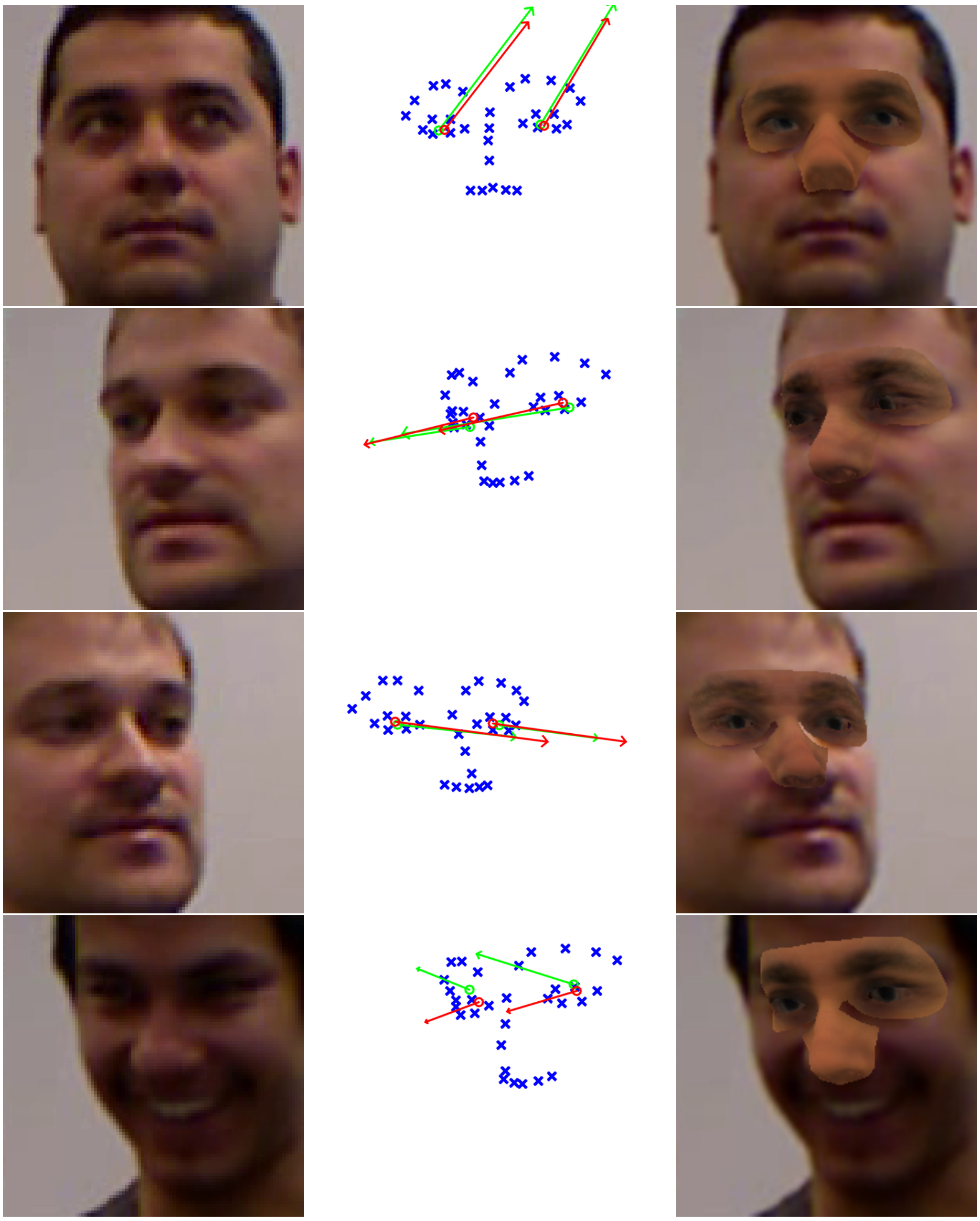
Due to the datasets employed, our evaluations are conducted under controlled environments (with camera calibration information, subjects' head position is relatively static). Thus, the performance on *in-the-wild* images is uncertain. Such images imply three difficulties: lower resolution over the face region, high variance in subject identity, and lack of camera calibration. The results have shown that our method inherits the appearance methods' feature extraction ability on low resolution frames from the Eyediap dataset. Future work on providing a larger training set and transforming images to a normalised camera space can mitigate the remaining two difficulties. Nonetheless, this method remains highly applicable when a controlled environment and camera calibration is available.

Differentiable rendering uses only the ambient light model and the Phong shader. Applying a more sophisticated light model and shader to higher resolution eye images to capture more refined details and eye surface reflections in the training process remains unexplored. Also, building a more sophisticated eyeball model is desirable, it can enable iris modelling, eyeball size modelling and pupil modelling. A high resolution eyeball model that can be integrated into the training process and balance different resolutions between eyeballs and faces is still an unsolved problem.

This system can potentially be used on devices with a front camera to model users' faces and gaze targets. Then infer the screen content that the user is viewing. Thus, this can potentially invoke privacy issues.

4.6 Conclusion

We presented a novel approach that reconstructs an eye-region model as well as the gaze direction by utilising the advantages of both appearance-based methods and model-based methods. Our results show that we achieve state-of-the-art performance on the gaze estimation task while reconstructing an eye-region model. Our method attempts to close the gap on gaze estimation task where model-based methods lack the raw feature extraction ability by utilising the state-of-the-art vision backbone network. Our work can be further applied to inter-ocular distance prediction, ear-to-ear face region modelling, and human head modelling with highly accurate gaze estimation. Our work contributes to human eye-region understanding, human-computer interaction, wearable devices and virtual reality.



(a) Raw (b) Prediction (c) Rendered

Figure 4.5. Raw input images, predicted landmarks (blue crosses), predicted gaze rays (red rays), ground truth gaze rays (green rays) and rendered eye-region model. First three rows are successful predictions for various head poses and various identities. The last row shows a failure case where extreme expression is present. Such expression is not included in the eye-region model space.

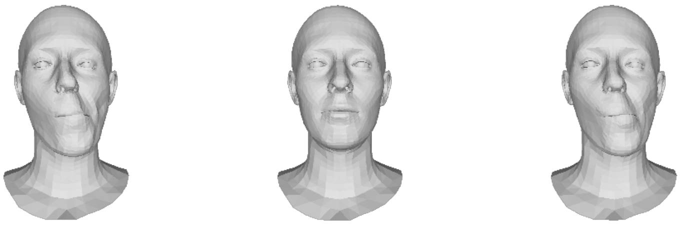
5 Full Head Reconstruction Autoencoder with Expression Disentanglement

5.1 Introduction

A 3D Morphable Model (3DMM) for human faces was proposed by Blanz and Vetter [12] more than 20 years ago. Since then, it has gained widespread use in a wide variety of both 2D and 3D applications. In more recent years, more non-linear 3D face models have been built that exploit powerful deep learning techniques. This has allowed more detailed reconstructions from more compressed latent representations [96]. Initially, models were built from neutral-expression faces only, but with more comprehensive datasets, newer approaches have also modelled facial expressions, for more general applications [93, 3].

A key ability is to disentangle the identity part and the expression part from any human face input data (see Fig. 5.1), and direct those disentangled parts into the corresponding model components. Such approaches can be beneficial for many applications, such as face reenactment and face recognition. Jiang *et al.* initiated this topic [95], followed by Zhang *et al.* with the previous state-of-the-art [54].

Here, we propose a concise architecture that improves disentanglement performance with fewer restrictions (*i.e.* topology information), compared



Raw Face Predicted Neutral Face Predicted Full Face

Figure 5.1. Disentangling identity from the full expressive face.

to the state-of-the-art [54], and we evaluate the results, such as is given in Fig. 5.1. To achieve this, we design two variational autoencoders (VAEs) for identity and expression separately, but are able to train them in an end-to-end manner without any pre-training. We employ the attention-based point cloud transformer (PCT) [17] as the encoder. This processes a set of points, which is unordered and without local neighborhood connectivity information. In other words, mesh topology is obviated, and we enable training on point cloud data for disentangled facial expression modelling. We use a point cloud transformer to process point cloud data instead of viewing the input as mesh data. In other words, we discard the mesh's topological information, which graph neural networks require. However, we still use point clouds with the same size and order and visualise experiment results with the predefined topologies. That is, for the input, we assume point-to-point correspondences are available. Also, for the labels, we assume a corresponded neutral face to every face with expression is available. We also follow the idea of the information bottleneck in information theory, using an additional mutual information regulariser to encourage disentanglement and allow tuning of the compression of the latent representation. Furthermore, we utilise expression label information provided by the datasets by employing a *conditional* VAE as an upgrade to the proposed method. This enforces more disentangled expression information and thereby contributes to the explainability of the generative model.

In summary, our main contributions are:

1. Incorporation of the point cloud transformer network, removing the requirement of a known mesh vertex topology, and leveraging the high performance of attention-based architectures.

2. Use of an information bottleneck on the identity reconstruction subsystem to encourage improved identity and expression disentanglement on 3D facial input data.

3. Application of a conditional VAE on top of the proposed method to further disentangle expression information and build a generator that generates from semantically meaningful expression latent variables.

5.2 Proposed Method

5.2.1 Architecture

Denoting a 3D face point cloud $\mathbf{X}_i \in \mathbb{R}^{M \times 3}$ where $i \in [1..N]$ and M is the number of points in each 3D face. We assume the dataset is comprised of $\{\mathbf{X}_1, \mathbf{X}_1^{id}, \ldots, \mathbf{X}_N, \mathbf{X}_N^{id}\}$, that means for each 3D face in the dataset, there is a corresponding identity face (*i.e.* neutral face). The goal is to reconstruct an identity face and full (expressive) face independently using their respective latent representations. We illustrate our architecture in Fig. 5.2.

We separate the whole 3D facial expression modelling system into two sub-systems: identity (ID) VAE and full-face VAE, sharing the same encoder and trained simultaneously in an end-to-end manner. We follow the common VAE structure to build each sub-system in the first place. The point cloud

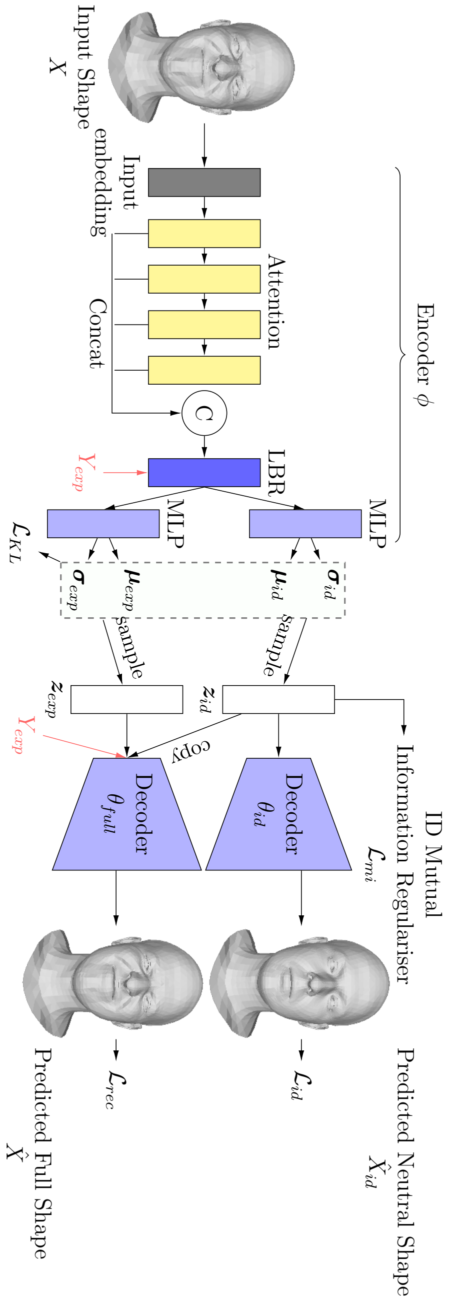


Figure 5.2. Overview of the architecture. *LBR* combines *Linear*, *BatchNorm* and *ReLU* layers. *MLP* stands for multi-layer perceptron. Additional conditional VAE add-ons are marked in red.

