Combined 3D Eye and Face Reconstruction using Monocular RGB Images

Chitra Singh

Saarland University

Saarbrücken, Germany

October 2018

Faculty of Mathematics and Computer Science

Department of Computer Science

Masters Program in Visual Computing

Supervisor:

Prof. Dr. Christian Theobalt, Max Planck Institute for Informatics

Saarbrücken, Germany

Co-supervisors:

Dr. Florian Bernard, Max Planck Institute for Informatics

Saarbrücken, Germany

M.Sc. Ayush Tewari, Max Planck Institute for Informatics

Saarbrücken, Germany

Reviewers:

Prof. Dr. Christian Theobalt, Max Planck Institute for Informatics

Saarbrücken, Germany

Prof. Dr. Andreas Bulling, University of Stuttgart

Stuttgart, Germany

Dean:

Prof. Dr. Sebastian Hack, Saarland University

Saarbrücken, Germany

Thesis submitted:

29.10.2018

Universität des Saarlandes

Fachrichtung 6.2 - Informatik

Im Stadtwald - Building E 1 3

66123 Saarbrücken

Declarations

Eidesstattliche Erklärung/Statement in Lieu of an Oath:
Ich erkläre hiermit an Eides Statt, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.
I hereby confirm that I have written this thesis on my own and that I have not used any other media or materials than the ones referred to in this thesis.

Signature:

Chitra Singh

Einverständniserklärung/Declaration of Consent:
Ich bin damit einverstanden, dass meine (bestandene) Arbeit in beiden Versionen in die Bibliothek der Informatik aufgenommen und damit veröffentlicht wird.
I agree to make both versions of my thesis (with a passing grade) accessible to the public by having them added to the library of the Computer Science Department.

Signature:

Chitra Singh

Abstract

Dense reconstruction of 3D geometry from 2D images is an ill-posed problem. When considering the reconstruction of specific 3D objects like faces from 2D images, prior knowledge of natural facial geometry and color can be exploited to regularize this problem. A 3D morphable face model (3DMM), constructed using multiple 3D face scans, is a widely used statistical model whose basis spans variations in natural facial geometry and color. This is used to regularize face reconstruction tasks from 2D images. One of the downsides of this model is that it does not explicitly represent eyes, which should have separate parametric control.

We design a method which can reconstruct and track facial geometry and appearance, along with two separate eye-ball geometries just from a sequence of monocular RGB images. We first use a close to frontal eye-gaze image of the person as a calibration image in order to determine correct placement of the eye-balls. We can then rotate the eye-balls to follow the gaze of the person by tracking other frames from the sequence of images. Changing face pose and expressions are also captured on the tracked sequence.

The reconstruction is done using an analysis-by-synthesis approach, wherein 3D face and eye models are used for synthesizing a face image close to the observed input monocular RGB image. From this, we recover the model parameters which best match both the images. This is posed as a non-convex problem that is solved iteratively by introducing the unknowns successively while going from an easy to a more difficult problem. Finally, we show qualitative and quantitative results of the reconstructed face and eye-balls.

Acknowledgments

First and foremost I would like to thank my supervisor Professor Dr. Christian Theobalt for giving me an opportunity to work on my thesis and his support throughout the thesis and beyond. The time I spent with his research group *Graphics, Vision and Video* is unforgettable.

I also extend my deep gratitude towards Prof. Dr. Andreas Bulling, whose domain knowledge in the field of eye-gaze estimation helped in developing ideas and some concrete evaluations.

I am grateful to have worked very closely with Florian and Ayush, where we spent time regularly in our weekly meetings developing the method and evaluations. They were easily approachable if I got stuck in between, and provided the required assistance. I also greatly appreciate their comments on my writing, which took proper shape with their invaluable feedback.

I would also like to thank my friends and colleagues Edgar, Franzika, Jiayi, Mayank and Shadi for proofreading my thesis.

Finally I would like to thank my parents Sunita and Kamalesh, my sister Kanchan, my boyfriend Mayank, my flatmates Antonia and Leonie who have been my constant support system throughout, and encouraged me to push through at all times.

This accomplishment would not have been possible without the encouragement from all the people mentioned above.

Contents

List of Figures

List of Abbreviations

2D	Two dimensional.
3D	Three dimensional.
3DMM	3D Morphaple Model.
4D	Four dimensional.
AAM	Active Appearance Model.
ASM	Active Shape Model.
AUC	Area Under the Curve.
BFM	Basel Face Model.
BRDF	Bidirectional Reflectance Distribution Function.
CED	Cumulative Error Distribution.
CLM	Constrained Local Model.
CNN	Convolutional Neural Network.
CPU	Central Processing Unit.
DEFM	Deformable Eye Face Model.
DFM	Deformable Face Model.
FAM	Flexible Appearance Model.
GD	Gradient Descent.
GN	Gauss-Newton.
GPU	Graphics Processing Unit.
ICP	Iterative Closest Point.
LM	Levenberg-Marquardt.
MAP	Maximum A Posterior.
MCMC	Markov Chain Monte Carlo.
NDC	Normalized Device Coordinates.
PCA	Principal Component Analysis.
PoR	Point of Regard.

RGB	Red Green Blue.
SfM	Stucture from Motion.
SH	Spherical Harmonics.

Chapter 1

Introduction

Perception studies (Smilek *et al.*, 2006) have shown that humans look at the eye region more than any other part of the face to recognize a person. This is probably because making eye contact is one of the key aspects of effective communication. Hence, it is important to investigate into eyes while reconstructing faces, which have been ignored by a vast majority of the methods which do the same (Blanz & Vetter, 1999; Garrido *et al.*, 2013; Tewari *et al.*, 2017). Eye-gaze tracking on the other hand, is a widely researched topic (Kar & Corcoran, 2017).

We develop an approach to reconstruct 3D faces from monocular RGB video or image sequences. This face reconstruction is done with an emphasis on the eyes which act as independent eye-balls placed behind the eye-lids. The steps are two-fold; firstly, a calibration step is used for positioning the eye-balls behind the reconstructed face mesh utilizing a close to frontal eye-gaze image. Secondly, this is followed by a tracking step on the rest of the images from the image sequence. Facial pose and expressions are tracked as well as the are eye-balls rotated to follow the eye-gaze of a person.

1.1 Motivation

Very recently, social media platforms like Snapchat, Facebook, and others have brought virtual make-up and even personalized facial avatars (Pinscreen, 2018) to the common public. This has been made possible by reconstructing 3D facial geometry directly by using input images from normal phone cameras, in unconstrained settings. Placing eye-balls consistently behind the reconstructed face would enhance the experience. This reconstructed geometry is also an important input for many other applications like face recognition, facial expression transfer for puppetry, and

facial reenactment (Cao *et al.*, 2014; Kim *et al.*, 2018a).

While most methods focus on capturing the facial skin as a single mesh, they often ignore the eyes or treat them as a textured part of the skin near the eye region. In reality, human eyes are placed behind the eye-lids in orbits, the two sockets in the skull, which allows them to rotate and explore a scene.

Recently there have been some developments wherein methods focus on 3D eye-gaze estimation by fitting a morphable eye region model to an image (Wood *et al.*, 2016a), and even facial geometry reconstruction along with 3D eye-gaze capture (Wang *et al.*, 2016) using a sequences of RGB monocular images.

Reconstructions by Wood *et al.* (2016a) lack full face geometry, and are also not very expressive since their morphable model was created only using 20 face scans. We want to utilize existing full face morphable models, and *couple* eye-balls during reconstruction.

On the other hand, the method of Wang *et al.* (2016) captures full face geometry, and places eye-balls behind the eye-lids. However, their eye-ball rotations are based on a Maximum A Posterior (MAP) framework, where the most probable eye-gaze state in each frame is estimated via sampling. This is inherently biased to the observed eye-gaze directions. In contrast, we want to model anatomical rotations for both eyes, such that the eye-balls rotate in a natural manner, based on the oculomotor system of the human eye.

1.2 Research goals

Using a sequence of RGB input images from a single camera, which is also termed as *monocular* images, we set to achieve the following:

- Reconstruct the facial geometry and appearance.
- Fit two eye-balls behind the eye-lids of the reconstructed face.
- Track the facial pose and expressions, along with eye-balls such that they follow certain established laws on natural eye orientations.

1.3 Overview

This thesis is set out to explore an *analysis-by-synthesis* approach, where 3D face and eye models are used for synthesizing a face image close to the observed input monocular RGB image, thereby *inversely rendering* to recover the model parameters

which best match both the images. The goal is to reconstruct the facial geometry and appearance along with the eye-balls placed behind the eye-lids. Hence, in a constrained setting of monocular images which also have depth ambiguities, a prior knowledge of natural facial geometry and color in the form of face and eye models is exploited to regularize the problem of recovering 3D geometry and appearance from images.

A non-linear and a non-convex optimization problem is posed in terms of model parameters. Detected 2D face landmarks and pupil positions on the image are used to better constrain the problem such that it can avoid local minima. It is then solved by introducing the unknown variables successively, while going from an easy to a more difficult problem.

The face and eye model parameters are recovered along with certain scene parameters like camera pose and lighting. These include a rigid face pose as well as non-rigid facial deformations to recover face shape and color, which form the identity of the person, along with facial expressions. Additionally eye-ball centers and size, along with eye-rotations are recovered. We also retrieve the eye-gaze directions implicitly from the reconstructed eye orientations.

1.3.1 Contribution

The main contribution of this thesis is the formulation of the optimization problem in terms of the model parameters of already existing 3DMMs and eye-models, employed *together* for face reconstruction.

A *coupling* term is introduced to bring the eye-balls and the face mesh together, such that the eye-balls fit behind the opening between the eye-lids. Using an image of a person with a frontal eye-gaze direction, we fix the eye-ball center positions, thereafter eye-gaze is tracked on a sequence of images of the same person. We introduce a way to model natural eye orientations in the optimization problem, using Listing's law (Helmholtz, 1867), explained further in Chapter 2.

1.3.2 Challenges and scope

Monocular 3D reconstruction is accompanied by depth ambiguities where the perceived size of an object can either be increased by scaling or by just bringing object closer to the camera.

While the facial skin is mostly matte-like, strong point lights and sweat can make it shiny and specular. However, we assume the skin to have a matte-like or *Lambertian* material for simplicity.

Wood *et al.* (2016a) make use of a learned eye-region morphable model, which has a learned coupled eye-ball around the eye-region, and hence they only have to solve for eye rotations to recover eye-model parameters. On the other hand, we make use of an existing morphable model, and couple the eye-balls jointly during optimization. This requires an extra step of using a frontal eye-gaze image to estimate the eye-ball positions to couple them to the face.

1.4 Structure of the thesis

We introduced this thesis by motivating the reader with some potential applications and answered the *what* and the *why* for the title *Combined 3D eye and face reconstruction using monocular RGB images*.

In Chapter 2, we start with a background on fundamental computer graphics and vision concepts, and relevant optimization algorithms. This is followed by an involved discussion on the human eye and its movements, which is fundamental to the thesis.

We then refer to the related work in Chapter 3 while describing their limitations in our context. This first covers various face and eye models created until now and then introduces various methods that have been proposed for face and eye reconstruction, some of which make use of the models described in order to establish the state-of-the-art.

The method developed is then elaborated in Chapter 4, where the energy formulations which form the optimization problem are described. It first introduces all the model parameters for which the problem is solved. This is followed by the exact formulations and optimization strategy for face and eye reconstruction.

Finally, in Chapter 5 we show qualitative and quantitative results for the facial geometry, appearance and eye-gaze. Chapter 6 concludes with closing remarks on the method proposed and the results obtained, as well as major opportunities for future work on this problem.

Chapter 2

Background

This thesis combines elements from the computer graphics and vision domains, and hence some prerequisite background is introduced. It starts with an introduction to the fundamentals of 3D geometry to show how transformations are represented linearly using matrices. These transformations are then used to render 3D models as synthetic images, hence the image formation process is introduced. It guides the reader through the various elements of the graphics pipeline, the camera model which captures 3D geometry as 2D images, material model of the 3D object, and the illumination model used to approximate scene lighting. Furthermore, the optimization algorithms used to solve our formulations in Chapter 4 are discussed. This is followed by an involved discussion on the human eye and natural eye orientations assumed by it, since it is fundamental to this thesis.

2.1 Fundamentals of 3D geometry

Polygonal meshes are a set of 3D vertices joined by edges to form polygons like triangles or quadrilaterals. A mesh represents the surface of an object. For further discussion, we assume that a 3D object is represented by a polygonal mesh, which contains a set of vertices and connected edges. Certain transformations for manipulating the meshes which are used in this thesis are discussed below.

A rigid 3D coordinate transformation is represented by a set of rotations $R \in SO(3)$ (see Appendix A) and translations $\mathbf{t} \in \mathbb{R}^3$, which can be written as a homogeneous transformation matrix $T \in SE(3)$.

$$T = \begin{bmatrix} R_{3\times3} & \mathbf{t}_{3\times1} \\ 0 & 1 \end{bmatrix}; T^{-1} = \begin{bmatrix} R^T_{3\times3} & -R^T_{3\times3}\mathbf{t}_{3\times1} \\ 0 & 1 \end{bmatrix} \tag{2.1}$$

A convenient property of such a transformation matrix is its simple to compute inverse $T^{-1} \in SE(3)$.

A homogeneous transformation allows to apply a general affine transformation in 3D based on a 4D linear transformation. If a 3D point is represented as $\mathbf{X} = (x, y, z) \in \mathbb{R}^3$, its homogeneous coordinates are represented as $\bar{\mathbf{X}} = (x, y, z, 1) \in \mathbb{R}^4$, while its 3D vector can be written as $\bar{\mathbf{X}} = (x, y, z, 0) \in \mathbb{R}^4$ as it is translation invariant. For instance, rotations $\text{rot}(R)$, translations $\text{trans}(\mathbf{t})$ and scaling with a scalar $\text{scale}(s)$ can be written as linear mappings in form of 4×4 matrices:

$$\text{rot}(R) = \begin{bmatrix} R_{3\times3} & 0 \\ 0 & 1 \end{bmatrix}; \text{trans}(\mathbf{t}) = \begin{bmatrix} I_{3\times3} & \mathbf{t}_{3\times1} \\ 0 & 1 \end{bmatrix}; \text{scale}(s) = \begin{bmatrix} sI_{3\times3} & 0 \\ 0 & 1 \end{bmatrix} \tag{2.2}$$

Since these transformations are not commutative, for cumulating them, $\text{trans}(\mathbf{t}) * \text{rot}(R) * \text{scale}(s)$ is followed to achieve

$$\begin{bmatrix} sI_{3\times3}R_{3\times3} & \mathbf{t}_{3\times1} \\ 0 & 1 \end{bmatrix}$$

Matrices for translation and perspective transformations (discussed in Section 2.2.2) can only be applied to homogeneous coordinates. In the next section, we describe the image formation process using the described transformations.

2.2 Image formation process

Since our method synthesizes face images from 3D geometry, we describe the process for rendering the same. The graphics pipeline describes the various stages of the rendering process, which makes use of a camera, object material, and illumination models explained further.

2.2.1 Graphics pipeline

Multiple 3D objects form a scene, whose 2D projections onto a particular view is finally rendered. We describe this process step-by-step:

1. Model transform — Multiple 3D objects, whose vertices are in their own model space can be brought to one general world coordinate system by a simple rigid transformation of each object.

2. Viewing transform — When a camera is placed in this scene, another rigid coordinate transform is applied such that all the object coordinates are camera

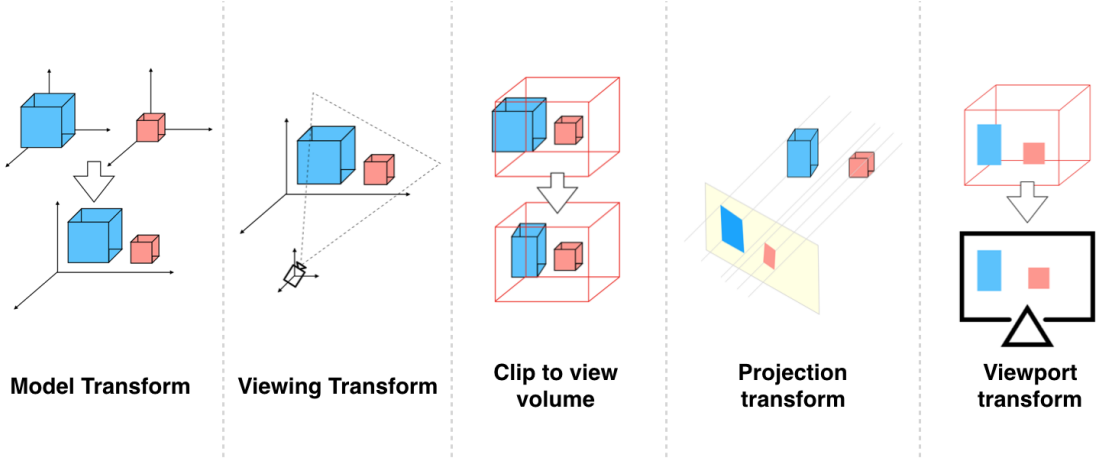


Figure 2.1: Image formation process in the graphics pipeline is shown, where the pipeline goes from left to right.

centric. Oftentimes, the camera is placed at the origin of the world coordinate system, and hence this transform is represented by an identity.

3. Clip to view volume transform — In order to limit the infinite camera view, a viewing volume is defined around the camera, and the object geometry outside of this view is excluded.

4. Projection transform — The 3D camera coordinates are converted into normalized device coordinates (NDCs), see Figure 2.3, such that the viewing area is converted into a cubic volume, to capture the process of projection. This is explained in detail in the next section.

5. Viewport transform — NDCs are finally converted into 2D screen coordinates to be rendered on a screen or saved an an image.

2.2.2 Camera models

This section describes the camera model which captures the process of converting 3D coordinates to 2D image coordinates. A camera maps 3D object(s) to a 2D image plane $Z = f$, where f is the focal length, and this mapping is termed as *projection*.

A pin-hole camera model is the simplest camera model which mimics how a real-world camera captures images. It can be seen as a box with a small hole C on one side, which allows light rays to enter and form an image on the image plane of the other side. An optical axis perpendicular to the image plane, intersects at a point called principal point $(p_x, p_y)^\top$. A point $(X_c, Y_c, Z_c)^\top$ in the camera coordinate

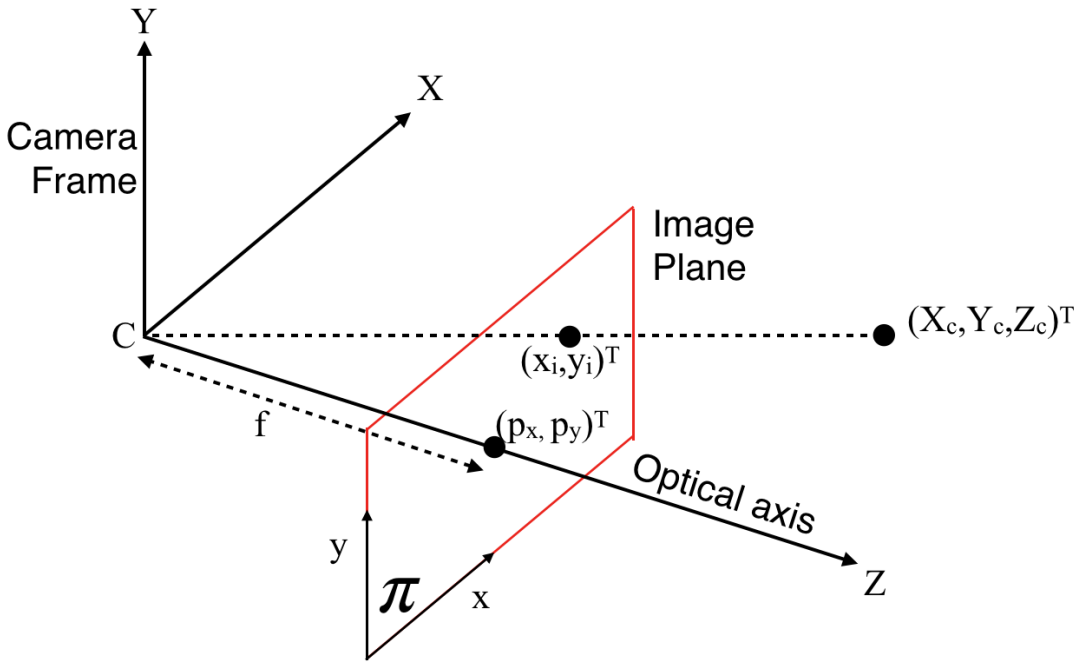


Figure 2.2: A pinhole camera model with a virtual image plane. (Figure adapted from Trucco & Verri (1998))

system maps to a 2D point $(x_i, y_i)^\top \coloneqq (fX_c/Z_c + p_x, fY_c/Z_C + p_y)^\top$ on the image plane, as seen in the Figure 2.2. Such a projection is called *perspective* projection, which depends on the depth Z of the object i.e., if the object is further from the camera, it would look smaller in the image.