transformer (PCT [17]) is used as our encoder network $q\left(\mathbf{z}_{id}, \mathbf{z}_{exp} \mid \mathbf{X}, \phi\right)$ with learned weights ϕ that extracts features from 3D face point clouds, then predicts two sets of latent code: the identity latent code $\mathbf{z}_{id}$ and the expression latent code $\mathbf{z}_{exp}$. We utilise two separate decoders $p\left(\hat{\mathbf{X}}_{id} \mid \mathbf{z}_{id}, \theta_{id}\right)$ and $p\left(\hat{X} \mid \mathbf{z}_{id}, \mathbf{z}_{exp}, \theta_{full}\right)$ with weights θ_{id} and θ_{full} to reconstruct the identity face and the full face respectively. We cut off the gradient back-propagation flow for $\mathbf{z}_{id}$ from decoder θ_{full} to avoid updating the identity latent code with respect to errors that contain expression information. This turns the full face VAE into a conditional VAE [145] that learns to extract an expression latent representation only. In early experimentation, we employed a unified decoder, effectively using a zero-padded full decoder $p\left(\hat{\mathbf{X}} \mid \mathbf{z}_{id}, \mathbf{z}_{exp} = 0, \theta_{full}\right)$ as our identity decoder. However, we found that the dual-decoder design achieves better reconstruction and disentanglement results.

Following the common VAE design for loss functions, we utilise reconstruction losses ($\mathcal{L}_{id}$ and $\mathcal{L}_{rec}$) and variational loss $\mathcal{L}_{KL}$ which will be explained in Section 5.2.2. In addition to the usual VAE structure, we utilise only an additional mutual information regularisation function on the identity latent code $\mathcal{L}_{mi}$, achieving significant improvement on disentanglement results compared to the current state of the art. We will explain the choice of this regulariser in Section 5.2.3, elaborate the two loss functions $\mathcal{L}_{id}$ and $\mathcal{L}_{mi}$ jointly as an information bottleneck. In Section 5.2.4, the four loss components are summed to give the loss function that enables end-to-end training of our network.

5.2.2 3D Face VAE

Variational Autoencoder (VAE)

The VAE, firstly proposed by [11], has a goal of maximising the real data likelihood $P\left(\mathbf{X}\right)$ by introducing a latent code vector $\mathbf{z}$ via:

$$P\left(\mathbf{X}\right) = \int P\left(\mathbf{X}|\mathbf{z}\right) P\left(\mathbf{z}\right) d\mathbf{z}. \tag{5.1}$$

Modern approaches use a deep neural network $p_\theta\left(\mathbf{X} \mid \mathbf{z}\right)$ to approximate the distribution $P\left(\mathbf{X} \mid \mathbf{z}\right)$ and name it decoder network (or generator). Since the decoder distribution's posterior $p_\theta\left(\mathbf{z} \mid \mathbf{X}\right)$ is intractable, VAE introduces an encoder network $q_\phi\left(\mathbf{z} \mid \mathbf{X}\right)$ to approximate it. To infer $P\left(\mathbf{X}\right)$, we start from the KL divergence between the two conditional distributions:

$$\begin{aligned} &KL\left(q_\phi\left(\mathbf{z} \mid \mathbf{X}\right) \parallel P\left(\mathbf{z} \mid \mathbf{X}\right)\right) \\ &\qquad =E_{\mathbf{z}\sim q_\phi}\left[\log \frac{q_\phi\left(\mathbf{z} \mid \mathbf{X}\right)}{P\left(\mathbf{z} \mid \mathbf{X}\right)}\right] \\ &\qquad =-E_{q_\phi}\left[\log P\left(\mathbf{X} \mid \mathbf{z}\right)\right] \\ &\qquad\quad + KL\left(q_\phi \parallel P\left(\mathbf{z}\right)\right) + \log P\left(\mathbf{X}\right), \end{aligned} \tag{5.2}$$

and can be written as:

$$\log P\left(\mathbf{X}\right) = E_{q_\phi}\left[\log P\left(\mathbf{X} \mid \mathbf{z}\right)\right] \tag{5.3}$$

$$- KL\left(q_\phi\left(\mathbf{z} \mid \mathbf{X}\right) \parallel P\left(\mathbf{z}\right)\right) \tag{5.4}$$

$$- KL\left(q_\phi\left(\mathbf{z} \mid \mathbf{X}\right) \parallel P\left(\mathbf{z} \mid \mathbf{X}\right)\right). \tag{5.5}$$

The objective is then to find the ϕ and θ that maximise $\log P\left(\mathbf{X}\right)$. Since the decoder distribution's posterior $P\left(\mathbf{z} \mid \mathbf{X}\right)$ in Eq. (5.5) is intractable and any KL divergence is greater than or equal to 0, we instead optimise the lower bound by ignoring the KL term (*i.e.* Eq. (5.5)). Result in the final Evidence lower bound (ELBO) loss function:

$$\begin{aligned} ELBO = & - E_{q_\phi}\left[\log P\left(\mathbf{X} \mid \mathbf{z}\right)\right] && (5.6) \\ & + KL\left(q_\phi\left(\mathbf{z} \mid \mathbf{X}\right) \parallel P\left(\mathbf{z}\right)\right), && (5.7) \end{aligned}$$

where q_ϕ is the encoder, and p_θ is the decoder.

Point Cloud Transformer (PCT)

We adopt the point cloud transformer (PCT [17]) as our encoder to extract a latent code from input data. The PCT is an attention-based [90] network that processes unordered point sets and employs farthest point sampling and nearest neighbor search for input embedding. The core component, the attention module, takes the embedded point cloud inputs, and generates refined attention features based on global context by connecting all pairs of point clusters with attention weights. The attention feature is then fed into MLPs to generate identity and expression latent codes. Our PCT-based encoder is depicted in the encoder part of Fig. 5.2.

Our Proposed Method

To practically construct VAEs for both the identity sub-system and the full face sub-system, we have to build variational inference for the latent codes. Thus we let the encoder output the mean μ and the standard deviation σ of an isotropic Gaussian distribution $\mathcal{N}\left(\mu, \operatorname{diag}\left(\sigma\right)\right)$ that represents the latent code's distribution. Then the latent code is sampled from the predicted latent code distribution. Here we follow the original VAE paper [11] and use the reparameterisation trick [11] for differentiable sampling. Therefore, the KL loss (Eq. (5.7)) from the ELBO loss for both identity and expression latent code can be formed as:

$$\mathcal{L}_{KL}^{id} = KL\left(q_{\phi}\left(\mathbf{z}_{id} \mid \mathbf{X}\right) \parallel \mathcal{N}\left(\mathbf{0}, \mathbf{I}\right)\right) \tag{5.8}$$

$$\mathcal{L}_{KL}^{exp} = KL\left(q_{\phi}\left(\mathbf{z}_{exp} \mid \mathbf{X}\right) \parallel \mathcal{N}\left(\mathbf{0}, \mathbf{I}\right)\right) \tag{5.9}$$

$$\mathcal{L}_{KL} = \frac{1}{\|\mathbf{z}_{id}\| + \|\mathbf{z}_{exp}\|}\left(\mathcal{L}_{KL}^{id} + \mathcal{L}_{KL}^{exp}\right), \tag{5.10}$$

Meanwhile, it is a common practice to replace the reconstruction term Eq. (5.6) in the ELBO loss with a loss function that is used in non-variational deep learning tasks, such as the L1 norm used in [23, 95, 54] or Mean Squared Error (MSE). We adopt the MSE, and the two reconstruction losses for identity and full faces are:

$$\mathcal{L}_{id} = \|\hat{\mathbf{X}}^{id} - \mathbf{X}^{id}\|_2^2 \quad \text{and} \quad \mathcal{L}_{rec} = \|\hat{\mathbf{X}} - \mathbf{X}\|_2^2. \tag{5.11}$$

By doing so, we ensure the same reconstruction goal with the loss function that is practically proven to work well with stochastic gradient descent. This leaves the latent code to be the only variational part that is represented in distributions. The KL loss can then be seen as a regulariser that pulls them towards unit isotropic Gaussian distributions.

Additional Experiment: Conditional VAE

Several datasets provide corresponding labels along with their data. Utilising such information in a generative model can be beneficial for its performance and explainability. Sohn *et al.* [145] propose the Conditional Variational Autoencoder (CVAE), which utilises label information to allow the modelling of raw data conditioned on it. The main modification to the original VAE is to make both the encoder distribution q_ϕ and the decoder distribution p_θ condition on corresponding labels Y. The ELBO loss function in our setting for the full face VAE is then modified to:

$$\begin{aligned} ELBO_{cvae} = E_{\mathbf{z}} \left[\log P\left(\mathbf{X} \mid \mathbf{z}_{id}, \mathbf{z}_{exp}, Y_{exp}\right)\right] \\ - KL\left(q_\phi\left(\mathbf{z}_{exp} \mid \mathbf{X}, Y_{exp}\right) \parallel P\left(\mathbf{z}_{exp} \mid Y_{exp}\right)\right), \quad (5.12) \end{aligned}$$

where Y_{exp} is a one-hot encoded expression label multiplied by the expression's level, which has the range $[0, 1]$.

As shown in Fig. 5.2 (red parts), we build a CVAE architecture upon our VAE architecture, by concatenating one-hot labels after the penultimate fully connected layer in our encoder. The one-hot encoded labels are also concatenated after the expression latent code, which is then passed to the full face decoder. With the CVAE, the trained decoder can generate new samples from given expression labels.

5.2.3 Mutual Information Regulariser

Due to the cost of 3D scans, most of the current 3D face datasets are obtained under specific experimental conditions rather than from in-the-wild. To increase the number of 3D faces collected, one has to acquire multiple scans of the same person. In this case, there exists groups of 3D face indices $K \subset [1..N]$, such that their corresponding 3D faces share the same corresponded neutral face, *i.e.* $\mathbf{X}_k = \mathbf{X}_{k'}, \forall k, k' \in K$. Therefore, the identity VAE part differs from the traditional VAE in two respects. Firstly, the ID decoder θ_{id} does not reconstruct the original input; rather, it reconstructs an expression-neutralised input, which has information content that is always less than or equal to that of the input. Secondly, assuming a single latent code, with the identity and the expression parts entangled, the ID decoder would have to reconstruct the same 3D face identity from different latent codes. That is, the identity latent code $\mathbf{z}_{id}$ would contain information about expressions on the input face. Thus, we propose an information bottleneck on the identity latent code, and address both challenges at the same time. We then modify it to work with a deep VAE model as a combination of a mutual information regulariser $\mathcal{L}_{mi}$ and the identity reconstruction loss $\mathcal{L}_{id}$. It simultaneously encourages the decoder to better reconstruct the identity face and forces the identity latent code to contain only the information of the reconstructed neutral faces and, therefore, achieves better identity and expression disentanglement.

Information Bottleneck

An information bottleneck is an information theory idea proposed by Tishby *et al.* [53]. The main idea is to build an objective function that jointly maximises the mutual information between the latent code and its reconstruction, and that minimises the mutual information between the input and the latent code. Putting more weight on the second of these terms allows for a more compressed latent representation [52].