Parallel projection called *orthographic* projection, is a special case of perspective projection, where the distance from the center of the projection to the image plane is considered to be infinite, and hence the depth of the object is simply ignored. In this case, the 2D point can simply be written as $(x_i, y_i)^\top \coloneqq (X_c, Y_c)^\top$. Orthographic projection can be seen in the 4th column of Figure 2.1. Although this simple model can be used for rendering, most real world cameras act as pin-hole perspective cameras, which is used in the rest of the thesis.

The focal length f is specific to the perspective camera, which has to be calibrated, along with other parameters which form the camera intrinsics. A standard camera calibration matrix K as formulated by Hartley & Zisserman (2000), is used to convert from camera coordinate system to image pixels position on the image

plane.

$$K = \begin{bmatrix} f\ h_v & -h_u\ \cot(\zeta) & p_x \\ 0 & f\ h_v / \sin(\zeta) & p_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.3}$$

Here, ζ is responsible for accounting for a skew caused in the image plane, introduced by manufacturing defects in camera-sensing element, which is almost always 90° for no skew. h_v and h_u denote pixel dimensions in image plane (Whitehead & Roth, 2004). When the camera intrinsics are not available, like Whitehead & Roth (2004) we also consider center of projection to be the same as the center of the image, provided that the image is not cropped.

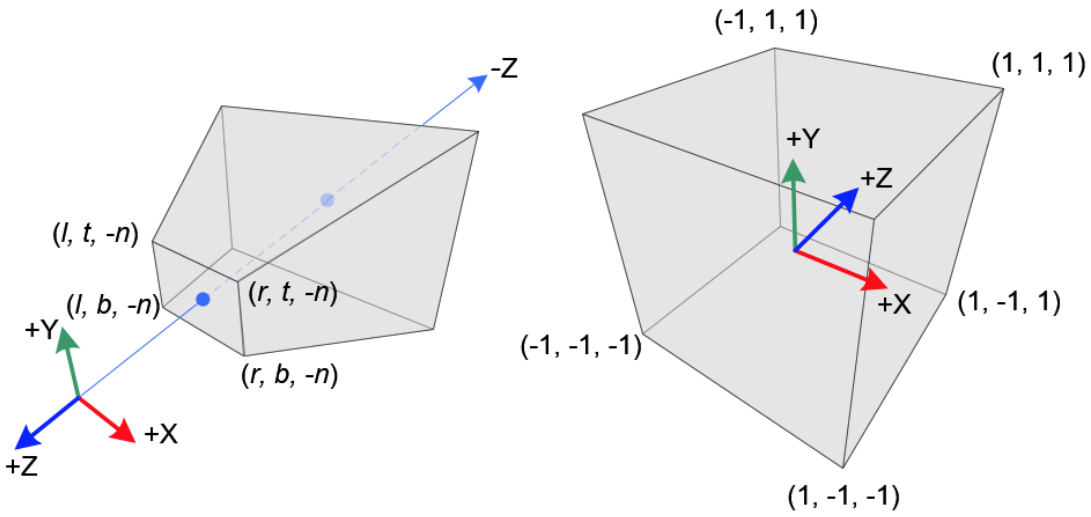


Figure 2.3: The projection matrix π maps the perspective viewing frustum (left) to a unit cube (right) which represents NDCs. (Image taken from Ahn (2008))

A projection matrix π captures all the mappings defined above in a matrix. It is defined over homogeneous 3D coordinates which transforms 3D points in camera space to NDCs. The perspective frustum, as seen in Figure 2.3 can be imagined as the view of a pin-hole camera, confined to a pyramid-like frustum, defined by $Z =$ near and $Z =$ far planes, which confine the 3D object(s) in consideration. Using π, we map a point in this frustum to a unit cube. This mapping $[0, w] \mapsto [-1, 1]$, $[0, h] \mapsto [-1, 1]$, $[\text{near}, \text{far}] \mapsto [-1, 1]$ is seen in Figure 2.3. Here image width is $w = r - l$ and image height is $h = t - b$. Note that near plane is -near, as the NDCs

are denoted by a left handed system in OpenGL[1].

$$\pi = \begin{bmatrix} 2f\ h_v/w & 0 & (2p_x/w) - 1 & 0 \\ 0 & 2f\ h_v/h & (2p_y/h) - 1 & 0 \\ 0 & 0 & -(\text{near} + \text{far})/(\text{far} - \text{near}) & -2 \cdot \text{near} \cdot \text{far}/(\text{far} - \text{near}) \\ 0 & 0 & -1 & 0 \end{bmatrix} \tag{2.4}$$

The above defined π captures the camera intrinsics and the perspective projection. Camera extrinsics describe the position and orientation of the camera with respect to the world coordinate system. An camera extrinsic matrix $\phi_{R,\mathbf{t}} \in SE(3)$ thereby maps object positions $(X_w, Y_w, Z_w)^\top$ in the world coordinate system to camera coordinate system $(X_c, Y_c, Z_c)^\top$, captured by a rotation $R \in SO(3)$ and translation $\mathbf{t} \in R^3$, $(X_w, Y_w, Z_w, 1)^\top = \phi_{R,\mathbf{t}}(X_c, Y_c, Z_c, 1)^\top$. Henceforth, the transformation from 3D world space coordinates to 2D image coordinates can be written as:

$$\begin{aligned} (x, y, z, w)^\top &= \pi\ \phi_{R,\mathbf{t}}\ (X_w, Y_w, Z_w, 1)^\top \\ (x_i, y_i)^\top &\coloneqq (x/w, y/w)^\top \end{aligned} \tag{2.5}$$

2.2.3 Material model

We want to model the material of the face and the eye to render them realistically. These object materials are modeled using a bidirectional reflectance distribution function (BRDF) (Cook & Torrance, 1982), a 4D function which describes how light reflects at a point on a 3D object. It describes if an object would look matte, blur, shiny, or mirror-like when it interacts with light from the surroundings.

For example, perfectly matte (or diffuse) surfaces are described using a *Lambertian* model which represents a constant BRDF. A matte-like surface has diffuse reflection, where the incident light falling on it is reflected equally in all directions. Other models like Phong and Bling-Phong describe mirror-like specular materials. For the sake of simplicity, in this thesis we assume that our model materials are Lambertian, represented by a constant *albedo*, a measure of diffuse reflection of an object.

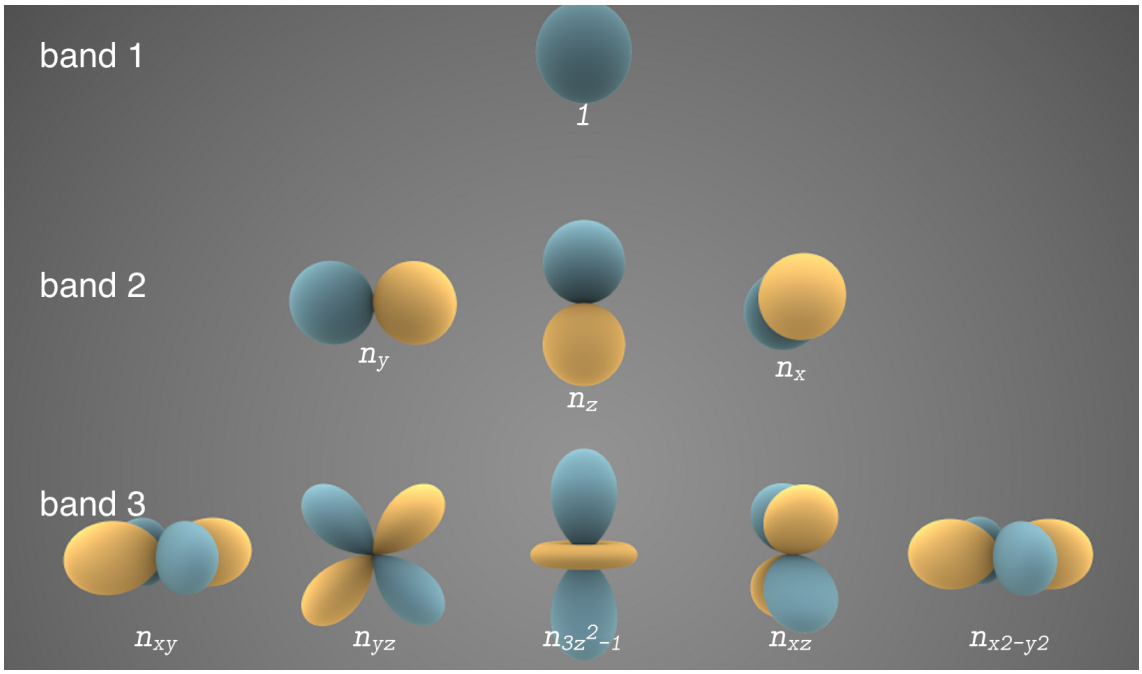


Figure 2.4: Visual representation of nine orthonormal SH basis functions in the first three bands. In this visualization, a unit sphere is distorted by scaling each point radially by the value of the basis function at that point, colored by the sign where blue represents positive, while yellow represents negative values. (Image taken from Wikipedia (2014))

2.2.4 Illumination model

Along with a material model for objects, scene lighting also affects their appearance, and hence a need for an illumination model arises. We make use of spherical harmonics (SHs), proposed by Basri & Jacobs (2003) and Ramamoorthi (2006). SHs represent distant illumination of Lambertian surfaces. This illumination model assumes that the 3D model's surface is convex, and hence intereflections and cast shadows are ignored and its surface orientation can be parametrized by surface normals $\mathbf{n}$. A linear lighting model can be described using the first three bands of SHs basis functions $H_b : \mathbb{R}^3 \mapsto \mathbb{R}$, as seen in Figure 2.4. For the first three bands, $B = 3$, illumination L_c at vertex $\mathbf{v}$, with surface normal $\mathbf{n}$, for color channel $c \in \{R, G, B\}$ can be written as:

$$L_c(\mathbf{n}, \boldsymbol{\kappa}) = \sum_{b=1}^{B^2} \boldsymbol{\kappa}_b H_b(\mathbf{n}) \tag{2.6}$$

[1]The open graphics library we use for rendering 2D and 3D vector graphics.

which can be further expressed in terms of $B^2 = 9$ coefficients $\boldsymbol{\kappa} \in \mathbb{R}^9$ for each color channel c as:

$$\begin{aligned} L_c([n_x, n_y, n_z]^\top, \boldsymbol{\kappa}) = \kappa_1 \text{ (band 1)} \\ + \kappa_2 n_y + \kappa_3 n_z + \kappa_4 n_x \quad \text{(band 2)} \\ + \kappa_5 n_x n_y + \kappa_6 n_y n_z + \kappa_7 (3n_z^2 - 1) \\ + \kappa_8 n_x n_z + \kappa_9 (n_x^2 - n_y^2) \quad \text{(band 3)} \end{aligned} \tag{2.7}$$

While we can render our face and eye models with the described image formation process, our formulations involve variable parameters of these models to be optimized for, such that the rendered images are close to the input images. Therefore, we discuss relevant optimization strategies in the next section.

2.3 Optimization

In this thesis, we aim to find a set of model parameters by posing an optimization problem as an energy function, thereby minimizing this energy (or cost) function E to find optimal values $\mathbf{x}^*$ for the parameters. This is posed as an unconstrained multi-variate non-linear least-squares problem (Bjõrck, 1996) as:

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} E(\mathbf{x}) = \arg\min_{\mathbf{x}} \sum_{j=1}^{m} (r_j(\mathbf{x}))^2 = \arg\min_{\mathbf{x}} \|\mathbf{r}(\mathbf{x})\|_2^2 \tag{2.8}$$

This energy function E comprises of non-linear scalar functions called residuals $\mathbf{r} = [r_1, r_2, ..., r_m]^\top$ (with $m > n$), which take variables $\mathbf{x} = [x_1, x_2, ..., x_n]^\top$ as input. Since each residual is posed as a non-linear and a non-convex function, the entire energy is also non-convex.

The least-squares(L_2) error comes from the assumption that the residuals are Gaussian distributed. The presence of outliers strongly impacts the solution, given the above assumption. Therefore, robust error norms $\rho(\mathbf{r})$ often replace the above L_2 norms $\|\mathbf{r}\|_2$. However, in this thesis we have not used them.

To solve such a minimization problem, we use iterative methods, which repeatedly modify an estimate from an initial guess. These small changes can be determined using the gradient information of the variable parameters. These methods linearize the residuals at first, by using first-order Taylor expansion for a small change $\mathbf{h}$ in $\mathbf{x}$:

$$\mathbf{r}(\mathbf{x} + \mathbf{h}) = \mathbf{r}(\mathbf{x}) + \mathbf{J}_r \mathbf{h} + \mathcal{O}(\|\mathbf{h}\|^2) \tag{2.9}$$

Here $\mathbf{J}_r$ denotes the Jacobian matrix made up of first order partial derivatives of each residual $\mathbf{r}$ w.r.t. $\mathbf{x}$, see Appendix A.2. The estimated small change for the first iteration $\mathbf{h}_1$ is then used to update the initial estimate $\mathbf{x_0}$ as $\mathbf{x_1} = \mathbf{x_0} + \mathbf{h}_1$ to compute the function and gradient values for the next iteration and so on.

In this thesis, we make use of Levenberg-Marquardt (LM), an iterative method to solve for E. Since LM is a combination of two iterative algorithms: gradient descent (GD) and Gauss-Newton (GN), we describe them in the next sections.

2.3.1 Gradient descent

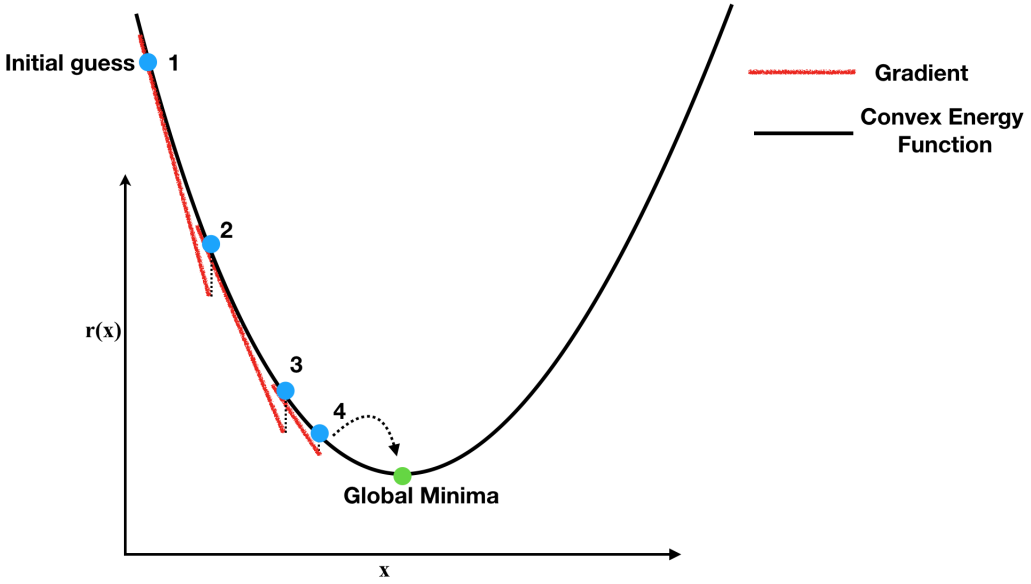


Figure 2.5: Gradient descent in a 1D convex landscape

Gradient descent (Figure 2.5) is a simple first order line-search algorithm, which takes a step in the negative gradient direction, also known as the direction of steepest descent, at a given point. The idea is to iteratively move in the direction where the energy reduces locally at each iteration. The small change $\mathbf{h}_i$ for parameter value $\mathbf{x}_{i-1}$ for i^{th} iteration can be written as:

$$\mathbf{h}_i = -\mathbf{J}_r^\top \mathbf{r}(\mathbf{x}_{i-1}) \tag{2.10}$$

2.3.2 Gauss-Newton optimization

This method approximates second-order (curvature) information, and thereby uses a more direct path to reach a local minima iteratively, by assuming that E is locally quadratic. Using the linear approximation of a non-linear residual from Equation 2.9, the energy E can be posed as

$$\begin{aligned} E(\mathbf{x}_{i-1} + \mathbf{h}_i) &= \|\mathbf{r}(\mathbf{x}_{i-1} + \mathbf{h}_i)\|_2^2 \\ &\approx \|\mathbf{r}(\mathbf{x}_{i-1}) + \mathbf{J}_r \mathbf{h}_i\|_2^2 \\ &= \mathbf{r}(\mathbf{x}_{i-1})^\top \mathbf{r}(\mathbf{x}_{i-1}) + 2\mathbf{h}_i^\top \mathbf{J}_r^\top \mathbf{r}(\mathbf{x}_{i-1}) + \mathbf{h}_i^\top \mathbf{J}_r^\top \mathbf{J}_r \mathbf{h}_i \\ &= E(\mathbf{x}_i) \end{aligned} \tag{2.11}$$

This results in a quadratic equation in $\mathbf{h}_i$, which can be seen as the increment which has to be solved for at every iteration i. To solve these equations, the first derivative of E is set to zero, which leads to

$$\mathbf{h}_i = -(\mathbf{J}_r^\top \mathbf{J}_r)^{-1} \mathbf{J}_r^\top \mathbf{r}(\mathbf{x}_{i-1}) \tag{2.12}$$

Thus the increment $\mathbf{h}_i$ is added to the evaluation point $\mathbf{x}_{i-1}$ from the previous iteration, and the process is repeated until convergence or a stable solution.

A good initial guess $\mathbf{x}_0$ is required for this method to reach a good local minima of a non-convex energy landscape, see Figure 2.6. Even though convergence is not always guaranteed even at local minima, under certain conditions (Chen & Li, 2005) convergence to a local minima is quadratic. Hence this is a faster method than simple gradient descent.

2.3.3 Levenberg-Marquardt optimization

This algorithm uses a mix of both Gauss-Newton and gradient descent steps using an adaptive dampening parameter to interpolate between the two. It is a trust-region method which approximates a model function over a subset of the search space, by comparing actual energy reduction versus predicted energy reduction and then expands or contracts the trust region via the above-mentioned dampening parameter λ. The increments are computed via:

$$\mathbf{h}_i = -(\mathbf{J}_r^\top \mathbf{J}_r + \lambda \, diag(\mathbf{J}_r^\top \mathbf{J}_r))^{-1} \mathbf{J}_r^\top \mathbf{r}(\mathbf{x}_{i-1}) \tag{2.13}$$

Here, smaller λ implies Gauss-Newton, while larger λ implies a gradient descent step, with step size $1/\lambda$.