In the identity sub-system scenario, the encoder encodes faces with expressions, which naturally introduces redundant expression information into the identity latent code, resulting in low compression efficiency. So we propose to put more weight on compression, jointly with reconstructing neutral faces, to eliminate expression information contained in the identity latent code. Also, the information bottleneck does not assume the reconstruction has to be identical to the input. Thus, to fit the information bottleneck to the identity sub-system, the objective can be formulated as:

$$J_{IB} = -I\left(\mathbf{X}_{id}; \mathbf{z}_{id}\right) + I\left(\mathbf{z}_{id}; \mathbf{X}\right), \tag{5.13}$$

Note that the Lagrangian multiplier in the second term from the original information bottleneck objective is ignored at this stage, because the weight of the second term is built into the deep learning framework via hyperparameters, described later in this section.

To begin with building this objective into the current 3D face VAE system, we reformulate the objective J_{IB}'s first term as:

$$\begin{aligned}
&-I\left(\mathbf{X}_{id}; \mathbf{z}_{id}\right) && (5.14)\\
&= -\int_{\mathbf{X}_{id}} \int_{\mathbf{z}_{id}} p\left(\mathbf{X}_{id}, \mathbf{z}_{id}\right) \log \frac{p\left(\mathbf{X}_{id} \mid \mathbf{z}_{id}\right)}{p\left(\mathbf{X}_{id}\right)} d\mathbf{z}_{id} d\mathbf{X}_{id} && (5.15)\\
&= -\int_{\mathbf{X}_{id}} \int_{\mathbf{z}_{id}} p\left(\mathbf{X}_{id}, \mathbf{z}_{id}\right) \log p\left(\mathbf{X}_{id} \mid \mathbf{z}_{id}\right) d\mathbf{z}_{id} d\mathbf{X}_{id} && (5.16)\\
&\quad - H\left(\mathbf{X}_{id}\right). && (5.17)
\end{aligned}$$

The entropy term $-H\left(\mathbf{X}_{id}\right)$ of the neutral faces in dataset is a constant during training so can be ignored. Following the assumption described in the architecture, we have:

$$\begin{aligned}
p\left(\mathbf{X}_{id}, \mathbf{z}_{id}\right) &= \int_{\mathbf{X}} p\left(\mathbf{X}, \mathbf{X}_{id}, \mathbf{z}_{id}\right) d\mathbf{X} && (5.18)\\
&= \int_{\mathbf{X}} p\left(\mathbf{z}_{id} \mid \mathbf{X}\right) p\left(\mathbf{X}, \mathbf{X}_{id}\right) d\mathbf{X}. && (5.19)
\end{aligned}$$

Since $\mathbf{X}$ and $\mathbf{X}_{id}$ form a data point in the dataset of size N, we can estimate $p\left(\mathbf{X}, \mathbf{X}_{id}\right)$ using the dataset (*i.e.* empirical data distribution), then further derive an empirical lower bound of the mutual information in Eq. (5.14):

$$
\begin{aligned}
&-I\left(\mathbf{X}_{id};\mathbf{z}_{id}\right) && (5.20)\\
\leq&\frac{1}{N}\sum_{n}^{N}-\int_{\mathbf{z}_{id}}p\left(\mathbf{z}_{id}\mid\mathbf{X}_{n}\right)\log p\left(\mathbf{X}_{n}^{id}\mid\mathbf{z}_{id}\right)d\mathbf{z}_{id} && (5.21)\\
\leq&\frac{1}{N}\sum_{n}^{N}-E_{\mathbf{z}_{id}\sim q_{\phi}}\left[\log p_{\theta_{id}}\left(\mathbf{X}_{n}^{id}\mid\mathbf{z}_{id}\right)\right], && (5.22)
\end{aligned}
$$

where the encoder network q_ϕ is used to estimate the conditional probability $p\left(\mathbf{z}_{id}\mid\mathbf{X}_n\right)$ and the identity decoder network $p_{\theta_{id}}$ is used to estimate the conditional probability $p\left(\mathbf{X}_n^{id}\mid\mathbf{z}_{id}\right)$, thus we have the final result in Eq. (5.22). By comparing with the reconstruction loss in the original ELBO loss in Eq. (5.6), we note that Eq. (5.22) is an aggregated negative likelihood of the reconstructed identity faces. In order to take advantage of minibatch training and stochastic gradient descent, we use mean squared error loss $\mathcal{L}_{id}$ in Eq. (5.11) to replace the original variational reconstruction loss. Using stochastic gradient descent to backpropagate from the loss function $\mathcal{L}_{id}$ on a minibatch basis can be seen as a practically effective way of estimating the gradient of the aggregated negative likelihood in Eq. (5.22) over the whole dataset. Thus we encourage better reconstruction of the identity face, by effectively maximising the mutual information between $\mathbf{X}_{id}$ and $\mathbf{z}_{id}$.

Using the encoder network to estimate the conditional probability $p\left(\mathbf{z}_{id}\mid\mathbf{X}\right)$ in second mutual information term in Eq. (5.13), results in the mutual information loss $\mathcal{L}_{mi}$, given as:

$$
\mathcal{L}_{mi}=I_q\left(\mathbf{z}_{id};\mathbf{X}\right)=E_{\mathbf{X}}\left[KL\left(q_\phi\left(\mathbf{z}_{id}\mid\mathbf{X}\right)\parallel q_\phi\left(\mathbf{z}_{id}\right)\right)\right]. \tag{5.23}
$$

However, obtaining the aggregated posterior $q_\phi\left(\mathbf{z}_{id}\right)=E_{\mathbf{X}}\left[q_\phi\left(\mathbf{z}_{id}\mid\mathbf{X}\right)\right]$ directly can be undesirable, since it requires a forward pass of the entire dataset on the encoder network for each backpropagation [9]. Therefore we applied the minibatch weighted sampling (MWS) technique proposed by [46], which was inspired by importance sampling, to estimate $q_\phi(\mathbf{z}_{id})$. Suppose we have a minibatch of $\{\mathbf{X}_1, \ldots, \mathbf{X}_B\}$, the estimator is formed as:

$$E_{q_\phi(\mathbf{z}_{id})}\left[q_\phi(\mathbf{z}_{id})\right] \approx \frac{1}{B}\sum_i^B \left[\log \frac{1}{NB}\sum_j^B q_\phi\left(\mathbf{z}_{id}\left(\mathbf{X}_i\right) \mid \mathbf{X}_j\right)\right], \tag{5.24}$$

where $\mathbf{z}_{id}(\mathbf{X}_i)$ is a sample from $q_\phi(\mathbf{z}_{id} \mid \mathbf{X}_i)$.

We have formed an information bottleneck on identity faces, resulting in the combination of two loss functions $\mathcal{L}_{id}$ and $\mathcal{L}_{mi}$. However, the original information bottleneck introduces a Lagrangian multiplier to allow tuning of the compression level. Since we replaced the variational reconstruction loss with a MSE loss, both loss functions have to be re-weighted to correctly balance the training process. Thus we introduce two hyperparameters β_{id} and β_{mi} for this purpose. To strengthen the information bottleneck, one can increase β_{id} for better reconstructed identity faces and increase β_{mi} for a more compressed identity latent code.

5.2.4 Final Loss Function

To balance the reconstruction loss and KL loss, two additional hyperparameters are introduced, resulting in the full loss function:

$$\mathcal{L} = \lambda_1\left(\mathcal{L}_{rec} + \beta_{id}\mathcal{L}_{id}\right) + \lambda_2\left(\mathcal{L}_{KL} + \beta_{mi}\mathcal{L}_{mi}\right). \tag{5.25}$$

Where we divide four loss components into two groups, balancing them with λ_1 and λ_2, then increasing the information bottleneck weights β_1 and

β_2 to strengthen its effect.

As there are 4 loss functions in the whole system, choosing the right four hyperparameters to balance them can be cumbersome. Therefore the tuning of the weighting parameters is segmented into two phases. The first phase is to set both β_{id} and β_{mi} to 1, then tune λ_1 and λ_2 to balance the MSE loss and KL loss because MSEs are used as reconstruction loss instead of the variational ones. The second phase is then to increase β_{id} and β_{mi} to strengthen the information bottleneck. Note that two other 3D facial expression disentangling works [95, 54] propose to use the same weight on all identity, expression and full face reconstruction losses, we argue that increase the weight of the identity reconstruction loss while keeping the rest reconstruction losses' weights unchanged can attribute to better disentanglement results.

Decomposition of the KL term

Kim *et al.* [9] propose a decomposition of the aggregated KL term (*i.e.* Eq. (5.4)) in the objective function, shown as follow:

$$E_{\mathbf{X}}\left[KL\left(q_\phi\left(\mathbf{z} \mid \mathbf{X}\right) \parallel p\left(\mathbf{z}\right)\right)\right] = I\left(\mathbf{X};\mathbf{z}\right) + KL\left(q_\phi\left(\mathbf{z}\right) \parallel p\left(\mathbf{z}\right)\right), \tag{5.26}$$

where $q_\phi\left(\mathbf{z}\right) = E_{p_{data(x)}}\left[q_\phi\left(\mathbf{z} \mid x\right)\right]$ is the marginal posterior of the encoding distribution, p_{data} is the empirical data distribution representing the whole dataset and $I\left(x;\mathbf{z}\right)$ is the mutual information between $\mathbf{X}$ and $\mathbf{z}$. The mutual information term represents how much information about the input face is stored in the latent code $\mathbf{z}$. The KL term represents how close is the distance between the encoded latent code distribution and the prior distribution (*i.e.* multi-variate unit Gaussian in the context).

A number of publications consider undesired to penalise the mutual information term in the decomposed KL since it can decrease reconstruction quality [9, 46]. However, penalising it can be beneficial in the facial expression disentangling setting since multiple faces are mapped to single neutral face, forming a many-to-one VAE. Inspired by this, while the publications penalise more on the second term for better disentanglement, we propose to add an additional mutual information regulariser on identity latent code to penalise the information about faces with expression stored in the identity latent code, therefore achieve less oscillated reconstruction identity faces for various faces share the same identity.

5.2.5 Implementation Details

PyTorch [146] is used to build the whole system. The encoder PCT uses the original PCT paper's architecture on the self-attention module, followed by fully connected layers that are configured as $\{1024/256/64/\left(\|\mathbf{z}_{id}\| + \|\mathbf{z}_{exp}\|\right) \times 2\}$. For decoders, the identity decoder and the full face decoder share the same architecture: an MLP with 256 hidden neurons. For a fair comparison with other models that evaluate on the CoMA dataset, we choose latent code sizes as $|\mathbf{z}_{id}| = 4$ and $|\mathbf{z}_{exp}| = 4$. We select loss function weights based on a cross-validation set from the training and empirical tuning the values, then train on the full training set. For the loss function weights, we use $\lambda_1 = 6.6 \times 10^{-2}$, $\lambda_2 = 3 \times 10^{-3}$, $\beta_{id} = 10$, $\beta_{mi} = 50$. The whole system is trained over 300 epochs with the Adam [135] optimiser and we set the learning rate to 5×10^{-5} with a L2 weight decay [147] set to 10^{-4} and a learning rate decay of 0.7 for every 50 epochs. The KL loss and mutual information regulariser weight λ_2 decays linearly to 0 over 350 epochs.

5.3 Evaluation

We now evaluate the performance of our proposed system. First, the two datasets employed for evaluation purposes are introduced. Second we present our evaluation metrics. Third, we compare our proposed network with three state-of-the-art systems. Finally, ablation studies are presented. After presenting the two datasets and the evaluation metrics, we compare our proposed VAE with three state-of-the-art systems in quantitative evaluations. We then present an ablation study and, finally, qualitative results for both proposed VAE (Fig. 5.4 and Fig. 5.5) and conditional VAE (Fig. 5.6 and Fig. 5.7) are presented. Note that all experiments are based on point clouds only, with the mesh topology only used for visualisation.

5.3.1 Datasets

CoMA Dataset [21] This contains scans of 12 individuals performing 12 different expressions. For each subject-expression pairing, there is a video of that person making the desired expression, giving a total of $20,466$ 3D scans in the dataset. All of the 3D face scans are registered with FLAME topology [3] and are pose normalised. Each 3D scan has 5023 vertices and 9976 triangle faces. We follow the data split scheme proposed by [95] and [21] that sorts all videos in alphabetical order, and then takes 10 frames for every 100 frames as the test set and train on the reminder.

BU-3DFE [148] This contains 100 individuals each with 6 different expressions over 4 different expression levels. For each subject, one neutral scan is performed, resulting in a total of $2,500$ scans. All the 3D faces are registered to the same topology. Each 3D scan has 5996 vertices and 11753 triangle faces. In order to further normalise the pose, we perform a rigid registration of all 100 neutral faces to their mean based on a number of landmarks. Then for each subject, their 24 expression scans are rigidly registered with the neutral face based on a number of expression invariant key points. Finally, following [54], the first 10 subjects are selected as the test set and the rest are used for training.

5.3.2 Evaluation Metrics

We employ the same evaluation metrics as the most closely related papers [95, 54], namely reconstruction error and disentanglement error (this is exactly the same error as the decomposition error used in [54]).

Reconstruction Error The fundamental metric for a generative model is reconstruction error. Since all the vertices are corresponded to the ground truth, we can use the Average Vertex (Euclidean) Distance (AVD) to measure the reconstruction quality:

$$E_{rec} = \frac{1}{M} \sum_{j}^{M} \left\| \hat{\mathbf{X}}_j - \mathbf{X}_j \right\|_2 , \tag{5.27}$$

where M is the number of vertices in a single face.

Disentanglement Error The disentanglement error measures the variance in the predicted identity faces from the same subject. Given a subset of the test set that contains various expressions from the subject d denoted as: $\{\mathcal{M}_i\}$, the predicted identity faces (*i.e.* neutral faces) can be generated by the system, denoted as: $\left\{\mathcal{M}_i^{id}\right\}$. Let $\mathcal{M}^d$ denote the mean face of all predicted neutral faces for subject d. The disentanglement error can then be formulated by:

$$E_{dis} = \mathrm{STD}\left(\left\{\left\|\mathcal{M}_{ij}^{id} - \mathcal{M}_{j}^{d}\right\|_2\right\}\right), \tag{5.28}$$

where j is the vertex index and STD estimates the standard deviation. Jiang *et al.* [95] propose to apply the same error metric on predicted expressions from different subjects that perform the same expression. We omit this analysis, because we assume different subjects perform the same expression in different ways, thus the expression disentanglement error is expected to be high. However, we use a *conditional* VAE to further disentangle expression information.

Jiang *et al.* [95] propose to perform the same error metric on predicted expressions from different subjects that perform the same expression. Both Zhang *et al.* [54] and this paper omit this error because we use a different assumption on expressions. Jiang *et al.* use an average expression as the ground truth expression for their expression model, while Zhang *et al.* argue that different subjects have different ways to express the same expression. In fact, we find that averaging all the angry expressions in the original BU-3DFE dataset results in a face that is very close to a neutral face. Therefore, Zhang *et al.* construct a unique ground truth expression for each subject. So unlike identity variance, the expression variance is expected to be greater than zero. Unlike Zhang *et al.*, we choose to model expression implicitly, that implies no ground truth expression. Additionally, the identity variance can measure how accurate the identity is disentangled from the expression face solely, therefore, along with reasonable reconstruction error, it becomes a sufficient metric for disentanglement.

5.3.3 VAE Quantitative Evaluation

Compared Methods We compare our work to a number of 3D face modelling methods based on the autoencoder structure. MeshAE [21] and SpiralNet++ [92] both focus on applying a GCN architecture to 3D-to-3D mesh reconstruction, regardless of disentanglement. FLAME [3] builds identity and expression latent representations separately and reconstructs using a linear system. The two most related works to our proposed system are Jiang *et al.* [95] and Zhang *et al.* [54]. Both of these focus on disentangled facial expression modelling using GCN architectures and are evaluated using same metrics. Note that We use the same scale for our heatmap, for visual comparison with existing methods, please refer to the original papers.

Tab. 5.1 gives disentanglement results for several systems. Our baseline, denoted as "Ours - No IB" which stands for no *Information Bottleneck*, sets β_{id} to 1 and β_{mi} to 0 and obtains a competitive result. Setting β_{id} to 1 means to disable the effect of the information bottleneck. One intermediate result "Ours - $\beta_{mi} = 0$" sets $\beta_{id} = 10$ and $\beta_{mi} = 0$ shows the effectiveness of the $\mathcal{L}_{mi}$ solely. Our final proposed method sets $\beta_{id} = 10$ and $\beta_{mi} = 50$ and surpasses the current state-of-the-art by a large margin.

Method	mean	median
FLAME [3]	0.599	0.591
Jiang *et al.* [95]	0.064	0.062
Zhang *et al.* [54]	0.019	0.020
Ours - No IB	0.025	0.022
Ours - $\beta_{mi} = 0$	0.016	0.013
Ours	**0.006**	**0.005**

Table 5.1. Disentanglement result (mm) compared with current literature on CoMA dataset.

The detailed results of reconstruction error are shown in Tab. 5.2. The reconstruction results are divided into two groups, where non-disentangling methods generally have better reconstruction results. One potential reason is that disentanglement methods add extra objectives for disentanglement that act similar to regularisers, which will make reconstruction performance drop. However, from the results, our model can still achieve competitive reconstruction results. Furthermore, by comparing the two results of our model, the performance drop on reconstruction quality introduced by the mutual information regulariser is a tolerable price to pay for disentanglement.

	Method	mean ± std	median	ne mean ± std	median
Non-disentanglement Methods	MeshAE[21]	0.845 ± 0.994	0.496	\	\
	SpiralNet++[92]	0.543 ± 0.663	0.320	\	\
	Ours - No IB	0.614 ± 0.192	0.594	0.065 ± 0.021	0.065
	Ours - $\beta_{mi} = 0$	0.604 ± 0.183	0.581	0.054 ± 0.020	0.049
Disentanglement Methods	FLAME[3]	1.451 ± 1.649	0.871	\	\
	Jiang *et al.* [95]	1.413 ± 1.639	1.017	\	\
	Zhang *et al.* [54]	0.665 ± 0.748	**0.434**	\	\
	Ours	**0.663 ± 0.215**	0.643	**0.051 ± 0.021**	0.048

Table 5.2. Reconstruction results: Average Vertex Distance (mm) compared with literature on the CoMA dataset (column 3 and 4). Our methods' reconstructed neutral faces AVD (column 5 and 6).

The results for disentanglement error and reconstruction error on the BU-3DFE dataset are shown in Tab. 5.3. Our approach here employs $\beta_{id} = 1$ and $\beta_{mi} = 50$ to obtain a better disentanglement result compared to current state-of-the-art, again with a competitive reconstruction error.

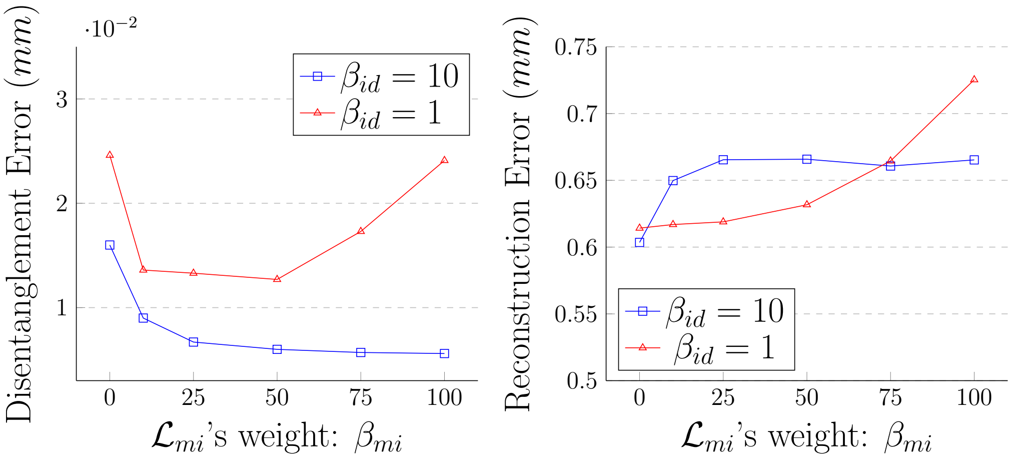
5.3.4 Ablation Studies

The effect of introducing the information bottleneck is now evaluated. In Fig. 5.3a and Fig. 5.3b, we study the impact of modifying the mutual information regulariser's weight $\beta_{mi} \in \{0, 10, 25, 50, 75, 100\}$ while keep the identity reconstruction loss weight at fixed values $\beta_{id} \in \{0, 10\}$. From the

Method	Disentanglement Error		Reconstruction Error	
	mean	median	mean ± std	median
FLAME[3]	0.600	0.632	2.596 ± 2.055	2.055
Jiang *et al.* [95]	0.611	0.590	2.054 ± 1.199	1.814
Zhang *et al.* [54]	0.361	0.327	$\mathbf{1.551 \pm 0.924}$	**1.375**
Ours	**0.328**	**0.296**	1.628 ± 0.333	1.589

Table 5.3. Disentanglement results (*mm*) compared with current literature on the BU-3DFE dataset.

graphs, one can observe that increasing identity loss weight and mutual information regulariser weight can result in lower disentanglement error and higher reconstruction error. The increase in reconstruction error is expected because there exists a fundamental trade-off for the information bottleneck (IB) between concise representation and good reconstruction power [149]. The mutual information regulariser encourages the first term, while the reconstruction errors encourage the other. Meanwhile, using overly large IB weights can harm performance. From the graphs, one can observe that given an identity reconstruction loss is not strongly weighted, the information bottleneck can constrain the necessary information to convey from input to latent code, resulting in an overly compressed latent representation. Therefore, it is critical to adjust the information bottleneck to the appropriate level, which can raise the difficulty in hyperparameter tuning in practice. Also, another drawback of the system is that the disentanglement error can have a relatively larger variance. When repetitively training three times without and with the mutual information regulariser, the variance of disentanglement error raises from 0.0002 to 0.0014. This is because we use sampling to obtain the mutual information term. Finally, we apply the mesh topology to the reconstructed point clouds to evaluate mesh quality in regard to self-intersecting faces (fewer is better). On the CoMA dataset, compared



(a)

(b)

Figure 5.3. Mutual information regulariser weights' impact on: (a) disentanglement error and (b) reconstruction error, CoMA dataset

to the ground truth data, which has 548.60 intersecting faces per mesh, the reconstructed data has an average of 510.06 intersecting faces per mesh. For the BU3DFE dataset, the number of intersecting faces per-mesh is 0.012 and 0.020 for ground truth and reconstruction respectively.