The method also allows for non-monotonic steps, which essentially relaxes the

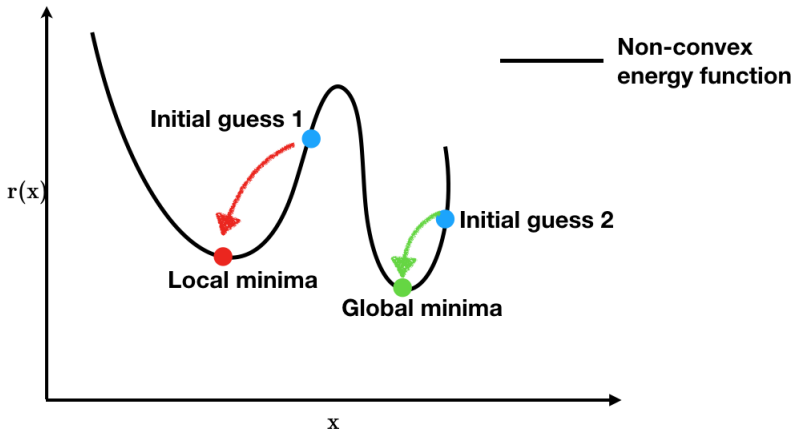


Figure 2.6: Local minima in a 1D non-convex landscape. This figure illustrates that the process of finding the right local minimum depends on a good initialization.

constraint that energy should strictly decrease after every iteration, therefore allowing jumps over some local minima, and sometimes avoids getting stuck at local minima.

While it is guaranteed to converge to a global minima if the energy is convex, in the non-convex case, it can also find the local minima if initialized close to it. Like Gauss-Newton, convergence is quadratic if the energy is not too non-linear under certain conditions (Chen & Li, 2005), else the convergence is linear from gradient descent steps. Like Gauss-Newton, it also requires a good initial guess for parameter values, but allows this guess to be further away, while being more robust, as it uses gradient descent when Gauss-Newton fails.

We use the Levenberg-Marquardt algorithm in this thesis to solve our formulated problems. We start by posing a problem using a small set of residual functions and an initial estimate for a small set of parameters, solve it, and then increase the number of parameters and residual functions to increase the problem size slowly. The estimated parameter values from the previously solved simpler problem are used as initial estimates for a bigger problem, along with an initial estimate for new parameters introduced. This incremental approach reduces the complexity of minimizing a highly non-convex energy, as opposed to directly minimizing the whole energy at once.

Since we make use of a technique which computes analytical gradients for the parameters needed for the optimization, we describe it in the next section.

2.3.4 Auto-differentiation

We formulate our problems using *Ceres* (Agarwal & Mierle, 2012), a C++ optimization library which provides an automatic differentiation framework. It is a scientific computing technique, which evaluates generic analytical derivatives of continuous composite mathematical functions using the chain rule $f(g(x))' = f'(g(x))g'(x)$. As we were prototyping and evaluating different types of energy terms, this was advantageous since it avoids tedious, error-prone, and time consuming hand-coded derivatives of energy terms.

While we describe most of our energy terms using continuous mathematical functions, our formulations also include gradients of discretized images. Since images are not continuous functions, we describe them in the next subsection, such that they can be used as functions in the auto-differentiation framework.

2.3.5 Images as functions

Let I be a 2D image, which stores discretized color values in a 2D grid. The image pixel color at discrete grid location $(u, v)^\top$ for a particular channel $c \in \{R, G, B\}$ can be represented as $I_c(u, v) : \mathbb{R}^2 \to \mathbb{R}$. Since the analytical gradient of such a discretized image cannot be calculated, it is numerically approximated with finite differences, which again makes use of a 2D first-order Taylor expansion. Image derivatives tell us about local pixel value changes (in each channel) around a pixel:

$$\begin{aligned} \frac{\partial I_c(i,j)}{\partial u} &\approx \frac{I_c(i+1,j) - I_c(i-1,j)}{2h_u} = I_{cu}(i,j) \\ \frac{\partial I_c(i,j)}{\partial v} &\approx \frac{I_c(i,j+1) - I_c(i,j-1)}{2h_v} = I_{cv}(i,j) \end{aligned} \tag{2.14}$$

Since images are represented as discrete functions, their continuous approximation for a given pixel position is calculated by their bi-linear interpolation over four nearest image pixels.

As a preprocessing step, images are also smoothed to reduce noise, and thus reduce noisy gradients. Smoothing is done by replacing a pixel by a weighted average of its neighborhood. A box kernel, Gaussian kernel, or edge preserving median and bilateral filters are used for the same.

Next, we overview the human eye shortly and then discuss the formulation for the three-dimensional orientation of the eyes and its axes of rotation based on Listing's law, which allows for natural eye-ball rotations of the eyes.

2.4 The human eye and eye orientations

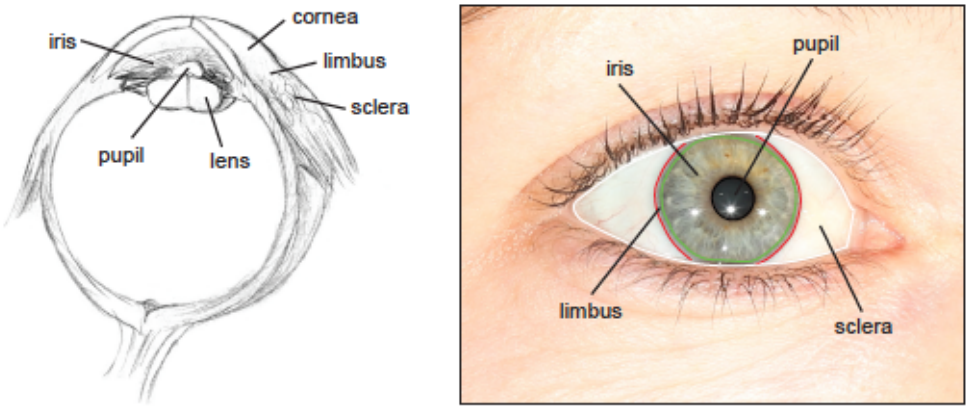


Figure 2.7: The human eye explained, left: a cross-section, right: front view. (Image from Bérard *et al.* (2016).)

The human eye, as seen in Figure 2.7 is roughly spherical which sits inside a socket called orbit in the human skull. Iris, the coloured part of the eye (which can be shades of brown, blue and green) controls the size of the pupil, and hence controls the amount of light entering. Furthermore, various eye muscles are involved in moving the each eye-ball independently as well as both eye-balls simultaneously. Since we model our eye rotations based on eye movements, we discuss them in detail here. Eye movements can be classified as follows:

2.4.1 Eye movements

Each eye-ball can move independently in each socket. Since both the eyes focus at some point in the environment, they move together such that their point of focus is the same. They have independent and coupled rotations, which are described further.

Independent eye-ball movement

Each individual eye-ball's rotation is controlled by three types of muscles, as seen in Figure 2.8. These movements about three axes are:

- Horizontal rotation (through angle H) about a head-fixed vertical axis — *abduction* and *adduction*.

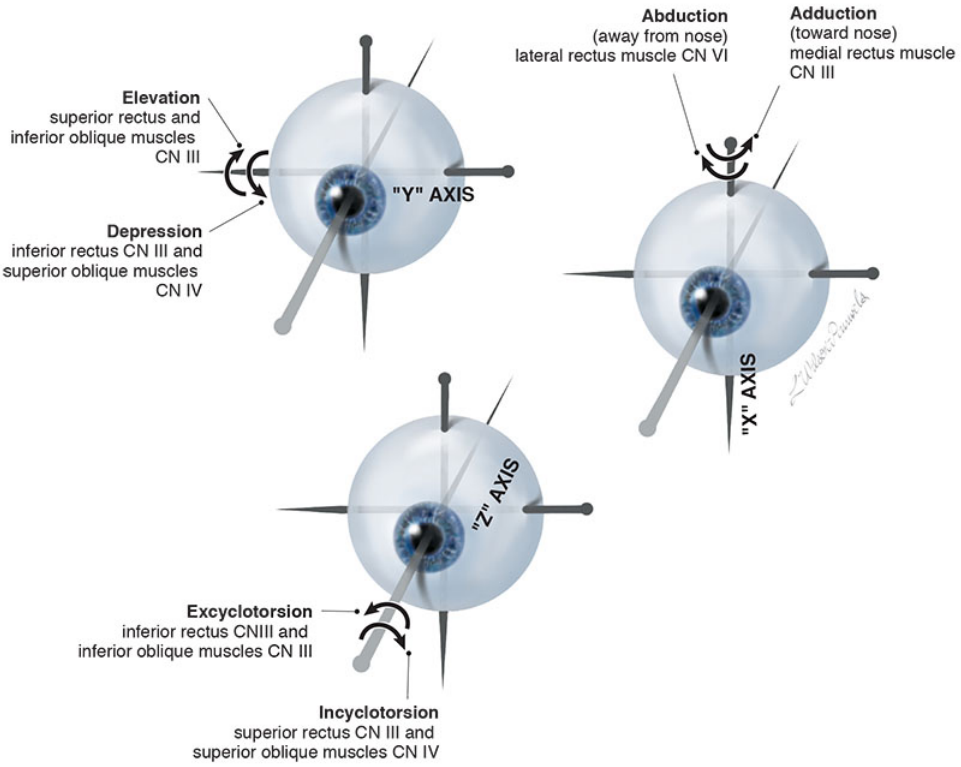


Figure 2.8: Independent eye-ball movements about X, Y, Z axes. (Image from Wilson-Pauwels *et al.* (2013))

- Vertical rotation (through angle V) about the interaural axis which connects the rotation centers of two eyes — *elevation* and *depression*.
- Torsional rotation (through angle T) about the line of sight — *excyclotorsion* and *incyclotorsion*.

Coupled eye-ball movement

While both the left and the right eyes have independent muscles for movement, they work together to explore the 3D environment to focus at a point or infinity. These conjugate movements, seen in Figure 2.9, can be classified as:

- *Version* — When the eyes move in the same direction, for example to explore an object on the left, this conjugate or symmetric movement is termed as version.
- *Vergence* — Eyes also move in opposite directions, for example, when they converge to focus on a point from infinity. This dis-conjugate or anti-symmetric movement is termed as vergence.

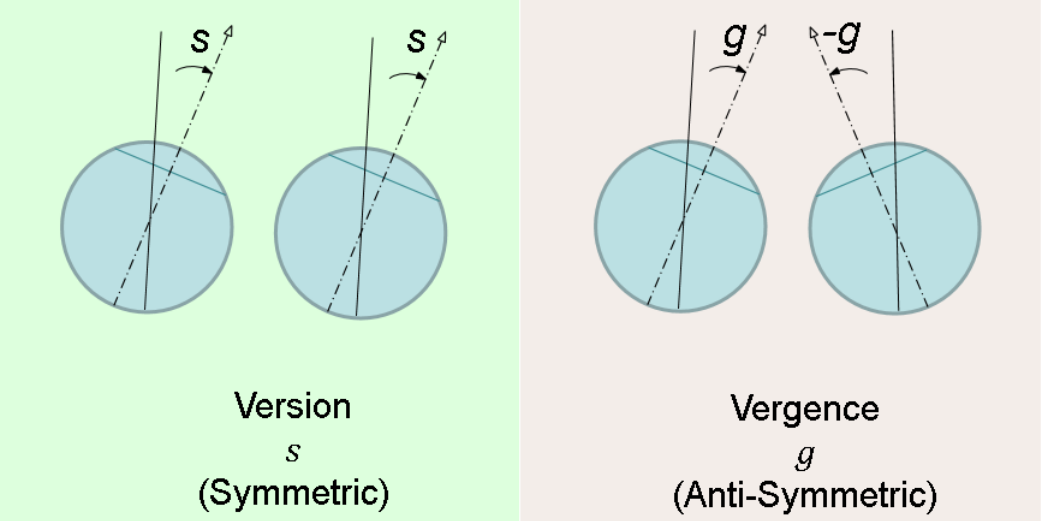


Figure 2.9: Binocular eye orientations for coupled eye-ball movement. (Figure adapted from Horner (1999))

In the next section, we describe the Listing's law which governs these movements.

2.4.2 Listing's law

Given the described eye movements, when the eye looks straight ahead, an infinite number of torsional rotations can be assumed by the eye while rotating about the line of sight, see Figure 2.10, a. However these rotations are restricted by the eye muscles and neural control, which have known to be studied since the 1800's by a famous German mathematician Johann Benedict Listing, who never produced any formal publication of his law. Later his work was tested experimentally, validated and published by Helmholtz as Listing's law (Helmholtz, 1867).

Listing's law is defined to disambiguate the infinite number of torsional rotations of the eye-ball. it states that from an initial orientation of the eye-ball, while the head is kept fixed, only the orientation which can be reached by a single rotation about an axis in the Listing's plane can be assumed by the eye (Wong, 2004). Listing's plane (as seen in Figure 2.10, b.) is orthogonal to the line of sight when the eye is in that initial position. The implication of Listing's law (Wong, 2004) is that for any combination of vertical and horizontal rotation of each eye, there exists only one torsional angle which is expressed as $T = -HV/2$ (Helmholtz, 1867).

However, this law is only valid if the eye fixates at a target at optical infinity, i.e. if the vergence is zero. This law was extended for two eyes which can focus at a point, and is explained below.

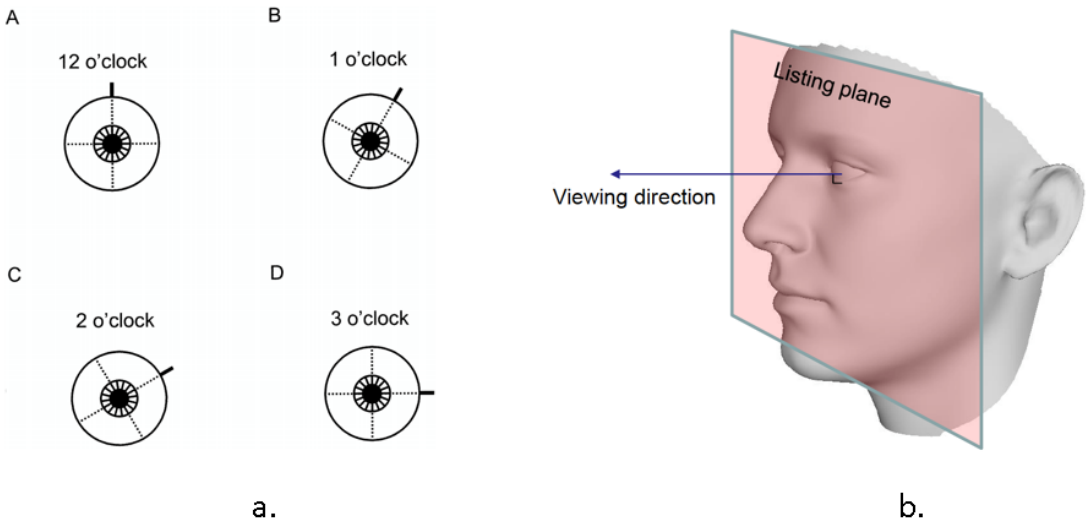


Figure 2.10: a. Different ambiguous torsional eye rotations possible when the eye looks straight ahead. b. Listing's plane. Image (a.) from Wong (2004)

Binocular extension of Listing's law

While each eye has its own set of muscles, the eye orientations for both the eyes are coupled. These coupled binocular eye orientations have been studied, and Listing's law is extended to include eye vergence for fixation of nearby targets. (Run & Berg, 1993) state that during convergence of eyes on a target, the Listing's plane is rotated anti-symmetrically by about a quarter $\nu/4$ as much as the vergence angle ν created between the two eyes in bi-polar Helmholtz coordinate system, as seen in the Figure 2.11. Using this, they described right eye rotation vector $\mathbf{r}_R$ and left eye rotation vector $\mathbf{r}_L$, the using only three angles θ, α, and nu in this Helmholtz coordinate system instead of three Euclidean angles for each eye (=6), which control both the eye orientations.

We can re-parameterize the three angles described above in the form of rotation vector (Appendix A) for each eye $\mathbf{r} = (V, H, T) \in \mathbb{R}^3$ in Euclidean space , can be decomposed into symmetrical $\mathbf{s}$ and anti-symmetrical $\mathbf{g}$ vectors. $\mathbf{s}$ being identical for both eyes, which controls the version, while $\mathbf{g}$ being equal in magnitude and opposite in direction for both eyes. Hence, the rotation vectors (as seen in Figure 2.9) can be written as $\mathbf{r}_R$ and $\mathbf{r}_L$ for the right and left eyes respectively:

$$\mathbf{r_R} = \mathbf{s} - \mathbf{g} = (\theta/2, \alpha/2 - \nu/4, -\theta\nu/4) \tag{2.15}$$

$$\mathbf{r_L} = \mathbf{s} + \mathbf{g} = (\theta/2, \alpha/2 + \nu/4, \theta\nu/4) \tag{2.16}$$

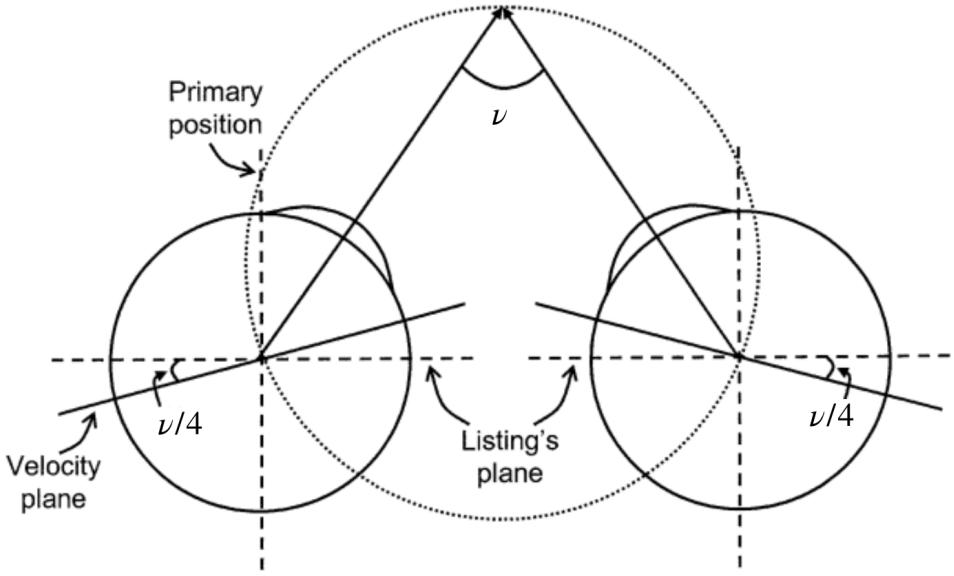


Figure 2.11: Binocular extension of Listing's Law (top view). The two eye-ball centers lie on the two poles of a bi-polar Helmholtz coordinate system. Image from Wong (2004)

where θ controls the vertical rotation for both eyes, α controls horizontal rotation and ν describes the amount of vergence, zero being fixation at optical infinity for which the original Listing's law holds true. These formulations are based on approximations on certain formulations from Taylor expansion. Therefore, this only holds for small angles ($< 30°$). We make use of these angles to formulate the eye-ball rotations in our proposed method.

2.5 Summary

Having gone through the background in detail, we make use of the transformations of 3D geometry, and Listings's law to formulate an optimization problem. Since we render images, we use OpenGL library for the image formation process. Levenberg-Marquardt optimization algorithm from Ceres C++ library is utilized for solving the same.

Chapter 3

Related Work

This chapter overviews the work done in the past relevant to the thesis. Since we make use of a face and an eye model to fit to images, various parametric face and eye models are introduced. This is followed by reconstruction methods, some of which make use of the described models. We position ourselves around the same, contrasting where we differ, while combining eye and face models to recover a 3D face mesh with eye-balls. Finally, past methods for eye-gaze tracking are explored, as our method implicitly recovers eye-gaze. We also use eye-gaze to quantify our evaluations on our eye reconstructions.

3.1 Parametric models

Parametric models are models which can be meaningfully controlled by a finite set of parameters. We describe some face and eye parametric models below:

3.1.1 Face models

Parametric face models are of great importance in the animation industry as they enable creation of new face models by simply moving some sliders. These could make the face look fatter or thinner, masculine or feminine, and even change their expression and face color. Otherwise, creating handcrafted realistic face models by artists is a tedious job.