Although we obtained the state-of-the-art disentanglement result on BU-3DFE dataset, the performance on unseen identities in the BU-3DFE dataset remains challenging, as the information bottleneck is not as effective as when it is applied to the CoMA dataset. We evaluate the effectiveness of mutual information loss on BU-3DFE. Raising β_{mi} from 0 to 50 only results in a small decrease of the disentanglement error from 0.332 to 0.328. The reconstructed neutral for $\beta_{mi} = 50$ has an average vertices distance of 2.264 mm, increased from 2.253 mm when $\beta_{mi} = 0$. The main reasons for the non-ideal overall performance on BU-3DFE are: 1) part of the faces are not well registered with each other, resulting in small pose differences and noise; 2) lack of data causes difficulties in avoiding overfitting in the current setup, and $\mathcal{L}_{mi}$ on a smaller dataset is not as effective as on a larger dataset *e.g.* CoMA; 3)

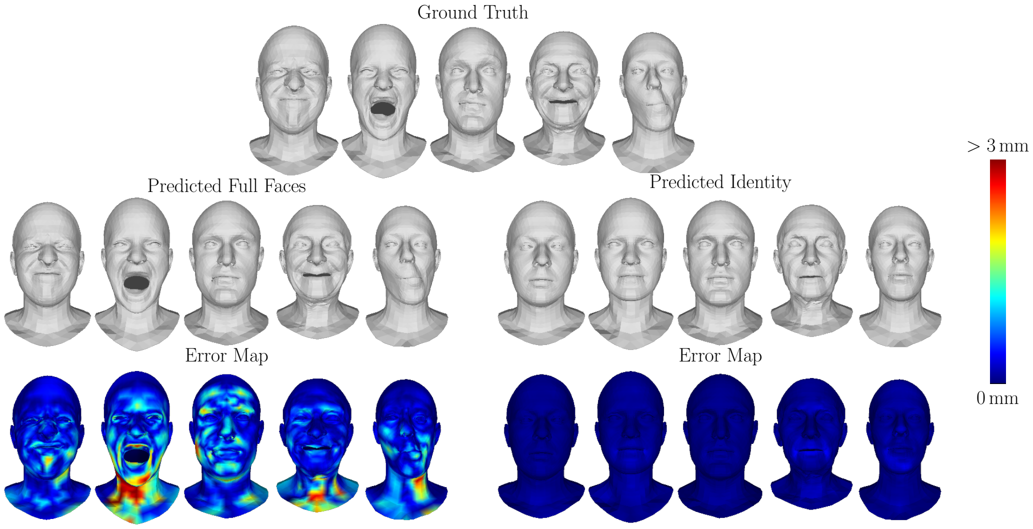


Figure 5.4. Visualisation of reconstruction quality on both identity and full face.

each subject contains only 24 faces with expressions, which is insufficient compared to $\sim 1,200$ per subject in the CoMA dataset; 4) 100 different identities are present, making it easier to learn the variation of expressions than identities, thus the neutral AVD is higher.

5.3.5 VAE Qualitative Evaluation

In Fig. 5.4, we demonstrate the reconstruction quality for both identities and full faces (*i.e.* with expression). With the information bottleneck applied, the reconstructed neutral faces have extremely low error.

In Fig. 5.5, we visualise the identity latent code on the model trained on BU-3DFE dataset by dimension reduction using PCA. We show that the learned model clusters the latent representation for similar faces.

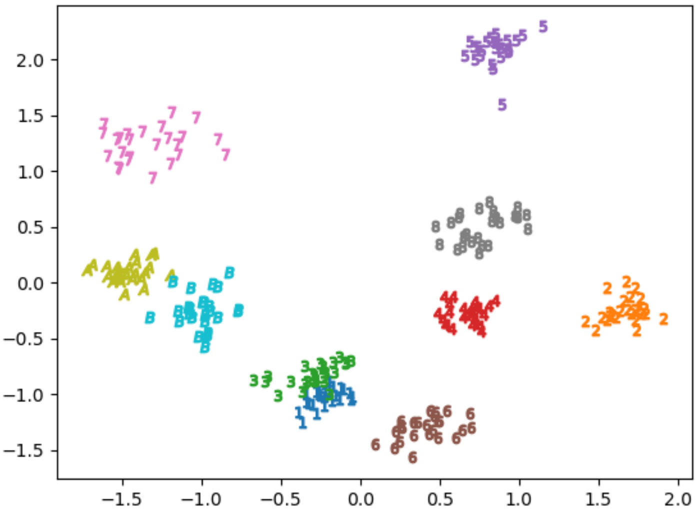


Figure 5.5. Scatter plot of PCA-processed identity latent code on BU-3DFE dataset. The axes represent dimensionally-reduced latent code values. Different subjects are marked with their hexadecimal ID and with different colours.

5.3.6 Conditional VAE Evaluation

We also perform evaluations using a conditional VAE on CoMA dataset. A Support Vector Machine (SVM) is trained and evaluated on the test set to predict the subject ID from the identity latent code $\mathbf{z}_{id}$ and expression latent code $\mathbf{z}_{exp}$ respectively. The accuracy results are 100% and 45.4%, while a random predictor will give an accuracy of 8.3%. The high accuracy result from $\mathbf{z}_{id}$ shows that the identity latent code contains adequate information for identities and different identities are trivial to separate. The high accuracy result from $\mathbf{z}_{exp}$ shows that expression latent code does contain information about identity. This is expected, since everyone has a different way of expressing the same expression, such that the expression latent code encodes personalised expressions. To further separate expression information out of $\mathbf{z}_{exp}$, we utilise dataset labels to train a conditional VAE on top of our proposed architecture. The identity recognition results are 100% for the identity code and 16.0% for the expression latent code. The drop of accuracy for expression is mainly because 3 out of 4 expression latent variables are collapsed to the standard Gaussian distribution, while the remaining one variable encodes the mouth direction. Visual details are demonstrated in our qualitative experiments. This reduces the uninterpretable latent variables from 8 to 5. Our conditional VAE gives a reconstruction error of 0.740 mm and disentanglement error of 0.008 mm. It enables the generation of expressions with semantically-meaningful expression labels in exchange for a small performance drop.

We also perform evaluations using a CVAE using the same hyperparameters as the proposed method on CoMA data. By providing 12 explainable expression-level variables to the decoder, 3 out of 4 expression latent variables are collapsed to the standard Gaussian distribution, while the remaining variable encodes mouth direction (visual details in Fig. 5.6). This reduces the uninterpretable latent variables from 8 to 5. That is 4 uninterpretable variables for identity, 1 uninterpretable variable and 12 interpretable variables for expression. Our CVAE gives a reconstruction error of 0.740 mm and disentanglement error of 0.008 mm. It enables the generation of expressions with semantic expression labels in exchange for a small performance drop.

In Fig. 5.6, we use the full face decoder part of the conditional VAE to generate faces directly. The upper row demonstrates the different expressions generated upon a fixed identity by providing an expression level for selected expression. We can also control which expression to generate by changing the corresponding variables. The CoMA dataset only provides one label for the mouth going to both left and right side, however, the conditional VAE still captures that information and stores it as one variable in $\mathbf{z}_{exp}$. Given other

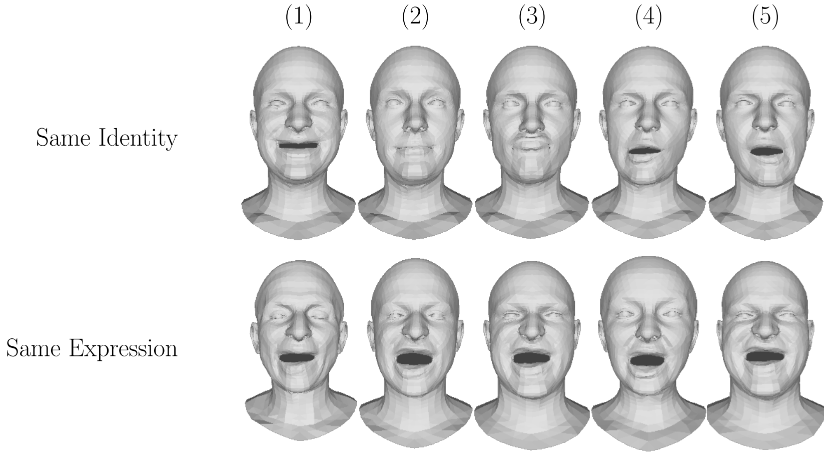


Figure 5.6. Conditional VAE generated samples. Upper row: same identity, different expressions; Lower row: same expression (mouth extreme), different identities. The expressions generated on the first row are: (1) bare teeth. (2) eyebrow, (3) lips up, (4) mouth side (left) and (5) mouth side (right)

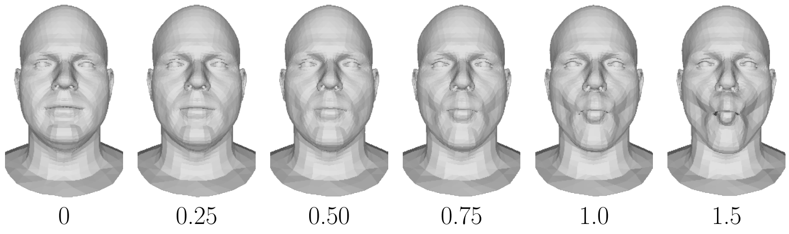


Figure 5.7. Conditional VAE generated samples with gradual increasing expression level on *cheeks in* expression, while keeping all the rest latent variables fixed.

latent variables are collapsed to a prior, modifying this uninterpretable variable results in (4) and (5) on the first row of Fig. 5.6. The bottom row shows generating the *mouth extreme* expression using different identities. Finally, Fig. 5.7 shows that with the CVAE, one can generate certain expressions with different levels of intensity.

5.4 Future Works

There are a number of potential future works. With the PCT as our encoder, the network itself is designed to work on point clouds whose vertices are permutation invariant. As PCT designs its architecture in transformer style, it can take point cloud inputs that vary in point amounts. This allows for the potential to combine multiple datasets to counter the issue of lack of data. Another interesting direction is to remove the restriction that demands a corresponding neutral face for every face with expression. Finally, the information bottleneck in this paper has the potential to be further optimised. Currently, it requires multiple experiments to find the best weights of the information bottleneck, this process can be potentially optimised as [150] proposes an optimal boundary for the information bottleneck.

5.5 Conclusion

We demonstrated identity and expression disentanglement, using an intuitive structure with an additional information bottleneck on the identity sub-system. We showed that the information bottleneck can be integrated with the current VAE training structure by adding an additional mutual information loss. Future work may include finding the optimal boundary for optimised weight selection and further increasing the method's efficiency for datasets with fewer scans per subject. Our results show that the our architecture performs better than the current state-of-the-art in term of disentanglement performance. Furthermore, with use of a CVAE, we are able to generate expressions using expression labels and their corresponding expression levels.

6 Conclusions

In this chapter, we summarise what has been achieved and give general conclusions. Finally, we discuss the potential future work, given the work that has been presented in this thesis.

6.1 Thesis Summary

In this thesis, we firstly present an overview of the field of research of 3D human-related object reconstruction including ears, eyes, eye regions, faces and facial expressions. In the literature review chapter, we review all the techniques and predecessor research works in detail. Then, we present the three technical works done during the PhD study. Summaries of these are in the following three subsections.

6.1.1 Human Ear: the HERA System

This work aims at reconstructing the underlying 3D ear geometry and colour details, given a monocular RGB input image. Modelling of ear shapes is an important part of human head modelling, yet it has received far less attention from the computer vision community, when compared to the face modelling. Inspired by previous work on monocular 3D face reconstruction using an autoencoder structure to achieve unsupervised learning, we aim to utilise such a framework to tackle the 3D ear reconstruction task, where more subtle and difficult curves and features are present. Our Human Ear Reconstruction Autoencoder (HERA) system predicts 3D ear poses and shape parameters for 3D ear meshes, without any supervision to these parameters. To make our approach cover the variance for in-the-wild images, even grayscale images, we propose an in-the-wild ear colour model. The constructed end-to-end model is then evaluated both with 2D landmark localisation performance and the appearance of the reconstructed 3D ears. Furthermore, we predict 3D ear landmarks on raw 3D head scans from the Headspace dataset. Such prediction is refined by an iterative ear model fitting process, after model pose and shape initialisation using the HERA system.

6.1.2 Human Gaze and Eye Region: Active-Gaze Morphable Model

Recently, appearance-based methods using deep networks to regress gaze direction directly from raw images have been extremely popular. While most of these methods focus on network architecture and loss function improvements, we show that adding a 3D shape model to regularise the network training process can: i) improve gaze estimation accuracy, ii) perform well with lower resolution inputs and iii) provide a richer understanding of the human eye-region and its constituent gaze system. Specifically, we use an 'eyes plus nose' 3D morphable model (3DMM) to capture the eye-region 3D geometry and appearance, and we equip this with a geometric vergence model of gaze to give an 'active-gaze 3DMM'. Specifically, this enables the combined rotation of the eyeballs for the expression of gaze under certain geometric constraints, such as coplanarity of the gaze vectors. This ensures accurate gaze estimation and eyeball positions that are consistent with both the face geometry and head pose. We show that our approach achieves state-of-the-art results on the *Eyediap* dataset and extensive ablation studies illustrate the contribution of each component. We also demonstrate that our method can learn with only the ground truth gaze target point and the camera parameters, without access to the ground truth gaze origin points. This widens the applicability of our approach compared to other methods.

6.1.3 Facial Expression Disentanglement VAE

Learning a disentangled representation is essential to build 3D face models that accurately capture identity and expression. We propose a novel variational autoencoder (VAE) framework to disentangle identity and expression from 3D input faces that have a wide variety of expressions. Specifically, we design a system that has two decoders: one for neutral-expression faces (i.e. identity-only faces) and one for the original (expressive) input faces respectively. Crucially, we have an additional mutual-information regulariser applied on the identity part to solve the issue of imbalanced information over the expressive input faces and the reconstructed neutral faces. Our evaluations on two public datasets (CoMA and BU-3DFE) show that this model achieves competitive results on the 3D face reconstruction task and state-of-the-art results on identity-expression disentanglement. We also show that by updating to a conditional VAE, we have a system that generates different levels of expressions from semantically meaningful variables.

6.2 Conclusions

In this thesis, we present various techniques for reconstructing different human head related objects with various input types and focuses. The initiative of the whole research is that when a reconstruction or modelling algorithm focuses on the whole object (*e.g.* human head) the details of the smaller objects (*e.g.* ears, eyes) on the bigger object can be easily ignored. That is, if the modelling of the whole head is employed, it is hard to learn the high frequency details of the ears and eyes. One solution, also the solution we proposed, is to model each part individually and joint them together to a composite model. In such way, the task of each model is more designated and we always have the option to tune each model individually to suit the needs.

Our proposed methods also diversify the forms of the data that we process and generate. For the ear and the eye works, our methods can process monocular 2D RGB images. For the facial expression work, our method processes 3D point cloud data. Both types of data are very common in real deployment. Standard 2D RGB images probably are the most common data type that can be obtained easily with mobile phones or webcams. Therefore, the methods have potential to be deployed to smart phones and home computers to obtain underlying 3D structures of eye-regions and ears. On the other hand, point cloud is a wildly used data type and can be obtained by modern mobile phones with a multi-camera system or a Lidar sensor.

There are also some general drawbacks of the proposed methods. Starting from the 2D-to-2D approaches for ear and eyes, where the input data contains no 3D information, this makes the task fully reliant on the underlying 3D model. Meanwhile, the proposed camera model is a depth-unaware scaled orthogonal projection model. These two factors jointly create more ambiguities in 3D coordinates such as pose ambiguity and pose-structure ambiguity similar to [151]. The other issue about the 2D-to-2D approaches is that although they can train without any supervision, the difficulties of training them are much harder than those with landmarks provided. Sometimes additional loss functions have to be employed to prevent the model from deviating too much [78]. We now consider the 3D-to-3D approach for facial expressions, where the whole pipeline is more concise, since no 3D-to-2D conversion is needed. The biggest limitation of this method is that it suffers from the curse of dimensionality. The current CoMA dataset contains head meshes with ~ 5000 vertices, but the real scans of the whole head can easily exceed 100000 vertices. The increase in the size of the input point cloud will put extra pressure on both the encoder network and the decoder network. Also, the experiments are done using registered point clouds. That is, although the method can process point clouds, there is still the need for it to learn in a fully unsupervised manner with real world scans. Mitigating this issue remains a general open question and can potentially be solved in the future by more cleverly designed networks or by better hardware equipment.

For the backbone networks we employed, we use popular or state-of-the-art networks at the time of the experiments. However, it is a fast developing area that we use a different backbone network for each of the three presented works. The analysis of applying different backbone networks to both our and other existing works remains unexplored.

6.3 Future Work

In this section, we will discuss the potential future works based on our three contributions, in order to show the their potential. The technical aspects will be discussed first, followed by the general direction of such a design choice of model-based methods. Finally, the potential of a composite model that contains individual models like those that we proposed will be discussed.

More Data Types One future work includes extending to more data types to analyse and synthesise. Since a number of robotic systems have both RGB camera and depth camera installed [152], they can feed in data with corresponded RGB images and depth images. This provides richer information than sole RGB images discussed above, such that depth information can be utilised to better infer the underlying 3D structure. The fusion of both features can be an interesting direction to go in 3D reconstruction tasks. Similar networks have been proposed to fuse both features, but they have not been applied to any 3D human-related object reconstruction [153]. On the other hand, volumetric data are not commonly used to create a statistical model or applied to 3D human-related object reconstruction. Volumetric data has been extensively used for medical imaging [26], and they have received much less attention for 3D volumetric reconstruction or statistical modelling of medical images compared to RGB images or point cloud data.

Better Model The synthesised images or point clouds from the model can have an implausible appearance, especially for the 2D image methods. Most of the literature uses a simplistic differentiable renderer where no lighting model or a simple lighting model is used. This simplifies the task and makes the deep neural networks easier to train. However, along with the linear texture model (some focus on realistic texture model, *e.g.* [32]), the simplistic renderer often synthesises unrealistic images. This situation is worse given more non-trivial surfaces, such as eyeball surfaces, which are moist and makes the reflection calculation much harder. Given all these, it makes improving a model-based generation model's plausibility an important direction to develop. In terms of plausibility, the end-to-end network-based generation methods (*e.g.* GAN) are the state-of-the-art methods for generating 2D images in an implicit way nowadays. However, generative models can still greatly benefit from underlying models to encode more valuable and versatile information explicitly. All our works explore different ways of the fusion of modern end-to-end network-based methods and model-based methods to benefit from both advantages.

Composite Model The ultimate goal of the direction that this thesis proposes is the ability to get a composite model that can represent and manipulate different properties given a high-quality raw scan of the human head, and has enough detail on every important sub-object. This thesis focuses on building the sub-objects' models individually, and leaves the composition as a future work. Although there is current work that aims to model-fit a complete head with sub-parts [61], we solve a similar problem with deep learning approaches instead of a model-fitting approach. This has various advantages, including fast inference time compared to model-fitting, and strong feature extraction ability that maps the raw features to model parameters directly with extra information that are of high accuracy, *e.g.* our active gaze morphable model system. Being able to composite all the sub-models together is a valuable future research direction.

Fully Unsupervised Learning For the topic of supervised vs. unsupervised learning, our works are enabled to perform unsupervised learning. However, additional supervision is often required to obtain satisfying results. In the meantime, under some of the circumstances, a learned model is required for providing ground truths, thus achieve unsupervised learning. This can introduce errors and bottlenecks to the unsupervised learning process, then impair the learning performance. To achieve high performance fully unsupervised learning for our tasks without any labels either from manual work or learned models remains unsolvable. It is of great importance to the research community if the high performance fully unsupervised learning for any of our works can be solved in the future.

Bibliography

[1] Y. Zhou and S. Zaferiou, “Deformable models of ears in-the-wild for alignment and recognition,” in *2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)*, pp. 626–633, IEEE, 2017.

[2] N. Ravi, J. Reizenstein, D. Novotny, T. Gordon, W.-Y. Lo, J. Johnson, and G. Gkioxari, “Accelerating 3d deep learning with pytorch3d,” *arXiv:2007.08501*, 2020.

[3] T. Li, T. Bolkart, M. J. Black, H. Li, and J. Romero, “Learning a model of facial shape and expression from 4D scans,” *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, vol. 36, no. 6, pp. 194:1–194:17, 2017.

[4] Q. Tan, L.-X. Zhang, J. Yang, Y.-K. Lai, and L. Gao, “Mesh-based variational autoencoders for localized deformation component analysis,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[5] J. Yang, L. Gao, Q. Tan, Y. Huang, S. Xia, and Y.-K. Lai, “Multiscale mesh deformation component analysis with attention-based autoencoders,” *arXiv preprint arXiv:2012.02459*, 2020.

[6] A. Tewari, M. Zollhofer, H. Kim, P. Garrido, F. Bernard, P. Perez, and C. Theobalt, “Mofa: Model-based deep convolutional face autoencoder for unsupervised monocular reconstruction,” in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pp. 1274–1283, 2017.

[7] K. J. Holyoak, “Parallel distributed processing: explorations in the microstructure of cognition,” *Science*, vol. 236, pp. 992–997, 1987.

[8] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[9] H. Kim and A. Mnih, “Disentangling by factorising,” in *International Conference on Machine Learning*, pp. 2649–2658, PMLR, 2018.

[10] H. Dai, N. Pears, and W. Smith, “Augmenting a 3D morphable model of the human head with high resolution ears,” *Pattern Recognition Letters*, vol. 128, pp. 378–384, 2019.

[11] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” *arXiv preprint arXiv:1312.6114*, 2013.

[12] V. Blanz and T. Vetter, “A morphable model for the synthesis of 3d faces,” in *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pp. 187–194, 1999.

[13] P. Baldi, “Autoencoders, unsupervised learning, and deep architectures,” in *Proceedings of ICML workshop on unsupervised and transfer learning*, pp. 37–49, JMLR Workshop and Conference Proceedings, 2012.

[14] E. Plaut, “From principal subspaces to principal components with linear autoencoders,” *arXiv preprint arXiv:1804.10253*, 2018.

[15] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, “SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python,” *Nature Methods*, vol. 17, pp. 261–272, 2020.

[16] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, “Pointnet: Deep learning on point sets for 3d classification and segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 652–660, 2017.

[17] M.-H. Guo, J.-X. Cai, Z.-N. Liu, T.-J. Mu, R. R. Martin, and S.-M. Hu, “Pct: Point cloud transformer,” *arXiv preprint arXiv:2012.09688*, 2020.

[18] H. Zhao, L. Jiang, J. Jia, P. H. Torr, and V. Koltun, "Point transformer," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16259–16268, 2021.

[19] J. Pang, D. Li, and D. Tian, "Tearingnet: Point cloud autoencoder to learn topology-friendly representations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7453–7462, 2021.

[20] Y. Yang, C. Feng, Y. Shen, and D. Tian, "Foldingnet: Point cloud auto-encoder via deep grid deformation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 206–215, 2018.

[21] A. Ranjan, T. Bolkart, S. Sanyal, and M. J. Black, "Generating 3d faces using convolutional mesh autoencoders," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 704–720, 2018.

[22] T. Bagautdinov, C. Wu, J. Saragih, P. Fua, and Y. Sheikh, "Modeling facial geometry using compositional vaes," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3877–3886, 2018.