There has been significant interest shown by computer graphics and computer vision researchers to create such parametric face models from 3D scans and images, and parameterize them based on certain properties. While some models are based on the human face anatomy, others directly learn the skin structure from a collection of 2D images or 3D scans. We discuss these two types further.

Anatomy based face models

DeCarlo *et al.* (1998) created an anthropometric face model using statistical measurements of faces, like common ratios between specific identifiable keypoints of faces called *landmarks*. Lee *et al.* (1995) proposed physics-based facial modeling, whereby they added parametric contractible muscles at anatomically correct positions over an underlying estimated skull and hinged jaw.

Quite recently, similar work on anatomical constraints was proposed by Wu *et al.* (2016), where they used an estimated skull and hinged jaw model to constrain local skin deformations on a parametric mesh, which constrained the face to valid expressions based on the bone structure below. They incrementally built a local subspace of their face model, using an iterative shape ranking strategy from 3D face scans in correspondence.

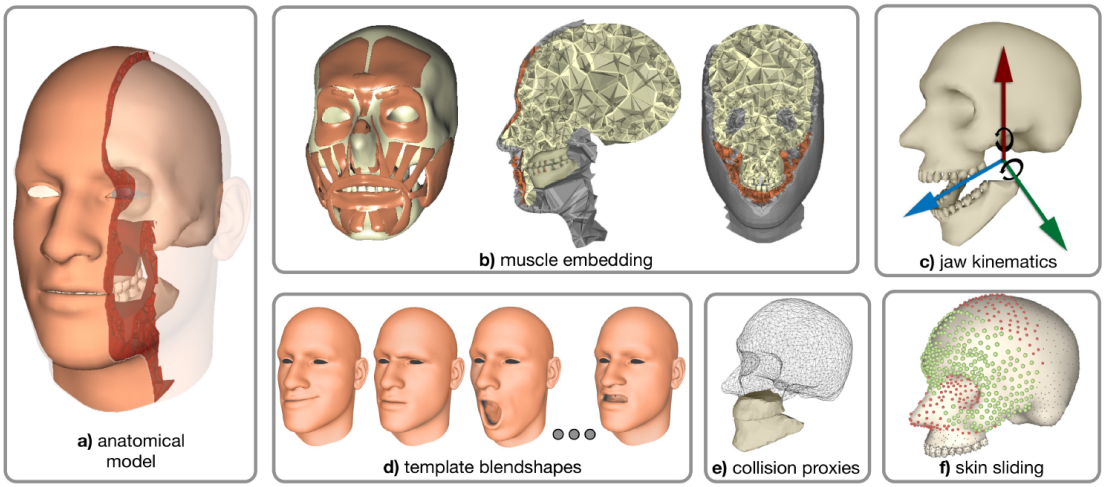


Figure 3.1: Anatomy based face model. Image from Ichim *et al.* (2017)

Ichim *et al.* (2017) also proposed a physics-based muscle activation parametric model (see Figure 3.1) using a few expression scans of a person, represented as tetrahedrized volumetric models. They did this by simulating collision and contact handling of the facial muscles with the underlying bone structure.

Face models learned from data

While the above methods used anatomy of the face, data based methods have been used to create parametric meshes to represent the face surface directly.

Active appearance models(AAMs) introduced by Edwards *et al.* (1998) capture statistical 2D face shape (active shape model, Cootes *et al.* (1995)) and face albedo

variations (flexible appearance model, Lanitis *et al.* (1995)) from labeled example images. This was a learned statistical face model, whose deformations represented characteristics of types of face models it was built on. Note that throughout this thesis, we refer to diffuse albedo as albedo, for a Lambertian material model.

Since there could be many such *example* faces, each exhibiting different types of correlated variations, Principal Component Analysis (PCA) was applied to reduce the dimensionality of the example face data variations. This created a small set of orthonormal shape and albedo basis, such that a new face was built by linear combinations of this basis.

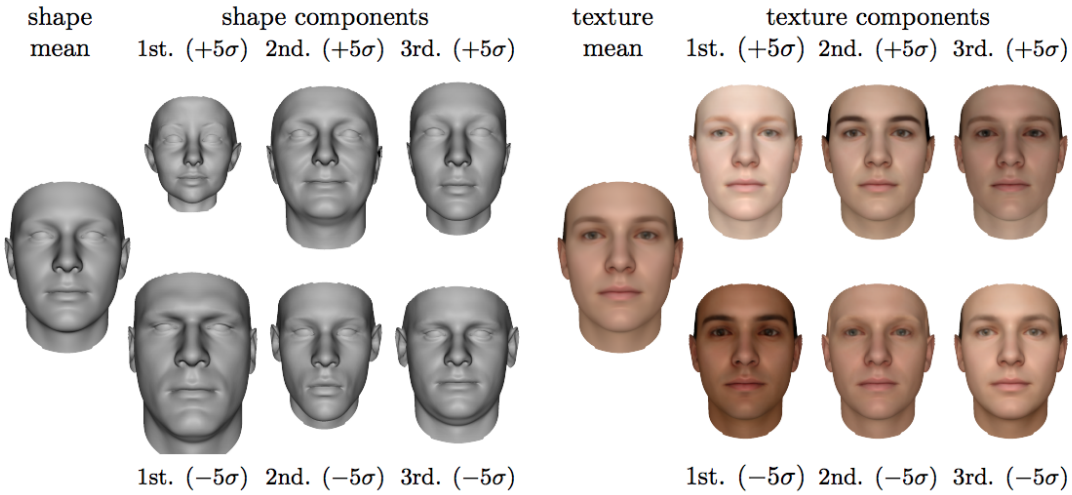


Figure 3.2: Data based 3D morphable face model. Image from Blanz & Vetter (1999)

Similar to the work proposed by Edwards *et al.* (1998), a 3D morphable model (3DMM) for the face was first introduced by Blanz & Vetter (1999), as in Figure 3.2. This was created by reconstructing many different face meshes in correspondence from 3D face scans, which were again decomposed by PCA into 3D shape and albedo basis to reduce dimensionality. This allowed for parametric non-rigid deformation of a generic 3D face model to create new identities of people.

Facial animation to control facial expressions of these face models is usually done using blendshapes (Lewis *et al.*, 2014), which are semantically meaningful additive deformations on a model with a neutral face expression. Garrido *et al.* (2013) incorporated such a personalized blendshape model for each of the 3D scanned meshes. Since many blendshapes also shared vertices, PCA was also applied to remove linear dependencies.

Generating such a semantic parameterization automatically can be achieved by transferring expressions from an existing generic face model to a target face mesh

either by deformation transfer (Sumner & Popović, 2004) or by doing non-rigid ICP (iterative closest point) for surface registration (Amberg *et al.*, 2007).

As an extension of the 3DMM, Schneider *et al.* (2018) proposed parametric and generative facial freckles model, for generation of freckle density and size distribution from large number of aligned 3D face textures.

In this thesis, we use the openly available statistical 3DMM model (Basel face model, Paysan *et al.* (2009)) which provides face shape and albedo basis, along with PCA blendshape basis created by Garrido *et al.* (2013). This model is used to constrain our space of plausible faces. However, this biases us towards European facial structure and appearance, as the 3D scans were done on them.

3.1.2 Eye models

While constructing parametric face 3DMMs, Blanz & Vetter (1999) considered eyes as a part of the same face mesh, capturing them as a part of face texture. However, eyes act as separate entities which move, and hence specific eye models are needed to represent them, which is mostly ignored when face models are created. Therefore we discuss some parametric eye models which could be used along with 3DMMs.

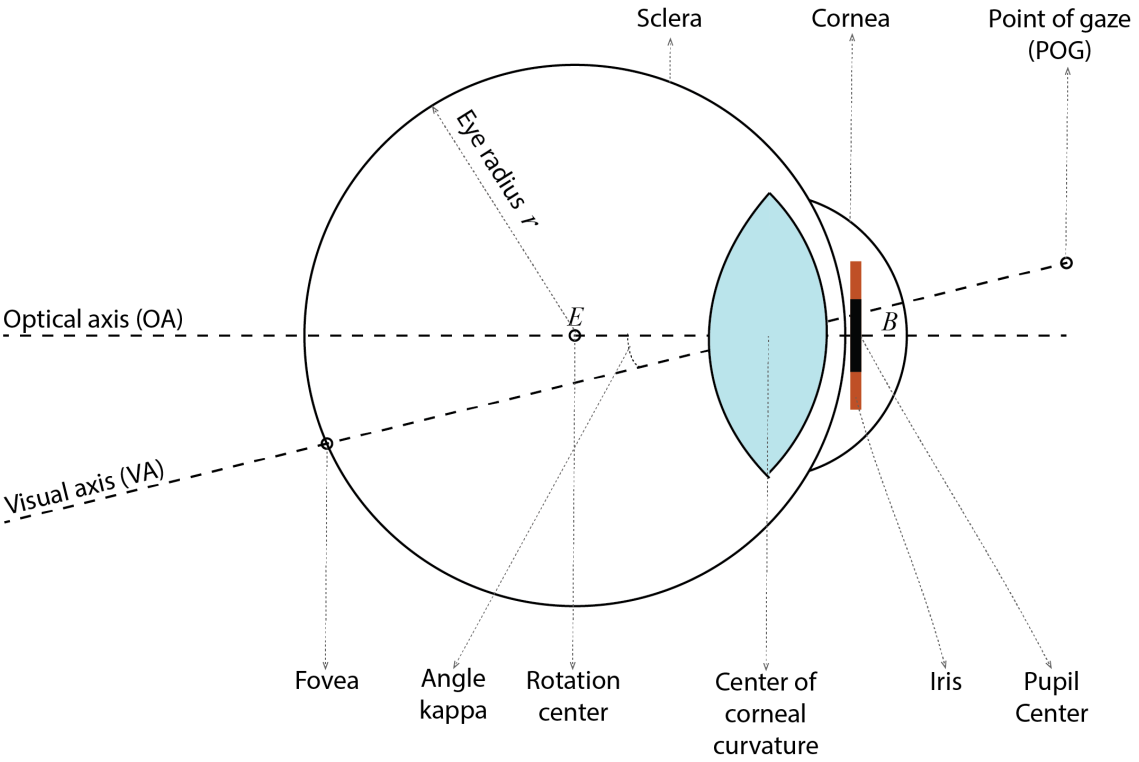


Figure 3.3: Side cross-section of the eye. Image adapted from Oyster. (1999).

An anatomical eye model is often represented by two differently sized spheres (Sagar *et al.* (1994)) as seen in Figure 3.3. The outer bigger sphere for white sclera and a smaller one for transparent cornea. While the sphere is just an approximation, various modes of variation in the spherical shape itself was captured by Bérard *et al.* (2016), as a morphable model similar to the work of Blanz & Vetter (1999). They also modeled detailed veins on the white sclera as a forest, where each vein was represented as a tree, and multiple veins emerged from other veins based on some rules. Such a detailed model was possible due to their both passive and active hardware setup with multi-view cameras.

The anterior part of the eye contains cornea, iris, pupil and lens which is represented by the smaller sphere in Figure 3.3. While the iris color, size and structure is unique for every person (Daugman (2009)), size of the pupil is controlled by the iris. This is done by planar radial motions (Bérard *et al.* (2014)) to constrict[1] and dialate[2] the pupil.

Furthermore, the eye-ball is rotated about the optical axis (OA), but the line of sight is along the visual axis (VA). According to Majid *et al.* (2013), an angle Kappa between OA and VA is about 2°-5°on an average, which is fixed for any individual. Thereby light rays are focused at Fovea, which is the central point of image focus on the VA.

Chen & Ji (2008) estimated the angle Kappa using a single camera, by making a person gaze at nine fixed points on a screen and then solved for the 3D position of the eye center, pupil and VA keeping the head fixed. We refer to this as calibration step to find person specific eye center. We ignore this angle kappa and assume that VA lies on OA, as assumed by Wen *et al.* (2017), Itti *et al.* (2004),Wood *et al.* (2016a), Wood *et al.* (2018). Using only a frontal gaze image of a person, we solve for person-specific eye-ball centers.

In computer graphics, the described eye model was rendered in a photorealistic manner by Jimenez *et al.* (2012), where light interaction effects from the buldged transparent cornea was also taken into consideration. They modeled and rendered reflections and refractions arising from transparent layers and wetness in the eyes. Wood *et al.* (2016b) used their sophisticated shading techniques for synthesizing one million eye images with multiple parameter combinations of the described eye model, to train for eye-gaze estimation.

However we use a simplified eye-model. Similar to Itti *et al.* (2004), Weissenfeld *et al.* (2009) Sagar *et al.* (1994), Wood *et al.* (2016a), Wood *et al.* (2018) we assume

[1]Narrowing of the pupil.
[2]Widening of the pupil.

that the eyeball is a sphere, which is flattened in the front. The iris and pupil lie on the flat area and their size can be controlled by shrinking or expanding radially on the flat surface. We use Wood *et al.* (2016a)'s linear mixture of aligned high resolution photos of the iris, which are UV mapped to the flattened sphere. Hence our parametric model can control the eyeball size, iris & pupil size along with an approximate iris color.

We now discuss how some of these models were used in face and eye reconstruction.

3.2 3D reconstruction

3D reconstruction is a field in computer vision in which 3D shape and appearance of objects is captured and recovered from laser scans or image sequences. Until now, most methods reconstructing faces treat eyes as a textured part of the same mesh (Banf & Blanz (2009)) or ignore them. Whereas methods which focus on eye reconstruction, focus on recovering the detailed eye-ball or just focus on extracting the gaze by reconstructing some area around the eye region. Hence we discuss them separately in the following sections:

3.2.1 Face reconstruction

Face reconstruction in particular is a challenging task as it can deform non-rigidly along with varied rigid poses. Currently, face reconstruction is either done using a 3D multi-view setups in constrained settings, or by utilizing lightweight binocular or monocular camera inputs along with a strong deformable *template* model as a regularizers. We discuss (1.) general multi-view 3D face reconstruction, and (2.) light-weight 3D face reconstruction based on template models in detail below:

General multi-view 3D face reconstruction

These methods generally use sequences of images which essentially provide multiple views of the same object to reconstruct geometry and appearance. They can be further classified as multi-view methods & video-based methods.

1. **Multi-view methods**: Zhang *et al.* (2004), Weise *et al.* (2007) employed multiple cameras along with active structured light projectors to reconstruct facial geometry. Since the skin on the face lacks dense structure, projected light

provided dense surface texture which allowed for pair-wise pixel correspondences in multiple views, and thereby reconstructed the whole facial geometry using bundle adjustment (Hartley & Zisserman (2000)).

Ghosh *et al.* (2011) proposed a method to reconstruct detailed facial geometry, which utilized diffuse and specular albedo and normal maps. Their method consisted of an active multi-view setup, illuminated by polarized lighting patterns, which allowed them to separate diffuse and specular lighting from appearance.

Figure 3.4: Detailed face geometry reconstructed from images from seven camera-studio setup cameras (left) and binocular stereo camera (right). Image from Beeler *et al.* (2010).

Beeler *et al.* (2010) also introduced a high-quality 3D static scanning method, utilizing consumer passive stereo cameras like kinect. Their novelity lied in the calibration method which just required a few views of a sphere (of the size of an average face) with fiducial markers, which allowed facial reconstruction detailed enough to capture *mesoscopic* geometry, as seen in Figure 3.4.

2. **Video-based methods**: The above multi-view active or passive methods require complex camera setup and calibration. Williams (1990) introduced performance driven facial animation using marker-based motion capture for tracking the face from a *video*. Computer vision tracking techniques like structure from motion (SfM) and optic flow (Hartley & Zisserman (2000)) are used to reconstruct general 3D geometry.

   Chowdhury *et al.* (2002) estimated 3D geometry from a video sequence using SfM, and then utilized Markov Chain Monte Carlo (MCMC) sampling strategy to add deformations over a generic face model using the estimate from SfM. Fidaleo & Medioni (2007) tracked sparse feature points on the face in a video. They relied on the video having multiple views of the same face, which were then densely tracked for correspondence using an estimate from tracked sparse

feature points. Ghosh *et al.* (2011) and Garrido *et al.* (2013) utilized optic flow to align images in a video, for small motions between subsequent frames.

Next we move on to light-weight methods which use prior knowledge of natural facial geometry and albedo to regularize the problem.

Light-weight 3D face reconstruction based on *template* models

The facial reconstruction community has seen a general trend from complex camera setup in constrained settings to more light-weight monocular methods in general environments. The later make use of parametric models as described in section 3.1.1 to constrain an optimization problem.

Face models (AAMs and 3DMMs) capture different modes of deformation for shape and texture. Computed from 3D scans, they form a strong prior to act as a regularizer. An analysis-by-synthesis approach is used to fit these models to images by rendering while optimizing for the face model parameters such that the rendered synthetic face is close to the face in the image.

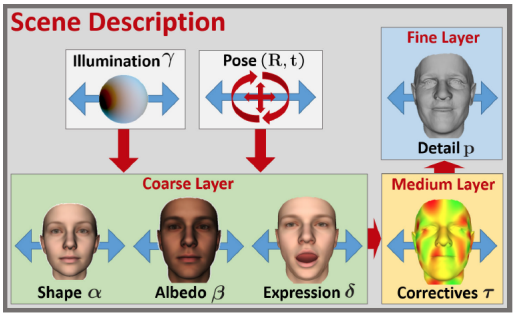


Figure 3.5: Face and scene model parameters used in the method by Garrido *et al.* (2016). Image from Garrido *et al.* (2016).

Garrido *et al.* (2013), Garrido *et al.* (2016) solved an optimization problem to obtain a set of facial shape, expression and pose paramters of the 3DMM, see Figure 3.5. Annotated 3D facial keypoints called *landmark* vertices on a 3DMM were projected to the detected 2D landmarks on the image, while solving for facial shape and blend-shape parameters of the face. Detected landmarks can be obtained from learned constrained local model(CLM) methods proposed by Cristinacce & Cootes (2006) and Saragih *et al.* (2010). Furthermore, Blanz & Vetter (1999) back-projected all the 3D vertices densely, which formed a photometric error to solve for face albedo and scene illumination. Simple illumination models like combination of ambient and one or more directional lights(used by Blanz & Vetter (1999)), or more versatile spherical harmonics (proposed by Ramamoorthi (2006)) are often used.

These generally consider the skin of the face as a Lambertian material. Garrido *et al.* (2013) also obtained detailed face geometry by utilizing monocular cues from shape-from-shading.

Quite recently, learning based methods (Richardson *et al.* (2016), Tewari *et al.* (2017)) have come into limelight. These methods utilize the power of diverse available image data to learn more modes of the base face model, in a semi-supervised fashion. An advantage of such methods over optimization based methods is that they don't require multiple iterations or a good initial guess. Since our method is not learning-based, we don't discuss them further.

Since a full projection from 3D to 2D is a non-linear operation, the problem is often posed as a non-linear and potentially non-convex function, which are iteratively solved using simple gradient descent or Gauss-Newton and Levenberg-Marquad solvers, which often require a good initial guess for the parameters.

3.2.2 Eye reconstruction

As we have seen in Section 3.1.2, eyes are spheres which sit inside the skull socket underneath (Lee *et al.* (1995)). But the frontal part of the eye is partially visible on the face, which includes iris, pupil and some parts of white sclera. We will now describe image-based methods which use these visible parts of the eye for reconstruction.

Iris structure reconstruction

Francois *et al.* (2009) used RGB images to recover detailed iris structure from color variation in an image. They approximated light subscattering properties of iridal tissues, which was then used to render a realistic eye model. A digital camera with a polarising filter was used to get images without specularities. Environment lighting was captured using multiple-exposure HDR images, which was used to calibrate their camera.

Eye-ball orientation from limbus boundary

Limbus is the elliptical boundary between sclera and cornea, and is often seen as a dark ring along the iris boundary. Methods proposed by Wood & Bulling (2014), Jeni & Cohn (2016), Dierkes *et al.* (2018) depend on the shape of limbus to retrieve the eye ball's orientation, by back projecting a 2D ellipse to a 3D circle using a full perspective matrix (Wu *et al.*, 2004). The normal of this circle then forms the eye-ball's optical axis. Eye size and iris size were kept equal to an average constant size known from physiological knowledge (Wang *et al.*, 2003). Limbus detection

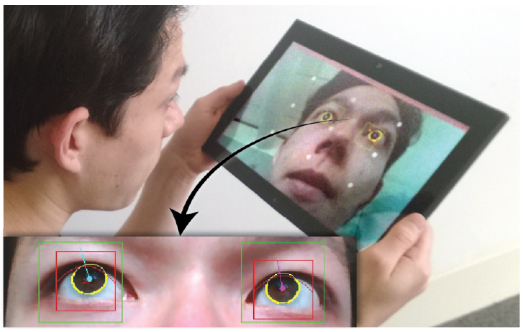


Figure 3.6: Eye-ball orientation from the limbus boundary. Image from Wood & Bulling (2014).

methods like circular hough transform (Xie & Ji, 2002), integro-differential operator (Daugman, 1993), RANSAC model fitting (Zhang *et al.*, 2010; **?**) were used by the above methods. However, the limbus detection is error prone as the eye lids make some horizontal and vertical parts of the limbus boundary invisible. Moreover, the orientation retrieved from the limbus boundary is not face-pose invariant, which is overcome by eye-model based approaches explained further.

Eye-ball reconstruction from images

Dierkes *et al.* (2018) proposed a 3D eye-ball fitting approach using input from head mounted near-eye cameras, which builds upon limbus fitting approach by Świrski & Dodgson (2013). They unproject pupil contour edges to 3D space, while fitting to an 3D model which also takes into account the geometry of buldged cornea and it's refraction effects. However the point of rotation of the eyeball and face pose is assumed to be fixed with respect to camera, as the input is taken from near-eye camera. Since we use a remote camera (which is not mounted on the head) in this thesis, we further discuss eye-model based reconstruction using inputs from remote cameras.

Bérard *et al.* (2016) constructed deformable eye model using data from Bérard *et al.* (2014), and used it to reconstruct high quality eye ball model from a multi-view images or a single image. The details included precise 3D eye ball structure, detailed iris and even the sclera vein network. However, their method required manual annotations of limbus boundary, pupil position, iris and sclera segments and specular highlights in the image.

Methods proposed by Ishikawa *et al.* (2004), Wen *et al.* (2017), Wang & Ji (2017), Chen & Ji (2008) assume that each person has a specific eye-ball radius, and their eye-ball centers are fixed, which can be called as individual anatomical

constraints. In an offline fashion, they calibrate for these constraints, where they collected multiple samples of gaze direction while keeping the head pose fixed, and solved for a personalized fixed 3D eye position and radius. More specifically, for example Wang & Ji (2017) calibrated the same using nine 3D pupil center and 2D PoR (Point of Regard) pairs to solve a linear system of equations. These methods the use these pre-calibrated parameters to track the eye for eye-gaze estimation. Some of these eye-gaze estimation methods are discussed in the next section.

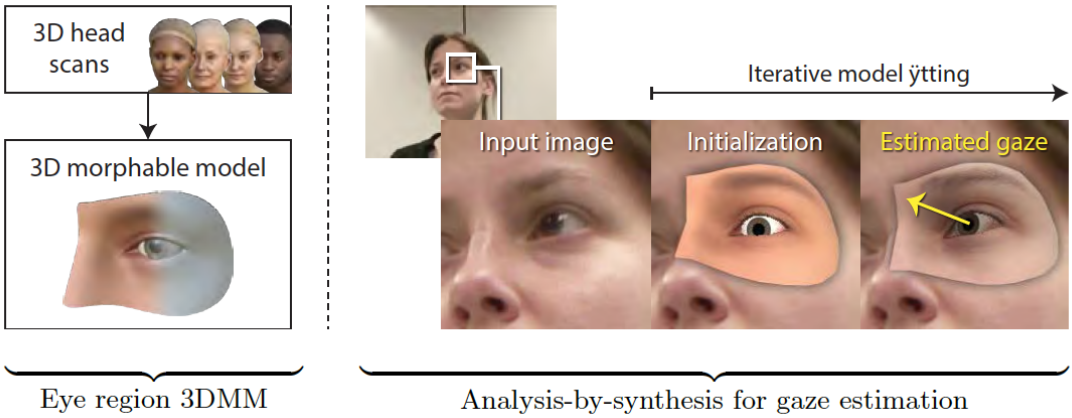


Figure 3.7: Eye-region morphable model. Image from Wood *et al.* (2016a).

Wood *et al.* (2016a) proposed an eye-region morphable model as seen in Figure 3.7, acting as the deformable *template* model used to inversely fit to images. The eye position and radius was already a part of this model, which was coupled to each face-scan used to create the model. Hence, while their eye ball center and radius were fixed and part of the combined eye-region model itself, we solve for a variable eye center and radius while coupling it with an existing full face morphable model which captures more variations as it was built using 10 times more faces.

While Wood *et al.* (2016a)'s model was built only for one side of the eye region with each eye ball rotating independently, Wood *et al.* (2018) extended it to two eyes using an extra constraint for interocular distance between the eyes.

Wang *et al.* (2016) on the other hand, reconstruct full face geometry, without the appearance of the face, along with eye-balls placed behind eye-lids, as seen in Figure 3.8. However, they do not explicitly model eye-ball rotations, and rather sample probable eye-gaze directions based on a MAP framework. This biases them to observed gaze directions of the person. We want to model explicit eye-rotations which can be part of the optimization problem.

Hence, this thesis aims to extend the works of Wood *et al.* (2016a) and Wang

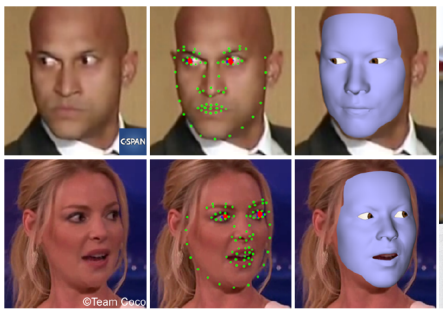


Figure 3.8: 3D face and eye-gaze performance capture. Image from Wang *et al.* (2016).

et al. (2016) to a generic full face 3DMM without constructing them again, but by optimizing for eye-ball parameters, such that fully functional eye-balls are reconstructed along with a 3DMM from monocular RGB face images. We also explore natural rotations of the eye-ball based on Listing's law (Helmholtz (1867)), which is not explicitly modeled in any of the methods which fit eyes. These were explained in detail in the background Section 2.4.2.

Since eye-gaze can be extracted from the orientation of the reconstructed eye-ball, we discuss some eye-gaze tracking methods.

3.3 Eye-gaze tracking methods

Eye-gaze tracking techniques have been studied thoroughly in the past, see Kar & Corcoran (2017) for full review. It is an important tool to conduct studies in psychology, ophthalmology and other medical research. Further-on, it also forms a basic input for gaze-based human computer interaction techniques. Since our approach aims to retrieve eye-ball position and orientation, we implicitly get the 3D eye-gaze from the eye orientation which is also head-pose invariant. Further, we discuss some different types of eye-gaze tracking methods, and then see how our method is similar or different from them.

In the past, invasive techniques which involved electro-occulography were used to study eye movements in three dimensions, which were further based on assumptions from Listing's law. These were done using search coils that are embedded into a tightly-fitting contact lens Robinson (1963).

With advances in computer vision, camera input is now used to infer eye gaze-directly. There are various methods which use multiple types of camera setups for the same. This includes head mounted eye trackers (Świrski & Dodgson, 2013; Dierkes *et al.*, 2018), multi-view camera systems with (Shih & Liu, 2004) or without NIR

(near-infrared) illumination (Zhao *et al.*, 2012), and now even with one camera in an unconstrained lighting setting (Zhang *et al.*, 2015; Sugano *et al.*, 2014). As the head moves, the inferred eye gaze should be ideally head-pose invariant. The head moves along with head-mounted eye-trackers, and so the gaze direction captured with them is independent of the head movement, i.e. head rotation is not coupled in the gaze direction. For all other methods involving cameras not mounted on the head, one has to explicitly de-couple the head rotation from eye-gaze direction. These camera-based methods can roughly be classified into 3 categories: NIR illumination based 2D regression of feature points, appearance based and model based methods.

NIR illumination based 2D regression of feature points These methods do a simple regression of the eye-gaze vector by detecting pupil center and corneal glint(s) created by NIR illumination, see figure 3.9. This is either done by a single camera NIR sensor (Blignaut, 2013), or multi-view camera systems (Arar & Thiran, 2017). Arar & Thiran (2017) use multi-view to get 3D head pose such that they can decouple the eye from it.

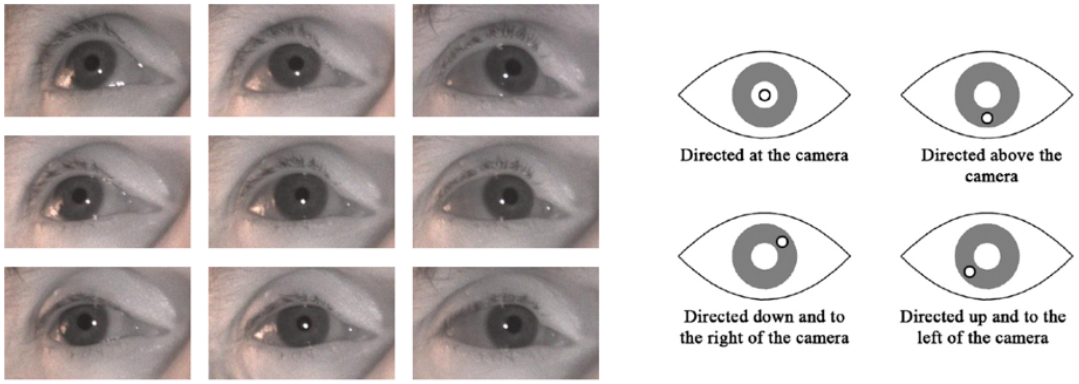


Figure 3.9: Left: Corneal reflection at various gaze positions (Blignaut, 2013). Right: Glint position changing according to point of regard (EL Haddioui & Khaldi, 2012)

Appearance-based methods These methods (Zhang *et al.* (2015), Sugano *et al.* (2014)) directly map image pixels of the eye region to an eye-gaze vector direction. The mapping is regressed or learned from labeled data. Ground truth data is collected and labeled by making people look at a specific point on a screen, the gaze vector then being the direction from the pupil center to the screen point (see Figure 3.10). Synthetically rendered eyes with ground truth gaze have also been used to learn the mapping between image pixels and eye region (Sugano *et al.* (2014), Wood

et al. (2016b)). Robustness of such appearance based methods depends on the data it was trained on, which includes varied facial features, illumination conditions and all the possible eye-gaze directions.

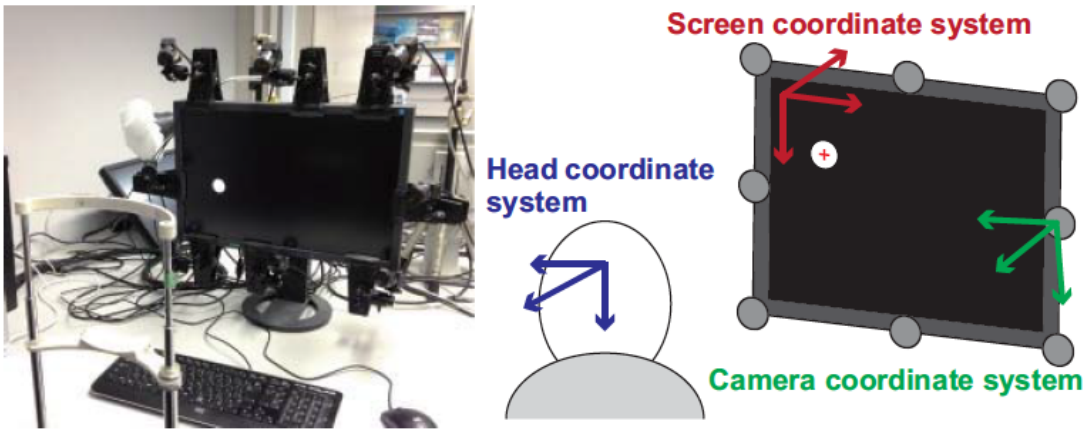


Figure 3.10: Ground truth collection setup for eye-gaze directions. Image from Sugano *et al.* (2014)

Model-based methods Eye-ball reconstruction methods as discussed in the section 3.2.2 directly infer eye-gaze direction based on the orientation of the eye-ball reconstructed from images. We also aim to infer gaze from our reconstructed eye-ball orientation. The method of Wood *et al.* (2016a) reconstructs eye-balls along with a small region around it, however want to capture the whole face along with the eyes.

3.4 Summary

In this thesis we make use of the 3DMM introduced by Blanz & Vetter (1999), which contains face shape and albedo basis to create a facial identity. Along with this, we use an extra set of basis vectors for facial expressions, created by Garrido *et al.* (2013). A simplified eye-ball model as explained is used along with this existing face model. It is a light-weight template-based method, which does face and eye-ball reconstruction from images. Eye-gaze is extracted from the retrieved eye-ball model's orientation.

Chapter 4

Methodology

This chapter discusses the method we have developed and implemented through the course of this thesis for reconstructing and tracking facial geometry and appearance along with the eye-ball orientations placed behind the mesh utilizing a sequence of monocular RGB images. We describe our formulations in terms of an optimization problem.

At first we use a close to frontal eye-gaze image of a person to fit a face and an eye model, thereby recovering model parameters which represent the reconstructed face. Then we use the rest of the images in the same sequence to track the person's facial expressions and pose, and the eye-gaze directions.

We take an analysis-by-synthesis model-fitting approach to inversely fit a morphable face model and two parametric eye models to the observed image. Section 4.1 formally describes the parameters which control the models, while Section 4.2 explains how these model parameters are used. This is done by defining the problem as an unconstrained non-linear least squares energy E, which is then minimized to recover a set of model parameters $\mathbf{x}^*$, representing the reconstructed mesh. Note that throughout this thesis, the model parameters are highlighted in red, and they act as free variables which are solved for. Section 4.3 then goes over the optimization strategy we use to solve the defined problem.

Similar to the structure of Chapter 3, we first describe the face and eye model parameters, followed by the energy formulations for the face and eye reconstruction.

4.1 Parametric models

We use the following as *template* models to regularize the ill-posed problem of recovering 3D geometry from monocular images. In further sections we explain the

morphable face model learned from data, followed by an anatomical eye model.

4.1.1 Face model

The face model is represented by a triangular face mesh, whose topology is assumed to be maintained while deforming it. The two types of transformations which are performed on this model are rigid and non-rigid. The pose of the face can be seen as the rigid transformation — a translation $\mathbf{t} \in \mathbb{R}^3$ and a rotation $R \in SO(3)$ applied on the whole mesh.

The parametric 3DMM face model as introduced in Section 3.1.1 describes 3D non-rigid face deformations linearly in a high-dimensional space. This model for face shape and albedo was created by linearly decomposing 200 scanned and reconstructed 3D face meshes that are in correspondence (Blanz & Vetter, 1999). For expressions, we use the linear blend-shape deformations on the same model from Garrido *et al.* (2016), which were created using 3D facial expression databases from Cao *et al.* (2014), Alexander *et al.* (2009), by taking per-vertex displacements from the neutral pose.

The face mesh is made up of $N_f = 24k$ vertices $\mathbf{v}_f = \{\mathbf{v}_{f,i} \in \mathbb{R}^3 |\ 1 \leq i \leq N_f\}$, each of which stores the vertex 3D position with respect to the face center, and we refer to this as the *face model space*. Each vertex is associated with a skin albedo color $\mathbf{a}_f = \{\mathbf{a}_{f,i} \in \mathbb{R}^3 | 1 \leq i \leq N_f\}$ in the RGB space. Associated vertex normals $\mathbf{n}_f = \{\mathbf{n}_{f,i} \in \mathbb{R}^3 | 1 \leq i \leq N_f\}$ are calculated using local one-ring neighborhood vertices (see Appendix A.3). The average face shape $\mathbf{f}_s$ for a neutral expression and average face albedo $\mathbf{f}_a$ for all the vertices of the face are stacked in vectors as:

$$\begin{aligned} \mathbf{f}_s \in \mathbb{R}^{3N_f} &= (x_1, y_1, z_1, x_2, y_2, z_2, ..., x_{N_f}, y_{N_f}, z_{N_f})^\top \\ \mathbf{f}_a \in \mathbb{R}^{3N_f} &= (R_1, G_1, B_1, R_2, G_2, B_2, ..., R_{N_f}, G_{N_f}, B_{N_f})^\top \end{aligned} \tag{4.1}$$

The decomposed linear orthonormal PCA basis which captures the highest modes of variation is represented as:

$$\begin{aligned} \text{shape } \mathbf{S} &= (\mathbf{s}_1, \mathbf{s}_2, .., \mathbf{s}_{M_s}) \in \mathbb{R}^{3N_f \times M_s} \\ \text{expression } \mathbf{E} &= (\mathbf{e}_1, \mathbf{e}_2, .., \mathbf{e}_{M_e}) \in \mathbb{R}^{3N_f \times M_e} \\ \text{albedo } \mathbf{A} &= (\mathbf{a}_1, \mathbf{a}_2, .., \mathbf{a}_{M_a}) \in \mathbb{R}^{3N_f \times M_a} \end{aligned} \tag{4.2}$$

where M_s, M_e, M_a were taken as the first $80, 64, 80$ modes of variations for face shape, expression, and albedo respectively. These were chosen to capture 99% of the variance of the data.

Finally, the face vertex positions $\mathbf{v}_f$ and albedo $\mathbf{a}_f$ can then be expressed as a linear combination of these variations captured by the basis (Equation 4.1) on top of the average vectors (Equation 4.2):

$$\begin{aligned}\mathbf{v}_f(\boldsymbol{\delta}, \boldsymbol{\epsilon}) \in \mathbb{R}^{3N_f} &= \mathbf{f}_s + \sum_{i=1}^{M_s} \delta_i \mathbf{s}_i + \sum_{i=1}^{M_e} \epsilon_i \mathbf{e}_i \\ &= \mathbf{f}_s + \mathbf{S}\boldsymbol{\delta} + \mathbf{E}\boldsymbol{\epsilon}\end{aligned} \tag{4.3}$$

$$\begin{aligned}\mathbf{a}_f(\boldsymbol{\beta}) \in \mathbb{R}^{3N_f} &= \mathbf{f}_a + \sum_{i=1}^{M_a} \beta_i \mathbf{a}_i \\ &= \mathbf{f}_a + \mathbf{A}\boldsymbol{\beta}\end{aligned} \tag{4.4}$$

where coefficients $\boldsymbol{\delta} \in \mathbb{R}^{M_s}$, $\boldsymbol{\epsilon} \in \mathbb{R}^{M_e}$ and $\boldsymbol{\beta} \in \mathbb{R}^{M_a}$ linearly parameterize face shape, expression and albedo respectively.

4.1.2 Eye model

A parametric eye model is represented by a spherical mesh which is flattened on the front (Wood *et al.*, 2016a), see Figure 4.1. We assume that both the eye-balls on the face have the same geometry and appearance while they only differ in the way they are rotated. This mesh is made up of $N_e = 4k$ vertices $\mathbf{v}_e = \{\mathbf{v}_{e,i} \in \mathbb{R}^3 | 1 \leq i \leq N_e\}$, each of which stores the 3D offsets of the vertices with respect to the sphere's center.

To avoid shading artifacts at the junction where the sphere is flattened, associated (pseudo-)normals for each eye vertex $\mathbf{n}_e = \{n_{e,i} \in \mathbb{R}^3 | 1 \leq i \leq N_e\}$ are calculated for the complete sphere (see Figure A.2 in Appendix), even for the vertices which lie on the flat area. These are represented by the vector joining the center of the sphere to any vertex on the eye model.