[23] K. Li, J. Liu, Y.-K. Lai, and J. Yang, "Generating 3d faces using multi-column graph convolutional networks," in *Computer Graphics Forum*, vol. 38, pp. 215–224, Wiley Online Library, 2019.

[24] C. Yuan, K. Li, Y.-K. Lai, Y. Liu, and J. Yang, "3d face reprentation and reconstruction with multi-scale graph convolutional autoencoders," in *2019 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1558–1563, IEEE, 2019.

[25] F. Milletari, N. Navab, and S.-A. Ahmadi, "V-net: Fully convolutional neural networks for volumetric medical image segmentation," in *2016 fourth international conference on 3D vision (3DV)*, pp. 565–571, IEEE, 2016.

[26] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241, Springer, 2015.

[27] M. Tatarchenko, A. Dosovitskiy, and T. Brox, “Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs,” in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2088–2096, 2017.

[28] G. Riegler, A. Osman Ulusoy, and A. Geiger, “Octnet: Learning deep 3d representations at high resolutions,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3577–3586, 2017.

[29] C. Häne, S. Tulsiani, and J. Malik, “Hierarchical surface prediction for 3d object reconstruction,” in *2017 International Conference on 3D Vision (3DV)*, pp. 412–420, IEEE, 2017.

[30] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, “Deepsdf: Learning continuous signed distance functions for shape representation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 165–174, 2019.

[31] L. Tran and X. Liu, “Nonlinear 3d face morphable model,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7346–7355, 2018.

[32] B. Gecer, S. Ploumpis, I. Kotsia, and S. Zafeiriou, “Ganfit: Generative adversarial network fitting for high fidelity 3d face reconstruction,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1155–1164, 2019.

[33] A. Ng *et al.*, “Sparse autoencoder,” *CS294A Lecture notes*, vol. 72, no. 2011, pp. 1–19, 2011.

[34] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, “Extracting and composing robust features with denoising autoencoders,” in *Proceedings of the 25th international conference on Machine learning*, pp. 1096–1103, 2008.

[35] S. Rifai, P. Vincent, X. Muller, X. Glorot, and Y. Bengio, “Contractive auto-encoders: Explicit invariance during feature extraction,” in *Icml*, 2011.

[36] C. Doersch, “Tutorial on variational autoencoders,” *arXiv preprint arXiv:1606.05908*, 2016.

[37] T. Salimans and D. A. Knowles, "Fixed-form variational posterior approximation through stochastic linear regression," *Bayesian Analysis*, vol. 8, no. 4, pp. 837–882, 2013.

[38] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[39] J. Schmidhuber, "Learning factorial codes by predictability minimization," *Neural computation*, vol. 4, no. 6, pp. 863–879, 1992.

[40] G. Desjardins, A. Courville, and Y. Bengio, "Disentangling factors of variation via generative entangling," *arXiv preprint arXiv:1210.5474*, 2012.

[41] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey, "Adversarial autoencoders," *arXiv preprint arXiv:1511.05644*, 2015.

[42] A. Makhzani and B. J. Frey, "Pixelgan autoencoders," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[43] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "Infogan: Interpretable representation learning by information maximizing generative adversarial nets," *Advances in neural information processing systems*, vol. 29, 2016.

[44] T. D. Kulkarni, W. F. Whitney, P. Kohli, and J. Tenenbaum, "Deep convolutional inverse graphics network," *Advances in neural information processing systems*, vol. 28, 2015.

[45] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, "beta-vae: Learning basic visual concepts with a constrained variational framework," 2016.

[46] R. T. Chen, X. Li, R. B. Grosse, and D. K. Duvenaud, "Isolating sources of disentanglement in variational autoencoders," *Advances in neural information processing systems*, vol. 31, 2018.

[47] H. W. Kuhn and A. W. Tucker, "Nonlinear programming," in *Traces and emergence of nonlinear programming*, pp. 247–258, Springer, 2014.

[48] W. Karush, "Minima of functions of several variables with inequalities as side constraints," *M. Sc. Dissertation. Dept. of Mathematics, Univ. of Chicago*, 1939.

[49] X. Nguyen, M. J. Wainwright, and M. I. Jordan, “Estimating divergence functionals and the likelihood ratio by convex risk minimization,” *IEEE Transactions on Information Theory*, vol. 56, no. 11, pp. 5847–5861, 2010.

[50] M. Sugiyama, T. Suzuki, and T. Kanamori, “Density-ratio matching under the bregman divergence: a unified framework of density-ratio estimation,” *Annals of the Institute of Statistical Mathematics*, vol. 64, no. 5, pp. 1009–1044, 2012.

[51] H. Sun, N. Pears, and Y. Gu, “Information bottlenecked variational autoencoder for disentangled 3d facial expression modelling,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 157–166, 2022.

[52] A. A. Alemi, I. Fischer, J. V. Dillon, and K. Murphy, “Deep variational information bottleneck,” *arXiv preprint arXiv:1612.00410*, 2016.

[53] N. Tishby, F. C. Pereira, and W. Bialek, “The information bottleneck method,” *arXiv preprint physics/0004057*, 2000.

[54] Z. Zhang, C. Yu, H. Li, J. Sun, and F. Liu, “Learning distribution independent latent representation for 3d face disentanglement,” in *2020 International Conference on 3D Vision (3DV)*, pp. 848–857, IEEE, 2020.

[55] S. Zhao, J. Song, and S. Ermon, “Infovae: Information maximizing variational autoencoders,” *arXiv preprint arXiv:1706.02262*, 2017.

[56] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial networks,” *arXiv preprint arXiv:1406.2661*, 2014.

[57] J. Thies, M. Zollhöfer, M. Nießner, L. Valgaerts, M. Stamminger, and C. Theobalt, “Real-time expression transfer for facial reenactment.,” *ACM Trans. Graph.*, vol. 34, no. 6, pp. 183–1, 2015.

[58] IEEE, *A 3D Face Model for Pose and Illumination Invariant Face Recognition*, (Genova, Italy), 2009.

[59] J. Booth, E. Antonakos, S. Ploumpis, G. Trigeorgis, Y. Panagakis, and S. Zafeiriou, “3d face morphable models” in-the-wild”,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 48–57, 2017.

[60] H. Dai, N. Pears, W. Smith, and C. Duncan, “Statistical modeling of craniofacial shape and texture,” *International Journal of Computer Vision*, vol. 128, pp. 547–571, Nov 2019.

[61] S. Ploumpis, E. Ververas, E. O’Sullivan, S. Moschoglou, H. Wang, N. Pears, W. Smith, B. Gecer, and S. P. Zafeiriou, “Towards a complete 3d morphable model of the human head,” *IEEE transactions on pattern analysis and machine intelligence*, 2020.

[62] C. Ferrari, G. Lisanti, S. Berretti, and A. Del Bimbo, “Dictionary learning based 3d morphable model construction for face recognition with varying expression and pose,” in *2015 International Conference on 3D Vision*, pp. 509–517, IEEE, 2015.

[63] A. Brunton, T. Bolkart, and S. Wuhrer, “Multilinear wavelets: A statistical shape space for human faces,” in *European Conference on Computer Vision*, pp. 297–312, Springer, 2014.

[64] P. Koppen, Z.-H. Feng, J. Kittler, M. Awais, W. Christmas, X.-J. Wu, and H.-F. Yin, “Gaussian mixture 3d morphable face model,” *Pattern Recognition*, vol. 74, pp. 617–628, 2018.

[65] E. Wood, T. Baltrušaitis, L.-P. Morency, P. Robinson, and A. Bulling, “A 3d morphable eye region model for gaze estimation,” in *European Conference on Computer Vision*, pp. 297–313, Springer, 2016.

[66] E. Wood, T. Baltrušaitis, L.-P. Morency, P. Robinson, and A. Bulling, “A 3d morphable model of the eye region,” *Optimization*, vol. 1, p. 0, 2016.

[67] P. Garrido, M. Zollhöfer, D. Casas, L. Valgaerts, K. Varanasi, P. Pérez, and C. Theobalt, “Reconstruction of personalized 3d face rigs from monocular video,” *ACM Transactions on Graphics (TOG)*, vol. 35, no. 3, pp. 1–15, 2016.

[68] J. Thies, M. Zollhofer, M. Stamminger, C. Theobalt, and M. Nießner, “Face2face: Real-time face capture and reenactment of rgb videos,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2387–2395, 2016.

[69] S. Suwajanakorn, I. Kemelmacher-Shlizerman, and S. M. Seitz, “Total moving face reconstruction,” in *European conference on computer vision*, pp. 796–812, Springer, 2014.

[70] S. Suwajanakorn, S. M. Seitz, and I. Kemelmacher-Shlizerman, “What makes tom hanks look like tom hanks,” in *Proceedings of the IEEE international conference on computer vision*, pp. 3952–3960, 2015.

[71] I. Kemelmacher-Shlizerman, A. Sankar, E. Shechtman, and S. M. Seitz, “Being john malkovich,” in *European Conference on Computer Vision*, pp. 341–353, Springer, 2010.

[72] E. Richardson, M. Sela, R. Or-El, and R. Kimmel, “Learning detailed face reconstruction from a single image,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1259–1268, 2017.

[73] A. Tuan Tran, T. Hassner, I. Masi, and G. Medioni, “Regressing robust and discriminative 3d morphable models with a very deep neural network,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5163–5172, 2017.

[74] X. Zhu, Z. Lei, X. Liu, H. Shi, and S. Z. Li, “Face alignment across large poses: A 3d solution,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 146–155, 2016.

[75] Y. Feng, F. Wu, X. Shao, Y. Wang, and X. Zhou, “Joint 3d face reconstruction and dense alignment with position map regression network,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 534–551, 2018.

[76] Y. Sun, X. Wang, and X. Tang, “Deep convolutional network cascade for facial point detection,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3476–3483, 2013.

[77] E. Zhou, H. Fan, Z. Cao, Y. Jiang, and Q. Yin, “Extensive facial landmark localization with coarse-to-fine convolutional network cascade,” in *Proceedings of the IEEE international conference on computer vision workshops*, pp. 386–391, 2013.

[78] H. Sun, N. Pears, and H. Dai, “A human ear reconstruction autoencoder,” in *16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, Springer International Publishing, 2021.

[79] T. F. Cootes, G. J. Edwards, and C. J. Taylor, “Active appearance models,” in *European conference on computer vision*, pp. 484–498, Springer, 1998.

[80] H. Dai, N. Pears, and W. Smith, "A data-augmented 3D morphable model of the ear," in *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*, pp. 404–408, IEEE, 2018.

[81] Ž. Emeršič, L. L. Gabriel, V. Štruc, and P. Peer, "Pixel-wise ear detection with convolutional encoder-decoder networks," *arXiv preprint arXiv:1702.00307*, 2017.

[82] M. Bizjak, P. Peer, and Ž. Emeršič, "Mask r-cnn for ear detection," in *2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pp. 1624–1628, IEEE, 2019.

[83] X. Zhu, X. Liu, Z. Lei, and S. Z. Li, "Face alignment in full pose range: A 3d total solution," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 1, pp. 78–92, 2017.

[84] F. Liu, D. Zeng, Q. Zhao, and X. Liu, "Joint face alignment and 3d face reconstruction," in *European Conference on Computer Vision*, pp. 545–560, Springer, 2016.

[85] J. McDonagh and G. Tzimiropoulos, "Joint face detection and alignment with a deformable hough transform model," in *European Conference on Computer Vision*, pp. 569–580, Springer, 2016.

[86] E. E. Hansley, M. P. Segundo, and S. Sarkar, "Employing fusion of learned and handcrafted features for unconstrained ear recognition," *IET Biometrics*, vol. 7, no. 3, pp. 215–223, 2018.