For eye orientations, we make use of a bi-polar Helmholtz coordinate system (Run & Berg, 1993), which utilizes only three angles θ, α and ν to control the rotations of two eye-balls in a natural manner, see Figure 2.11. The rotation matrices $R_{L|R}$ for both eyes can be derived from the respective rotation vectors $\mathbf{r}_L$, $\mathbf{r}_R$ as described in the background Section 2.4.2, Equation 2.16. Subscript $L|R$ is used to differentiate between the left and the right eye. This angle formulation implicitly captures natural eye-ball rotations such that both rotate together to focus on a point or infinity. This is in contrast to the method by Wood *et al.* (2016a) which solves only for one eye-ball rotation at a time, and hence each eye is free to move independently, which can result in unnatural conditions where both eyes can focus on two different points.

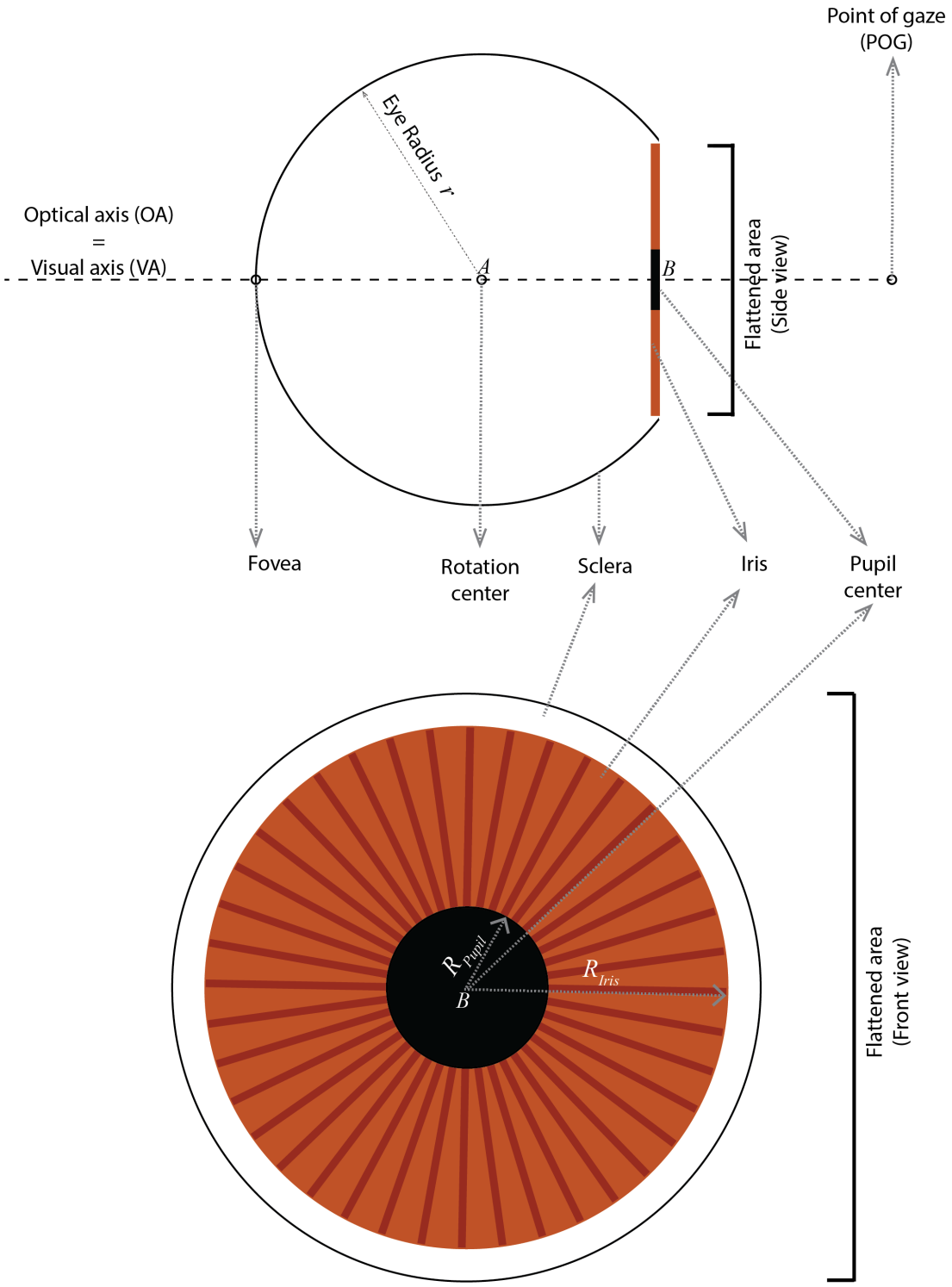


Figure 4.1: The simplified eye model described.

Since the size of the eye-ball varies from person to person, we define a scaling factor $s \in \mathbb{R}$ which scales all the vertices from the center of the sphere. Hence the scaled eye vertices can be written as:

$$\bar{\mathbf{v}}_{e,i}(s) = \text{scale}(s)(\mathbf{v}_{e,i}, 1)^\top \in \mathbb{R}^4 \tag{4.5}$$

Note that the eye-ball radius scales (Equation 2.2) by the same factor , such that the scaled eye-ball radius becomes sr_{orig}, where r_{orig} is the original eye-ball radius.

A set of aligned iris images, as seen in Figure 4.2 are uv mapped[1] to the flattened sphere, such that each vertex is assigned a color. Since we have only $k = 6$ iris images $T_j \in \mathbb{R}^{M \times M} | \{1 \leq j \leq k\}$, where M is the resolution of the texture image T, a simple color mixture model can be employed to get a new iris image. Then the albedo $\mathbf{a}_{e,i} \in \mathbb{R}^3$ of the i^{th} eye-ball vertex $v_{e,i}$ which is uv mapped on the texture can be described as:

$$\mathbf{a}_{e,i}(\boldsymbol{\tau}) = T_a(u_i, v_i) + \sum_{j=1}^{k} \tau_j \left(T_j(u_i, v_i) - T_a(u_i, v_i)\right) \in \mathbb{R}^3 \tag{4.6}$$

$$\text{where } T_a(u_i, v_i) = \frac{1}{k} \sum_{j=1}^{k} T_j(u_i, v_i) \tag{4.7}$$

The coefficients $\boldsymbol{\tau} \in \mathbb{R}^k$ parameterize the eye color of the i^{th} vertex and u_i, v_i is the pixel position on the texture map to which the vertex i is mapped.

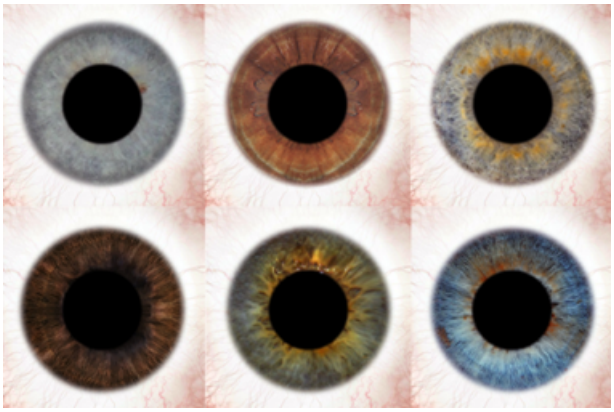


Figure 4.2: Iris texture images T from Wood (2017).

Since the iris and pupil are mapped to a flat circle on the sphere, the vertices on this flat circle can be scaled radially about the center of this circle $p \in \mathbb{R}^3$, which is

[1] 2D texture image is projected on a 3D model for texture mapping

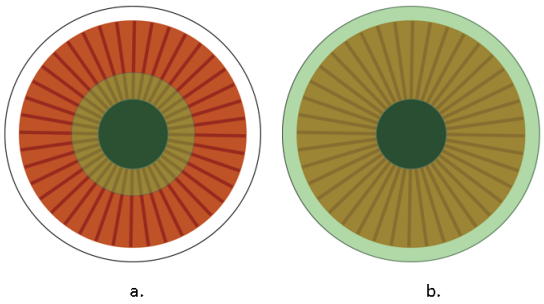


Figure 4.3: The mask used to scale the (a.) pupil and (b.) iris vertices in Equation 4.9 are highlighted in green. This mask is the area which is uv mapped to the vertices. Note that pupil vertices also extend to the iris area to avoid scaling artifacts on the boundary.

also called the *pupil center*. The size of the iris and pupil can henceforth be defined as the radial scaling of vertices:

$$v_{iris,i} = (v_{iris,i} - p)s_{iris} + p \in \mathbb{R}^3 \tag{4.8}$$

$$v_{pupil,i} = (v_{pupil,i} - p)s_{pupil} + p \in \mathbb{R}^3 \tag{4.9}$$

where $v_{iris,i}, v_{pupil,i}$ are vertices on the flattened area (as in Figure 4.3) and scale factors $s_{iris} \in \mathbb{R}$ and $s_{pupil} \in \mathbb{R}$ control the size of the iris and the pupil respectively.

All the above transformations are applied to the eye-ball vertices in the *eye model space* centered about the sphere center. The left and the right eye-balls are then *coupled* to the face mesh by translating the sphere centers to a position behind the opening between the eye-lids on the face mesh. This position of the eye-balls is defined in the face model space. Figure 4.4 illustrates this coupling.

We make use of symmetry of the face to define the eye-center positions in the face model space. By assuming that these center positions are equidistant from both the sides of the nose, the x-coordinate can be written as $-c_x$ and c_x for the left and the right eyes respectively. Furthermore, since both the eyes-balls are of the same size, their distance from the eye-lids is the same, and hence a common c_z is valid. Owning to face symmetry, c_y is common for both the eyes. Thereby the left $\mathbf{c}_L$ and the right $\mathbf{c}_R$ eye-centers in the face model space can be defined as:

$$\begin{aligned} \mathbf{c}_L &= (-c_x, c_y, c_z)^\top \\ \mathbf{c}_R &= (c_x, c_y, c_z)^\top \end{aligned} \tag{4.10}$$

In the following section, we see how these model parameters are used in our energy formulations.

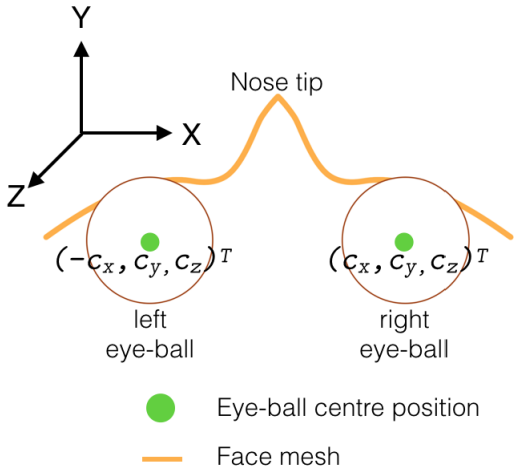


Figure 4.4: Top-view of the face mesh along with coupled eye-balls. The eye-ball centers for the left and the right eye-balls are shown.

4.2 3D reconstruction

This section describes the method for reconstruction and tracking of faces and eyes from monocular 2D input. This includes both geometry and appearance of the reconstructed face mesh, as well as the eye ball positions and orientations. The problem is formulated as an unconstrained optimization problem, such that we find a set of parameters $\mathbf{x} \in \{\mathbf{x}_f, \mathbf{x}_e\}$, by minimizing an energy E, which we define as:

$$\begin{aligned} E(\mathbf{x}) &= E_f(\mathbf{x}_f) + E_{L|R}(\mathbf{x}_e) \\ E_f(\mathbf{x}_f) &= \underbrace{E_{sparse}(\mathbf{x}_f) + w_{photo}E_{photo}(\mathbf{x}_f)}_{\text{data terms}} + \underbrace{w_{reg}E_{reg}(\mathbf{x}_f)}_{\text{regularizers}} \\ E_{L|R}(\mathbf{x}_e) &= \underbrace{w_{coupling}E_{coupling}(\mathbf{x}_e) + w_{pupil}E_{pupil}(\mathbf{x}_e)}_{\text{data terms}} \\ &\quad + \underbrace{w_{size,reg}E_{size,reg}(\mathbf{x}_e) + w_{ortho,reg}E_{ortho,reg}(\mathbf{x}_e)}_{\text{regularizers}} \end{aligned} \tag{4.11}$$

Model parameters $\{\mathbf{x}_f, \mathbf{x}_e\}$ used in our energy formulations are defined in Tables 4.1, 4.2. Note that we do not use all the parameters described in Section 4.1.2. We explain each of the above described energy terms $E_f(\mathbf{x}_f)$ for the face and $E_{L|R}(\mathbf{x}_e)$ for the eyes in detail in the next sections. Each energy term is formulated as the sum of residual errors that define a non-linear least squares problem. Note that the subscript $L|R$ implies that terms for each individual left and right eyes are

formulated separately.

Table 4.1: Face model parameters used $\mathbf{x}_f$

$\boldsymbol{\delta} \in \mathbb{R}^{M_s}$	face shape
$\boldsymbol{\epsilon} \in \mathbb{R}^{M_e}$	face expression
$\boldsymbol{\beta} \in \mathbb{R}^{M_a}$	face albedo
$\phi_{R,\mathbf{t}}, R \in SO(3), \mathbf{t} \in \mathbb{R}^3$	face rigid pose — rotation and translation
$\boldsymbol{\kappa} \in \mathbb{R}^{9\times 3}$	scene illumination

Table 4.2: Eye model parameters used $\mathbf{x}_e$

$c_x \in \mathbb{R}$	eye center parameter (x-coordinate) w.r.t. the face model space
$c_y \in \mathbb{R}$	eye center parameter (y-coordinate) w.r.t. the face model space
$c_z \in \mathbb{R}$	eye center parameter (z-coordinate) w.r.t. the face model space
$\theta \in \mathbb{R}$	vertical rotation
$\alpha \in \mathbb{R}$	horizontal rotation
$\nu \in \mathbb{R}$	eye vergence
$s \in \mathbb{R}$	eye-ball radius scale

While the data terms are posed in terms of the above defined model parameters, regularizers are needed to prevent over-fitting such that the parameter values do not go outside an expected range. The weights in front of the energy terms control their importance, i.e. the higher the weight, the more is the penalty, and the optimizer will work more towards minimizing that energy term compared to the others. These weights were determined empirically during our experiments. The weights $w_{photo} = 5$, $w_{reg} = 2.5 \times 10^{-6}$, $w_{coupling} = 2$, $w_{pupil} = 20$, $w_{size,reg} = 1$, and $w_{ortho,reg} = 0.0125$ balance the importance of terms.

As explained in the Section 2.3.3, we take a hierarchical approach for minimizing the non-convex E. At first we recover the face model parameters $\mathbf{x}_f$ by minimizing $E_f(\mathbf{x}_f)$, and the go on to add $E_e(\mathbf{x}_e)$ to the problem while keeping $\mathbf{x}_f$ constant. This will be further discussed in the section below.

4.2.1 Face reconstruction

$$\begin{aligned} E(\mathbf{x}) &= E_f(\mathbf{x}_f) + E_{L|R}(\mathbf{x}_e) \\ E_f(\mathbf{x}_f) &= \underbrace{E_{sparse}(\mathbf{x}_f) + w_{photo}E_{photo}(\mathbf{x}_f)}_{\text{data terms}} + \underbrace{w_{reg}E_{reg}(\mathbf{x}_f)}_{\text{regularizers}} \end{aligned} \tag{4.12}$$

The energy formulation $E_f(\mathbf{x}_f)$ for the face is based on Garrido *et al.* (2016) (coarse scale), where a synthesized image of the 3D model of the face is matched to the

observed monocular RGB image, at first sparsely and then densely while solving for parameters $\mathbf{x}_f$ described in Table 4.1. This is done at first on the frontal gaze image before solving for eye model parameters, then on rest of the images of the sequence. The optimized face model parameters from the first image are used to initialize the the problem for the rest of the images in the sequence.

As described in Equation 4.3, at first face vertices are non-rigidly deformed in the face model space, using variable parameters $\boldsymbol{\delta}$, $\boldsymbol{\epsilon}$. In monocular reconstruction there is an inherent depth and scale ambiguity where the size of the face can be increased either by scaling or by coming closer to the camera. We assume the latter and do not scale the mesh, but translate it towards the camera to increase the size. So the whole face mesh is then rigidly rotated and translated by $\phi_{R,\mathrm{t}}$, to assume a face pose. This transforms all the face vertex positions from the face model space to the world space. We consider the camera to be located at the origin of the world space, and hence the transformation from the world to the camera space can be seen as an identity transform.

Screen space position $\mathbf{p}_i$ of a homogenized vertex position $\bar{\mathbf{v}}_{f,i} \in \mathbb{R}^4$ of the mesh can be seen as a full perspective projection by π as explained in the background Section 2.2.2, Equations 2.4, 2.5:

$$\mathbf{p}_i(\mathbf{x}_f) = \pi \cdot \phi_{R,\mathrm{t}} \cdot \bar{\mathbf{v}}_{f,i}(\boldsymbol{\delta}, \boldsymbol{\epsilon}) \in \mathbb{R}^2 \tag{4.13}$$

A color $\mathbf{c}_i \in \mathbb{R}^3$ is associated with the i^{th} projected vertex, which accounts for the appearance of the face. $\mathbf{c}_i$ is defined as the skin albedo color multiplied by the scene illumination L (as described in the background Section 2.2.4, Equation 2.6):

$$\mathbf{c}_i(\mathbf{x}_f) = \mathbf{a}_{f,i}(\boldsymbol{\beta}) \cdot L(\mathbf{n}_i(\boldsymbol{\delta}, \boldsymbol{\epsilon}), \boldsymbol{\kappa}) \in \mathbb{R}^3 \tag{4.14}$$

Note that $\mathbf{n}_i$ is the vertex normal calculated using the one-ring neighborhood of the deformed vertices. Since the illumination function is defined using the camera space normals, the normals are transformed to the camera space by $\phi_{R,\mathrm{t}}^{-1}\ (\mathbf{n}_i, 0)^\top = R^\top\ (\mathbf{n}_i, 0)^\top$ using the property that vectors are not affected by translations and $R^{-1} = R^\top$ for orthonormal matrices.

We now move on to residual formulations defined as the difference between *estimated* and *observed* values in a least-squares sense, such that after iteratively minimizing the residuals by the combined energy terms, we reach a set of *estimated* parameters $\mathbf{x}_f^*$, which synthesize the face image close to the original monocular face image. Such an approach is also known as *analysis-by-synthesis*.

As the hierarchical approach explained before, we formulate an energy term

E_{sparse} which makes use of a subset of landmark vertices on the mesh to create a simpler problem. This term estimates the non-rigid facial shape deformations as well as the rigid face pose jointly. After minimizing E_{sparse}, a dense term E_{photo} is then added to the energy E which makes use of all the vertices on the face mesh to further solve for the face appearance. The solution after minimizing E_{sparse} provides a good initialization for the harder problem when E_{photo} is added.

For sparse face reconstruction, we use a small set of detected landmarks (Cristinacce & Cootes, 2006; Saragih *et al.*, 2010) on the face image to sparsely solve for face pose $\phi_{R,\mathbf{t}}$, facial shape $\boldsymbol{\delta}$ and expression $\boldsymbol{\epsilon}$. Then for dense face reconstruction, we use the all face vertices to densely solve for face color $\boldsymbol{\beta}$ and scene illumination $\boldsymbol{\kappa}$.

Sparse face reconstruction

A set of $N_{LM} = 66$ landmarks $\{\mathbf{d}_i \in \mathbb{R}^2 |\ 1 \leq i \leq N_{LM}\}$ on the face image are detected using Saragih *et al.* (2010)'s method. The same landmark feature points are annotated on the neutral face mesh as 3D landmark vertices. Corresponding N_{LM} vertex indices are $k_i \in \{1, ..., N_{LM}\}$ for the these 3D landmark vertices. Projected screen space positions $\mathbf{p}_{k_i}$ for the 3D landmark vertices are described by Equation 4.13. We then enforce that these 3D landmark vertices should be projected close to the detected 2D landmarks on the image, in a least-squares sense:

$$E_{sparse}(\mathbf{x}_f) = \frac{1}{N_{LM}} \sum_{i=1}^{N_{LM}} \|\mathbf{p}_{k_i}(\mathbf{x}_f) - \mathbf{d}_i\|_2^2 \tag{4.15}$$

A statistical regularization prior is enforced on the face model parameters $\boldsymbol{\delta}, \boldsymbol{\epsilon}, \boldsymbol{\beta}$ as it is assumed that the PCA basis was built on a normally distributed subset of face meshes. This enforces that face shape, expression and albedo parameters do not degenerate the mesh to non-plausible facial geometry and color. This is done by making the model parameters stay close to the mean:

$$E_{reg}(\mathbf{x}_f) = \boldsymbol{\delta}^\top \boldsymbol{\delta} + w_\epsilon \boldsymbol{\epsilon}^\top \boldsymbol{\epsilon} + w_\beta \boldsymbol{\beta}^\top \boldsymbol{\beta} \tag{4.16}$$

As in Tewari *et al.* (2017), our PCA basis vectors $\mathbf{b}_i \in \{\mathbf{s}_i, \mathbf{e}_i, \mathbf{a}_i\}$ are variance scaled, as they are scaled by their respective standard deviations σ_i, such that $\mathbf{b}_i^\top \mathbf{b}_i = \sigma_i^2$. The weights were chosen to be $w_\epsilon = 10^{-3}$ and $w_\beta = 1$.

During optimization, after the sparse term E_{sparse} initializes the face geometry $\boldsymbol{\delta}, \boldsymbol{\epsilon}$ and face pose $\phi_{R,\mathbf{t}}$ parameters, we move on to adding a dense term which works on all the face vertices.

Dense face reconstruction

Let $I_i := I_i(\mathbf{p_i}(\mathbf{x}_f)) \in \mathbb{R}^3$ be the *observed* color intensity in the RGB space at the projected screen space vertex position $\mathbf{p}_i$ of the vertex $\mathbf{v}_{f,i}$. The original monocular face image is denoted by I. The *estimated* color of this vertex $\mathbf{v}_{f,i}$ is $\mathbf{c}_i$ (Equation 4.14). A photometric loss is employed such that the *observed* pixel color intensity I_i is close to the *estimated* vertex color $\mathbf{c}_i$ for all the vertices $\mathbf{v}_f$ on the face mesh:

$$E_{photo}(\mathbf{x}_f) = \frac{1}{N_f} \sum_{i \in \mathbf{v}_f} \| I_i(\mathbf{p}_i(\mathbf{x}_f)) - \mathbf{c}_i(\mathbf{x}_f) \|_2^2 \tag{4.17}$$

Here we solve for all parameters namely, face geometry $\boldsymbol{\delta}, \boldsymbol{\epsilon}$, face albedo $\boldsymbol{\beta}$, face pose $\phi_{R,\mathbf{t}}$, and scene illumination $\boldsymbol{\kappa}$ parameters. Since there are $24k$ vertices which can be hard for the optimizer to solve at once, at every 10th iteration we sub-sample a set of $2k$ vertices to be used for solving, and iterate for 100 iterations.

Since the nature of the problem is non-linear and non-convex, without solving for the sparse term in the beginning, we can get stuck in local minima if we directly try to solve the sparse E_{sparse} and the dense terms E_{sparse} together.

In the next section, we formulate the energy terms $E_{L|R}$ for eye reconstruction, which are used for optimization after face reconstruction.

4.2.2 Eye reconstruction

This section explains how we formulate our energy terms to fit the eye-balls to the reconstructed face mesh, such that they are placed behind the eye-lids and assume a natural orientation. A sequence of monocular RGB images are used for the same. An observed face image with a frontal gaze of a person, is used to *couple* the eye-balls to the reconstructed face geometry. This is a person-specific step used to find the eye-ball center parameters c_x, c_y, and c_z and eye-ball radius scale s. Post this, the eye rotation parameters θ, α, and ν are estimated for the other images in the same sequence. This is a tracking step, which captures changes in the face pose and expression, along with the eye-gaze direction.

Table 4.2 lists the eye-model parameters explained in Section 4.1.2 which we solve for using $E_{L|R}$ during optimization.

$$\begin{aligned} E_{L|R}(\mathbf{x}_e) = & \underbrace{w_{coupling} E_{coupling}(\mathbf{x}_e) + w_{pupil} E_{pupil}(\mathbf{x}_e)}_{\text{data terms}} \\ & + \underbrace{w_{size,reg} E_{size,reg}(\mathbf{x}_e) + w_{ortho,reg} E_{ortho,reg}(\mathbf{x}_e)}_{\text{regularizers}} \end{aligned} \tag{4.18}$$

The eye centers $\mathbf{c}_L$, $\mathbf{c}_R$ are defined by Equation 4.10 with respect to the face model space. Hence the eye pose can be defined by a rigid transformation $\phi_{L|R}(R_{L|R}, \mathbf{c}_{L|R})$ based on a rotation $R_{L|R}(\theta, \alpha, \nu)$ and a translation $\mathbf{c}_{L|R}(c_x, c_y, c_z)$. $\phi_{L|R}$ rotates each eye-ball about the sphere center and then translates them from the eye model space to the face model space, positioning them behind the eye-lids of the face. ϕ, the face rigid pose further transforms each eye-ball to the world space from the face model space. As stated before, the world to camera space is an identity transform. Like for the face, the screen space position $\mathbf{p}_{e,i}$ of a homogenized eye-ball vertex $\bar{\mathbf{v}}_{e,i} \in \mathbb{R}^4$ at index i can be written as:

$$\mathbf{p}_{e,i}(\mathbf{x_e}) = \pi \cdot \phi_{R,\mathbf{t}} \cdot \phi_e(R_{L|R}, \mathbf{c}_{L|R})\bar{\mathbf{v}}_{e,i}(s) \in \mathbb{R}^2 \tag{4.19}$$

Eye-ball calibration from frontal eye-gaze image

This section elaborates on the energy terms formulated, which make use of the RGB monocular image with a frontal gaze to find the eye-ball centers $\mathbf{c}_{L|R}$ and radius sr_{orig}. Among a sequence of given face images for a person, we select the one which has a frontal gaze, i.e. the one in which the person is looking straight in front. This the initial calibration step, after which we track the eye-gaze on other images from the given image sequence. This selection of the frontal gaze image can be automatic, by using a gaze-tracker to select the frontal-gaze image.

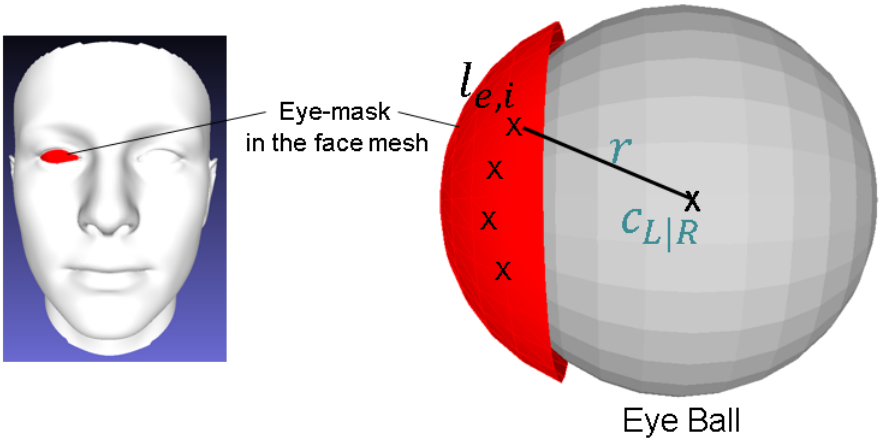


Figure 4.5: Eye-ball coupled with the face behind the eye-lids. The eye-mask contains the face vertices to which we couple the eye-ball.

1. **Eye-ball coupling**

   After the face reconstruction, we use a coupling term $E_{coupling}$ to bring two spheres near the eye region. Our assumption is that the face mesh vertices

around the eye-lids $\mathbf{l}_{e,k_i}$ hold a sphere, and hence their distance $\|\mathbf{l}_{e,k_i} - \mathbf{c}_{L|R}\|_2$ from the center $\mathbf{c}_e$ of the eye-ball should be approximately equal to the radius of the eye-ball sr_{orig}, as seen in Figure 4.5. This is again proposed in a least-squares sense:

$$E_{coupling}(\mathbf{x}_e) = \frac{1}{N_{l_e}} \sum_{i=1}^{N_{l_e}} \| \|\mathbf{l}_{e,k_i} - \mathbf{c}_{L|R}\|_2^2 - (sr_{orig})^2 \|_2^2 \tag{4.20}$$

Here, N_{l_e} are the number of face vertices $\mathbf{l}_{e,k_i} \in \{\mathbf{v}_{f,i}(\boldsymbol{\delta}, \boldsymbol{\epsilon}) \mid 1 \leq k_i \leq N_{l_e}\}$, around the eye-lid which are considered for this coupling term. We will call them *eye-mask* vertices. Here corresponding N_{l_e} vertex indices are $k_i \in \{1, ..., N_{l_e}\}$.

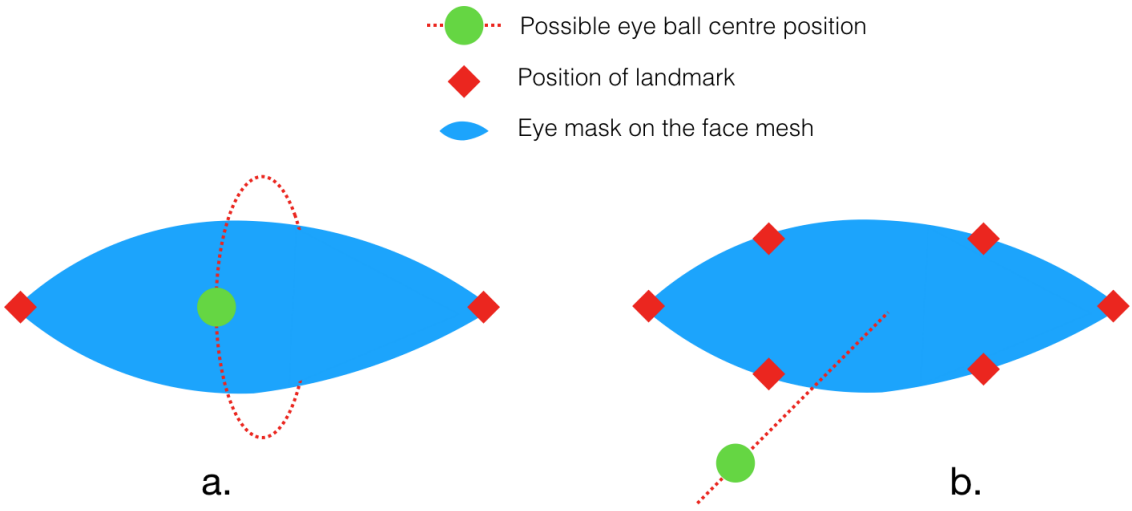


Figure 4.6: Eye-ball center locus around the eye-lids. **a**. The two landmarks shown are eye corner points. **b**. The landmarks shown are all the detected landmarks around the eye region.

Figure 4.6 shows the result of considering a subset of these vertices $\mathbf{l}_{e,i}$, and the possible eye center position loci. If we only use two eye-corner points of the eye region to fit a sphere, the locus of the sphere center lies on a circular disk with infinite radius, as seen in Figure 4.6 **a**. Using six landmark points will restrict this space to two points along the line, one in front of the eye-mask and one behind it, as in Figure 4.6 **b**. Furthermore, as we assume that these eye-mask vertices hold a sphere, these eye-mask vertices on the face do not

lie on a plane but form a curve, which restricts the eye ball center to lie only behind the face mesh.

In our experiments, this curve was not completely reliable to give an exact radius. The curvature was small and bloated up the fitted eye-ball radius. Hence, we regularized the eye-ball size, by employing an empirical anatomical soft constraint $E_{size,reg}$ which keeps the eye-ball diameter roughly equal to the distance between the outer and the inner eye-corners (seen in Figure 4.7):

$$E_{size,reg} = \|\|\mathbf{l}_{e,outer} - \mathbf{l}_{e,inner}\|_2^2 - (2 \cdot 0.8 \cdot s\ r_{orig})^2\|_2^2 \tag{4.21}$$

The factor 0.8 is based on the fact that the eye-ball diameter is bigger than the distance between eye-corner points, and is selected empirically.

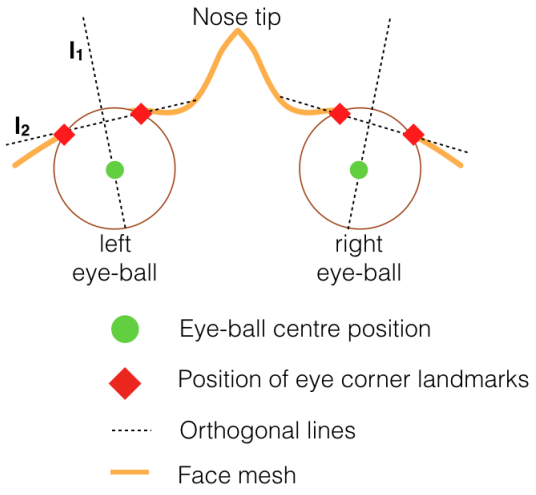


Figure 4.7: Eye-ball centers with respect to the face mesh

Since the curve of the eye-mask vertices is not reliable to find an eye-ball center point, we want to restrict the line along which the eye-ball center $\mathbf{c}_{L|R}$ should lie. This line $\mathbf{l}_1$ can be seen in the Figure 4.7. The eye region of the face mesh is tilted, such that if the face mesh is facing us, the inner eye-corner is close to the nose is closer to us than the outer eye-lid corner.

To define $\mathbf{c}_{L|R}$ in terms of line $\mathbf{l}_1$, a soft orthogonality constraint $E_{orth,reg}$ is proposed. If the line $\mathbf{l}_2$ between the two the eye-corners is perpendicular to the line $\mathbf{l}_1$, defined from $\mathbf{c}_{L|R}$ to the center between two the eye-corners, the space is limited to a plane, defined by $E_{orth1,reg}$. To further restrict the space to a line, the vertical axis in eye model space $\mathbf{l}_3$ is enforced to be perpendicular

to $\mathbf{l}_1$.

$$\begin{aligned}\mathbf{l}_1 &= \mathbf{c}_{L|R} - (\mathbf{l}_{e,outer} + \mathbf{l}_{e,inner})/2 \\ \mathbf{l}_2 &= \mathbf{l}_{e,outer} - \mathbf{l}_{e,inner} \\ \mathbf{l}_3 &= (0,1,0)^\top\end{aligned} \tag{4.22}$$

These are enforced by minimizing the dot product between the lines:

$$\begin{aligned}E_{orth,reg} &= E_{orth1,reg} + E_{orth2,reg} \\ E_{orth1,reg} &= (\hat{\mathbf{l}}_1 \cdot \hat{\mathbf{l}}_2)^2 \\ E_{orth2,reg} &= (\hat{\mathbf{l}}_1 \cdot \hat{\mathbf{l}}_3)^2\end{aligned} \tag{4.23}$$

as $\hat{\mathbf{l}}_a \cdot \hat{\mathbf{l}}_a = \|\hat{\mathbf{l}}_a\|\|\hat{\mathbf{l}}_b\| cos(\omega)$ is zero when $\omega = 90°$, where $\hat{\mathbf{l}}$ represents a unit vector.

The $E_{coupling}$, $E_{size,reg}$, and $E_{orth,reg}$ ensure that the eye-balls lie behind the face mesh and find an eye-ball radius.

2. **Eye-ball alignment** We further formulate E_{pupil} which makes the xy coordinates of eye-ball centers person-specific.

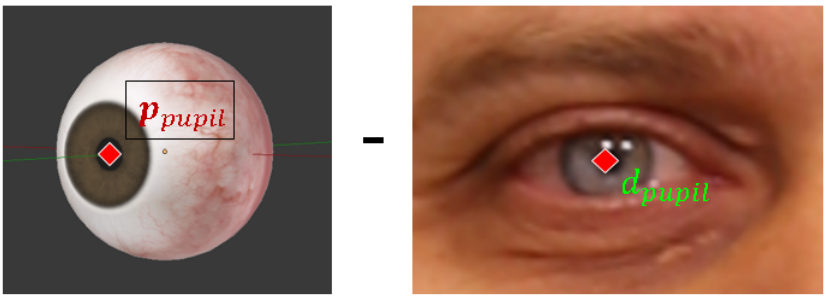


Figure 4.8: Projection of the 3D pupil position to the detected 2D pupil position.

The E_{pupil} term aligns the projected pupil 3D position $\mathbf{p}_{pupil}$ on the eye-ball mesh to a detected 2D pupil position d_{pupil} on the image (as seen in Figure 4.8):

$$E_{pupil} = \|\mathbf{p}_{pupil}(\mathbf{x_e}) - d_{pupil}\|_2^2 \tag{4.24}$$

While $\mathbf{p}_{pupil}$ is a function of eye rotation parameters θ, α, and ν), eye-ball center parameters c_x, c_y, and c_z and eye-ball radius scale s, we keep the rotations fixed when we use the frontal eye-gaze image. This is done because the optimizer can both rotate the eye-balls or translate them to align to the detected pupil

position. To disambiguate this, we either solve for translations or rotations at a time. For the frontal gaze image, we assume that the eye-ball is not rotated from the frontal direction.

Eye-ball orientation for eye-gaze tracking

After the eye-ball calibration using a monocular RGB image with a frontal gaze, we move to the tracking phase. Here we use other monocular RGB images to first solve for face pose and expression, given the previous set of optimizer parameters. Then $E_{L|R}(\mathbf{x}_e)$ is used to solve for eye rotations θ, α, and ν, keeping the eye-ball radius and centers same as the one solved for using the frontal gaze image.

4.3 Optimization strategy

The method described above is a multi-step reconstruction strategy to reconstruct and track face geometry and appearance along with coupled eyes from monocular images. We follow a step-wise approach, where we keep adding more energy terms to the previous formulation, as we go from a simple to a more difficult problem. The solution from the previous problem is used as an initialization for the parameters already solved for. This avoids getting stuck in local minima of the highly non-convex problem. For instance, it is easier to find the eye-ball orientations with respect to the face after a rigid face pose is found.

As seen in Figure 4.9, at first E_{sparse} is minimized, by solving iteratively for t_{sparse} iterations, followed by adding E_{photo} to this problem, which is again solved for t_{photo} iterations and so on. Note that the previous energy terms are still active, while the parameters listed in front of each steps are jointly solved for, keeping the other parameters constant.

The energy terms form an unconstrained non-linear least-squares problem. It is solved using Levenberg-Marquardt optimization algorithm explained in the background section 2.3, a pseudo-second order method, which makes use of the gradient information of the parameters, along with an estimate of the Hessian. As this algorithm requires an initial estimate, our joint optimization strategy provides the same while going from an easy to a difficult problem. For the beginning, we provide parameter values for a neutral face, and a frontal face pose, along with initial guess for the eye positions for the same face mesh.