[87] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," *arXiv preprint arXiv:1312.6203*, 2013.

[88] F. Liu, L. Tran, and X. Liu, "3d face modeling from diverse raw scan data," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9408–9418, 2019.

[89] H. Dai and L. Shao, "Pointae: Point auto-encoder for 3d statistical shape and texture modelling," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5410–5419, 2019.

[90] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *arXiv preprint arXiv:1706.03762*, 2017.

[91] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[92] S. Gong, L. Chen, M. Bronstein, and S. Zafeiriou, "Spiralnet++: A fast and highly efficient mesh convolution operator," in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pp. 0–0, 2019.

[93] T. Gerig, A. Morel-Forster, C. Blumer, B. Egger, M. Luthi, S. Schönborn, and T. Vetter, "Morphable face models-an open framework," in *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*, pp. 75–82, IEEE, 2018.

[94] C. Cao, Y. Weng, S. Zhou, Y. Tong, and K. Zhou, "Facewarehouse: A 3d facial expression database for visual computing," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 3, pp. 413–425, 2013.

[95] Z.-H. Jiang, Q. Wu, K. Chen, and J. Zhang, "Disentangled representation learning for 3d face shape," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11957–11966, 2019.

[96] B. Egger, W. A. Smith, A. Tewari, S. Wuhrer, M. Zollhoefer, T. Beeler, F. Bernard, T. Bolkart, A. Kortylewski, S. Romdhani, *et al.*, "3d morphable face models—past, present, and future," *ACM Transactions on Graphics (TOG)*, vol. 39, no. 5, pp. 1–38, 2020.

[97] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," in *International Conference on Learning Representations*, 2018.

[98] C. Chan, S. Ginosar, T. Zhou, and A. A. Efros, "Everybody dance now," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 5933–5942, 2019.

[99] T. Li, T. Bolkart, M. J. Black, H. Li, and J. Romero, "Learning a model of facial shape and expression from 4d scans.," *ACM Trans. Graph.*, vol. 36, no. 6, pp. 194–1, 2017.

[100] L. Li, J. Bao, H. Yang, D. Chen, and F. Wen, "Faceshifter: Towards high fidelity and occlusion aware face swapping,"

[101] Y. Shen, J. Gu, X. Tang, and B. Zhou, "Interpreting the latent space of gans for semantic face editing," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9243–9252, 2020.

[102] M. Kocaoglu, C. Snyder, A. G. Dimakis, and S. Vishwanath, "Causalgan: Learning causal implicit generative models with adversarial training," in *International Conference on Learning Representations*, 2018.

[103] Ž. Emeršič, V. Štruc, and P. Peer, "Ear recognition: More than a survey," *Neurocomputing*, vol. 255, pp. 26–39, 2017.

[104] Ž. Emeršič, A. K. SV, B. Harish, W. Gutfeter, J. Khiarak, A. Pacut, E. Hansley, M. P. Segundo, S. Sarkar, H. Park, *et al.*, "The unconstrained ear recognition challenge 2019," in *2019 International Conference on Biometrics (ICB)*, pp. 1–15, IEEE, 2019.

[105] H. Dai, N. Pears, W. A. P. Smith, and C. Duncan, "A 3d morphable model of craniofacial shape and texture variation," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[106] H. Dai, N. Pears, W. Smith, and C. Duncan, "Statistical modeling of craniofacial shape and texture," *International Journal of Computer Vision*, vol. 128, no. 2, pp. 547–571, 2020.

[107] H. Dai, N. Pears, P. Huber, and W. A. Smith, "3d morphable models: The face, ear and head," in *3D Imaging, Analysis and Applications*, pp. 463–512, Springer, 2020.

[108] E. Richardson, M. Sela, and R. Kimmel, "3d face reconstruction by learning from synthetic data," in *2016 fourth international conference on 3D vision (3DV)*, pp. 460–469, IEEE, 2016.

[109] M. Zollhöfer, J. Thies, P. Garrido, D. Bradley, T. Beeler, P. Pérez, M. Stamminger, M. Nießner, and C. Theobalt, "State of the art on monocular 3d face reconstruction, tracking, and applications," in *Computer Graphics Forum*, pp. 523–550, Wiley Online Library, 2018.

[110] A. Kessy, A. Lewin, and K. Strimmer, "Optimal whitening and decorrelation," *The American Statistician*, vol. 72, no. 4, pp. 309–314, 2018.

[111] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[112] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in neural information processing systems*, pp. 8026–8037, 2019.

[113] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255, Ieee, 2009.

[114] Z.-Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu, "Object detection with deep learning: A review," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 11, pp. 3212–3232, 2019.

[115] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[116] O. Van Kaick, H. Zhang, G. Hamarneh, and D. Cohen-Or, "A survey on shape correspondence," in *Computer graphics forum*, vol. 30, pp. 1681–1707, Wiley Online Library, 2011.

[117] T. Groueix, M. Fisher, V. G. Kim, B. C. Russell, and M. Aubry, "3d-coded: 3d correspondences by deep deformation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 230–246, 2018.

[118] E. Rodolà, L. Cosmo, M. M. Bronstein, A. Torsello, and D. Cremers, "Partial functional correspondence," in *Computer graphics forum*, vol. 36, pp. 222–236, Wiley Online Library, 2017.

[119] B. Amberg, S. Romdhani, and T. Vetter, "Optimal step nonrigid icp algorithms for surface registration," in *2007 IEEE conference on computer vision and pattern recognition*, pp. 1–8, IEEE, 2007.

[120] Z. Zhang, "Iterative point matching for registration of free-form curves and surfaces," *International journal of computer vision*, vol. 13, no. 2, pp. 119–152, 1994.

[121] J. A. Nelder and R. Mead, "A simplex method for function minimization," *The computer journal*, vol. 7, no. 4, pp. 308–313, 1965.

[122] G. K. Tam, Z.-Q. Cheng, Y.-K. Lai, F. C. Langbein, Y. Liu, D. Marshall, R. R. Martin, X.-F. Sun, and P. L. Rosin, "Registration of 3d point clouds and meshes: A survey from rigid to nonrigid," *IEEE transactions on visualization and computer graphics*, vol. 19, no. 7, pp. 1199–1217, 2012.

[123] X. Zhang, Y. Sugano, M. Fritz, and A. Bulling, “It’s written all over your face: Full-face appearance-based gaze estimation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 51–60, 2017.

[124] A. Patney, J. Kim, M. Salvi, A. Kaplanyan, C. Wyman, N. Benty, A. Lefohn, and D. Luebke, “Perceptually-based foveated virtual reality,” in *ACM SIGGRAPH 2016 Emerging Technologies*, pp. 1–2, 2016.

[125] Y. Cheng, H. Wang, Y. Bao, and F. Lu, “Appearance-based gaze estimation with deep learning: A review and benchmark,” *arXiv preprint arXiv:2104.12668*, 2021.

[126] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, “Swin transformer: Hierarchical vision transformer using shifted windows,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 10012–10022, October 2021.

[127] W. A. P. Smith, A. Seck, H. Dee, B. Tiddeman, J. Tenenbaum, and B. Egger, “A morphable face albedo model,” in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5011–5020, 2020.

[128] G. Stratou, A. Ghosh, P. Debevec, and L.-P. Morency, “Effect of illumination on automatic expression recognition: a novel 3D relightable facial database,” in *Proc. International Conference on Automatic Face and Gesture Recognition*, pp. 611–618, 2011.

[129] C. Chen, “PyTorch Face Landmark: A fast and accurate facial landmark detector,” 2021. Open-source software available at https://github.com/cunjian/pytorch_face_landmark.

[130] M. Sandler, A. G. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “Mobilenetv2: Inverted residuals and linear bottlenecks,” in *CVPR*, 2018.

[131] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker, “Multi-pie,” in *2008 8th IEEE International Conference on Automatic Face Gesture Recognition*, pp. 1–8, 2008.

[132] K. A. Funes Mora, F. Monay, and J.-M. Odobez, “Eyediap: A database for the development and evaluation of gaze estimation algorithms from rgb and rgb-d cameras,” in *Proceedings of the Symposium on Eye Tracking Research and Applications*, pp. 255–258, 2014.

[133] X. Zhang, S. Park, T. Beeler, D. Bradley, S. Tang, and O. Hilliges, "Eth-xgaze: A large scale dataset for gaze estimation under extreme head pose and gaze variation," in *European Conference on Computer Vision*, pp. 365–381, Springer, 2020.

[134] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.

[135] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[136] Y. Cheng and F. Lu, "Gaze estimation using transformer," *arXiv preprint arXiv:2105.14424*, 2021.

[137] X. Zhang, Y. Sugano, M. Fritz, and A. Bulling, "Appearance-based gaze estimation in the wild," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4511–4520, 2015.

[138] N. Sinha, M. Balazia, and F. Bremond, "Flame: Facial landmark heatmap activated multimodal gaze estimation," in *2021 17th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2021.

[139] P. Kellnhofer, A. Recasens, S. Stent, W. Matusik, and A. Torralba, "Gaze360: Physically unconstrained gaze estimation in the wild," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6912–6921, 2019.

[140] T. Fischer, H. J. Chang, and Y. Demiris, "Rt-gene: Real-time eye gaze estimation in natural environments," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 334–352, 2018.

[141] Z. Chen and B. E. Shi, "Appearance-based gaze estimation using dilated-convolutions," in *Asian Conference on Computer Vision*, pp. 309–324, Springer, 2018.

[142] Y. Cheng, Y. Bao, and F. Lu, "Puregaze: Purifying gaze feature for generalizable gaze estimation," *arXiv preprint arXiv:2103.13173*, 2021.

[143] X. Zhang, S. Park, T. Beeler, D. Bradley, S. Tang, and O. Hilliges, "Eth-xgaze: A large scale dataset for gaze estimation under extreme head pose and gaze variation," in *European Conference on Computer Vision (ECCV)*, 2020.

[144] X. Cai, B. Chen, J. Zeng, J. Zhang, Y. Sun, X. Wang, Z. Ji, X. Liu, X. Chen, and S. Shan, "Gaze estimation with an ensemble of four architectures," *arXiv preprint arXiv:2107.01980*, 2021.

[145] K. Sohn, H. Lee, and X. Yan, "Learning structured output representation using deep conditional generative models," *Advances in neural information processing systems*, vol. 28, pp. 3483–3491, 2015.

[146] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32* (H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, eds.), pp. 8024–8035, Curran Associates, Inc., 2019.

[147] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

[148] L. Yin, X. Wei, Y. Sun, J. Wang, and M. J. Rosato, "A 3d facial expression database for facial behavior research," in *7th international conference on automatic face and gesture recognition (FGR06)*, pp. 211–216, IEEE, 2006.

[149] N. Tishby and N. Zaslavsky, "Deep learning and the information bottleneck principle," in *2015 IEEE Information Theory Workshop (ITW)*, pp. 1–5, IEEE, 2015.

[150] A. Alemi, B. Poole, I. Fischer, J. Dillon, R. A. Saurus, and K. Murphy, "An information-theoretic analysis of deep latent-variable models," 2018.

[151] E. Sariyanidi, C. J. Zampella, R. T. Schultz, and B. Tunc, "Can facial pose and expression be separated with weak perspective camera?," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7173–7182, 2020.

[152] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *2012 IEEE/RSJ international conference on intelligent robots and systems*, pp. 573–580, IEEE, 2012.

[153] A. Piergiovanni, V. Casser, M. S. Ryoo, and A. Angelova, “4d-net for learned multi-modal alignment,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 15435–15445, 2021.