We use *Ceres* C++ library which provides the Levenberg-Marquardt optimization algorithm, along with an auto-differentiation framework which evaluates generic

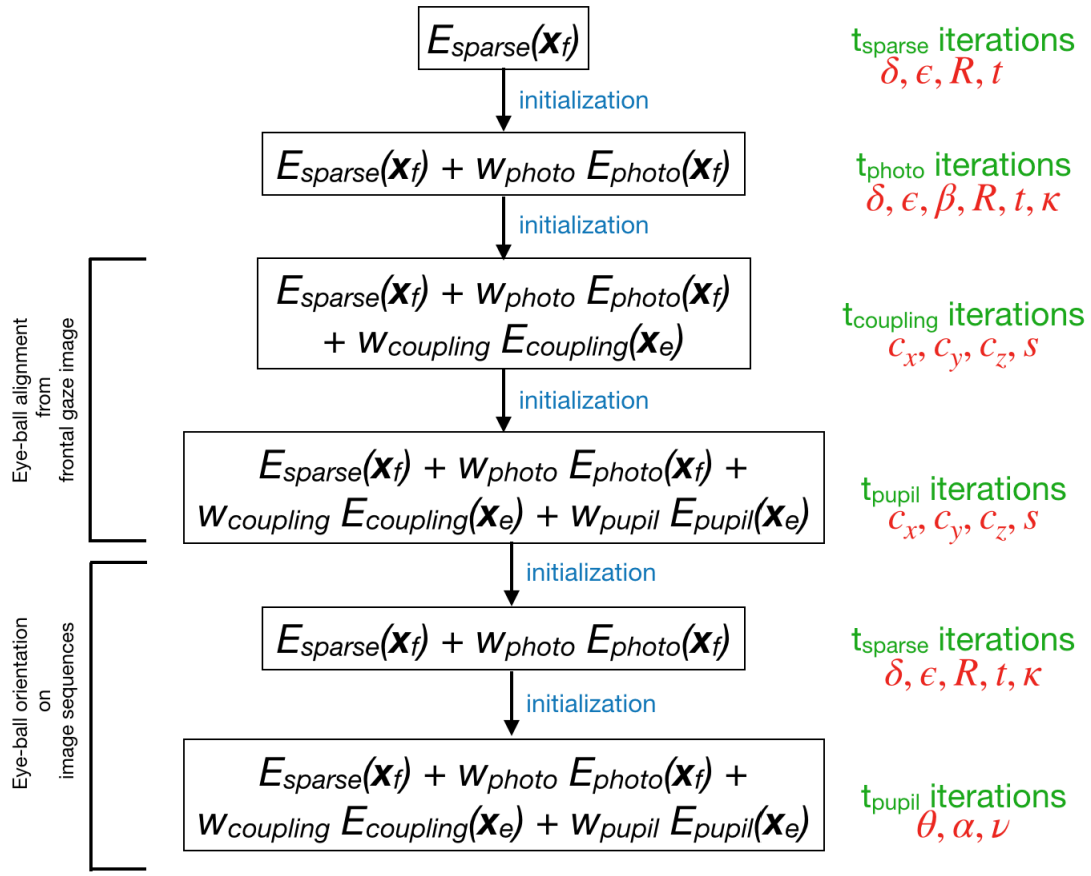


Figure 4.9: Our multi-step optimization strategy. Note that here we only show the data terms for brevity, data terms are accompanied by regularization terms where-ever stated above.

analytical derivatives for the formulated energy functions.

Chapter 5

Evaluation and results

In this chapter, we detail the qualitative and quantitative evaluations performed on the methodology for reconstruction explained in Chapter 4. We then present our results obtained, compare them to the state-of-the-art, and discuss inferences.

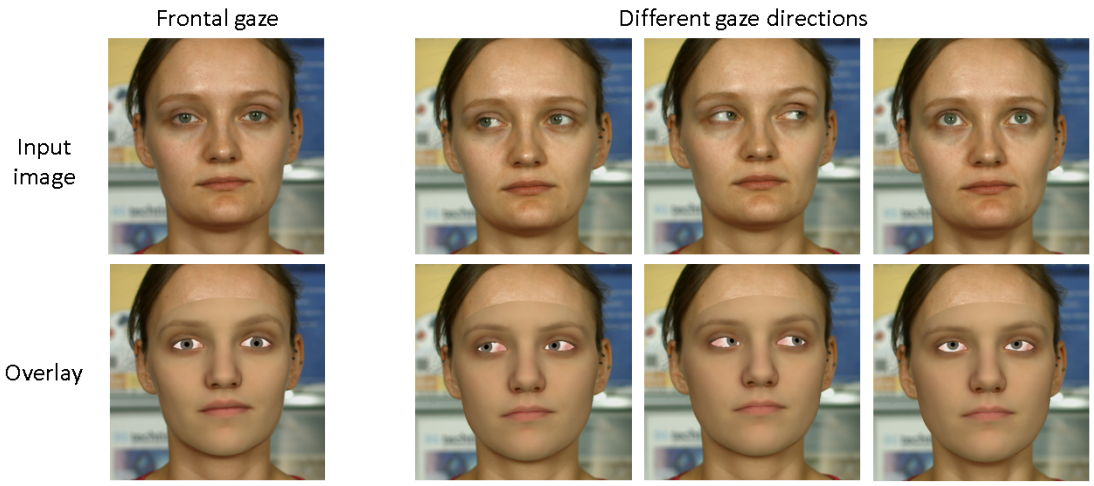
We begin by first qualitatively evaluating the output of the joint reconstruction. Thereafter, we delve into specific evaluations pertaining to the face reconstruction and eye-ball reconstruction individually.

5.1 Evaluation of joint reconstruction

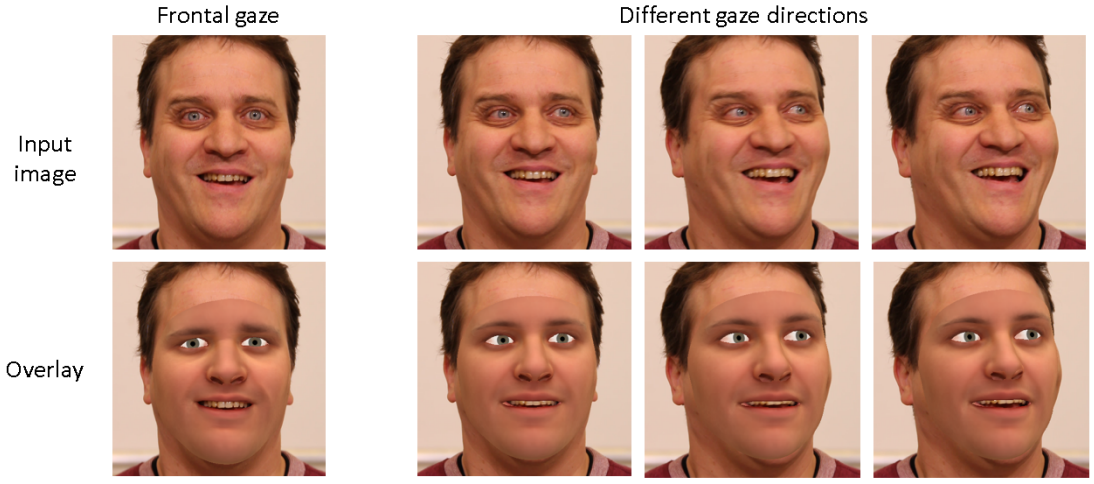
As described in the method in the previous chapter, we first calibrate for the eye-ball centers and radius for which we use a frontal gaze image, after reconstructing the face. At this stage, the E_{pupil} term is used to find these eye centers and radius, keeping the eye rotations fixed. Figure 5.1 shows the qualitative results of the estimated eye-balls fitted consistently behind the opening of the eye-lids in the reconstructed face mesh. In the figure, the result from the first calibration step is shown under the column for frontal gaze.

Once the initialization is done, as described in the method, we perform tracking on another image from the same sequence for a given input. The output for tracking three different eye-gaze directions is shown in the Figure 5.1. The optimized face and eye parameters from the calibration step are used as an initialization to reconstruct the face for the same person, where the pose, expression, and the eye-gaze of the person keeps varying. The eye-balls are allowed to rotate about the sphere center, to align to the detected pupil positions while minimizing the energy.

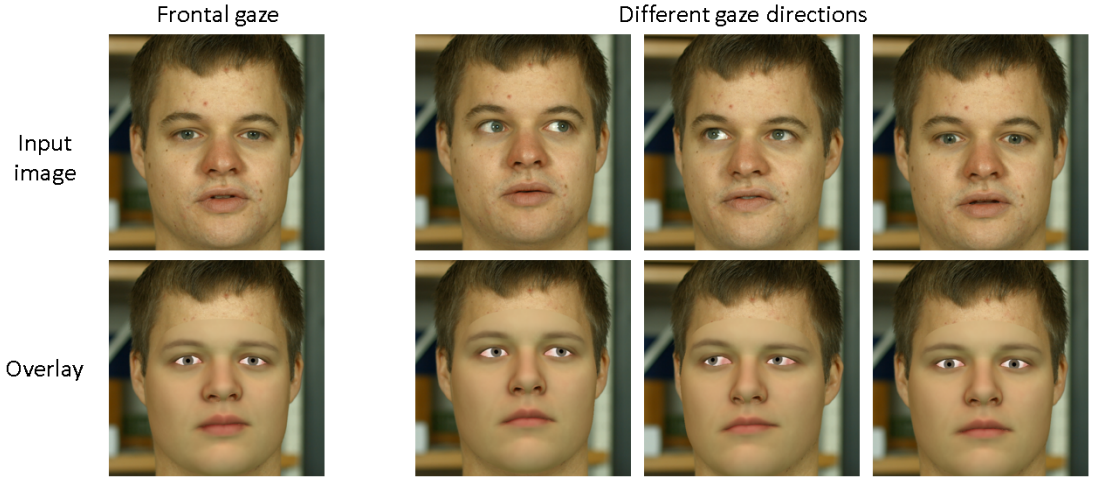
As seen in the figure, the predicted eye-balls rotate consistently without violating any anatomical constraints. This consistency is a very strong indicator of the



(a) Person 1



(b) Person 2



(c) Person 3

Figure 5.1: Qualitative results (on 3 people) of the fitted eye-balls along with the reconstructed face model.

importance of the multi-step approach to solve the non-linear and non-convex problem. We introduce energies one by one in an incremental fashion, initializing such that the result of the last smaller problem is used as an initialization for the next larger problem. In order to highlight this point, we performed a run while solving for all energies at once as well. Figure 5.2 shows that such an approach leads to the detected pupil landmarks being aligned in the end, but the eyes protrude out because of the wrong face geometry.

The following sections detail qualitative and quantitative evaluations for the reconstruction of faces and eyes individually.

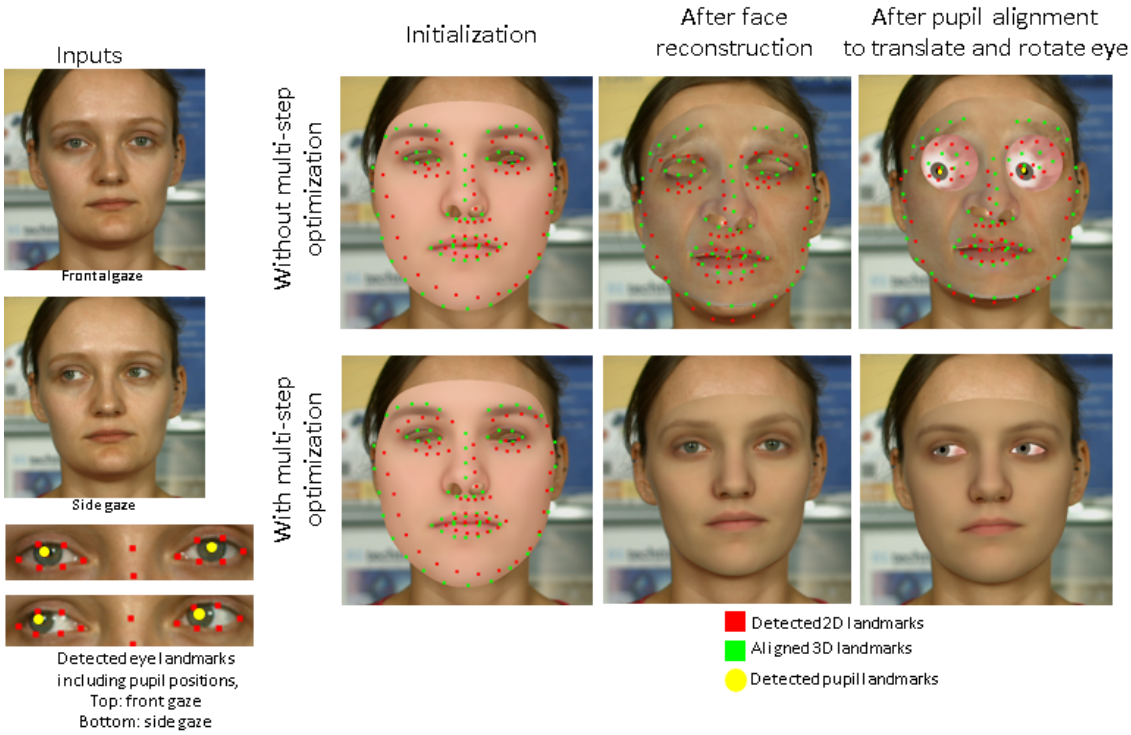


Figure 5.2: Importance of our multi-step approach to solve the optimization problem.

5.2 Evaluation of face reconstruction

This section details the qualitative and quantitative evaluations performed on the initial face reconstruction component of our methodology. While the camera intrinsics are assumed to be known, all other face parameters $\mathbf{x}_f$ are solved for.

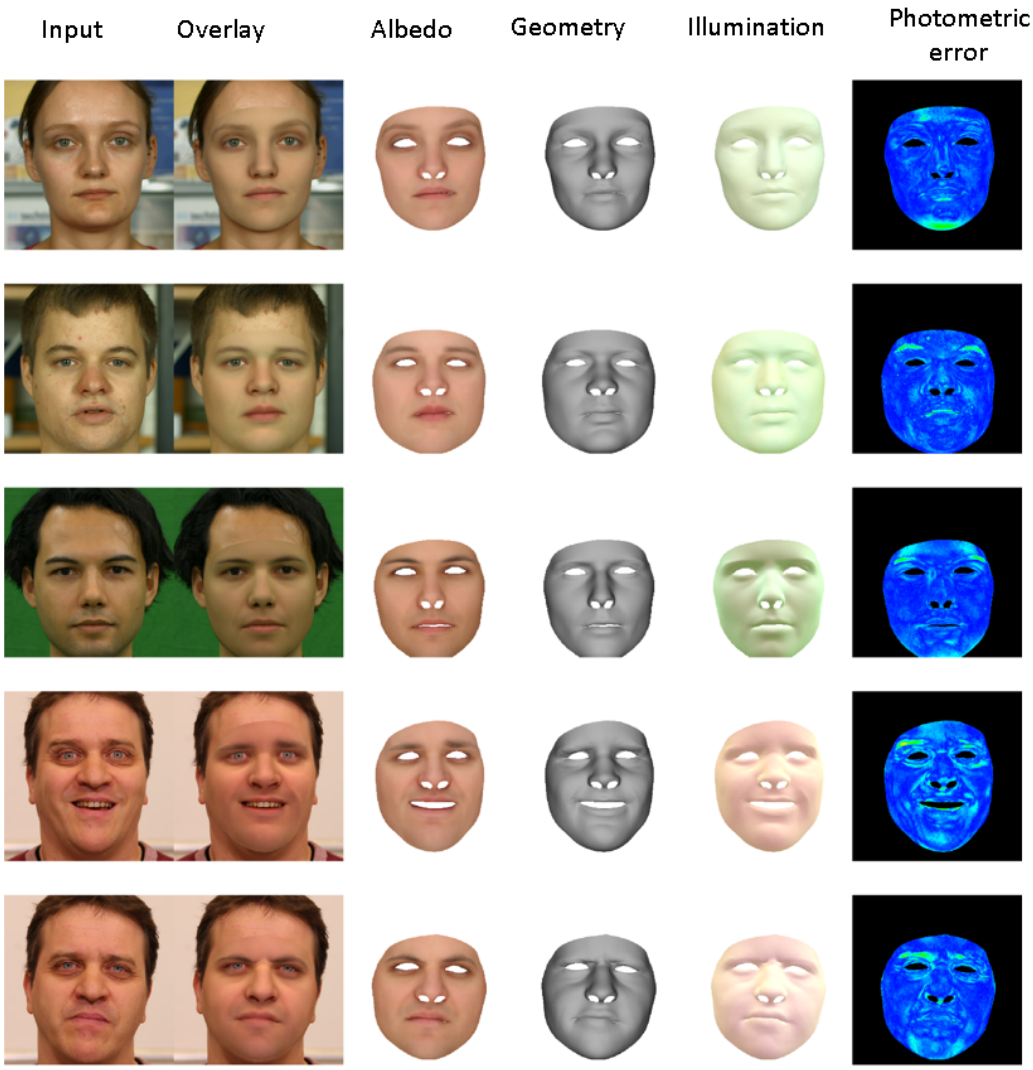


Figure 5.3: Our qualitative results on face reconstruction using a monocular RGB input image.

5.2.1 Qualitative evaluation

In Figure 5.3, we show the qualitative face reconstruction results on five input face images. The final reconstructed meshes can be viewed as the predicted overlays in the second column. The face color comes from albedo and illumination, which are decomposed and shown. Phong shading is applied to depict the face geometry separately in the figure. Reconstruction quality can be seen from the photometric error, which is the pixel-wise difference in RGB space between the projected reconstructed mesh and the input face image.

The first four rows depict different identities of people, demonstrating that our method generalizes well for multiple facial shapes. The last two rows capture different expressions for the same person. Note the consistency in the illumination and face albedo in the last two rows. These images were captured in similar conditions, and only the expressions of the person changed. However, the reconstructions of these images were obtained independently without this knowledge. These results capture variations in facial shape, expression and scene illumination, and demonstrate high-quality face reconstructions on the same.

To show the effect of individual face energy terms in an ablative sense, we show intermediate results between the optimizer iterations in Figures 5.4, 5.5.

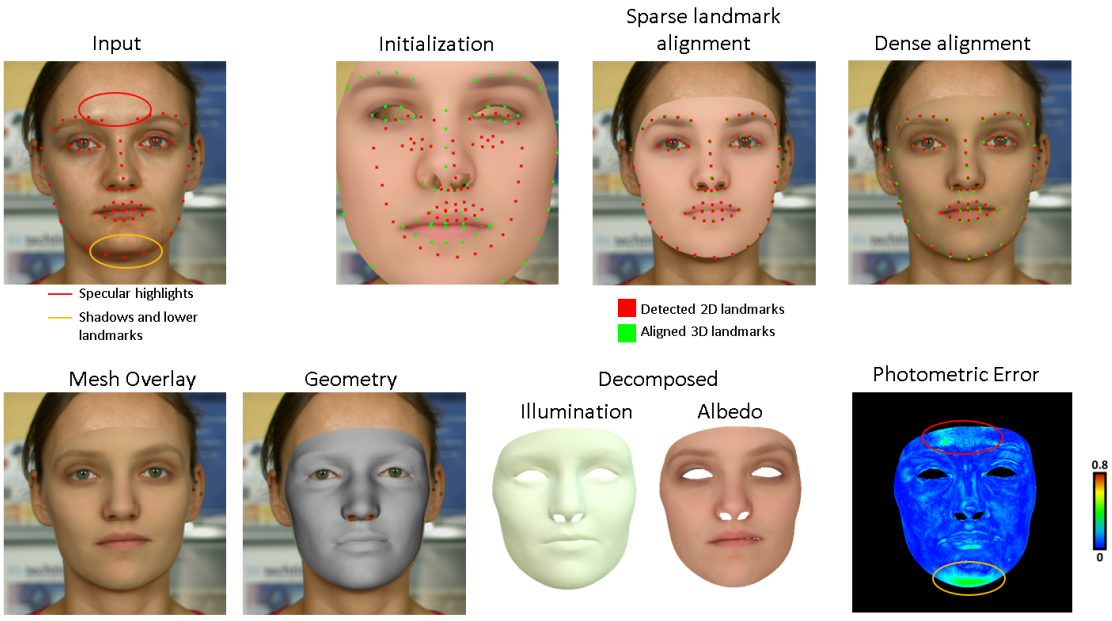


Figure 5.4: Intermediate qualitative results for face reconstruction between the optimizer iterations.

Figure 5.4 portrays the mesh alignment to the input image at various optimizer stages. The face is assumed to be approximately in the middle of the frame captured,

and hence the mesh is initialized such that it projects to the image center, as seen in the figure. The 3D mesh landmarks (in green) are aligned to detected 2D landmarks (in red) by solving for the face rigid pose as well as non-rigid facial shape and expression deformations using the sparse term $E_{sparse}(\mathbf{x}_f)$ for 30 iterations. Next, we add the dense photometric term $E_{photo}(\mathbf{x}_f)$ to further solve for face albedo and scene illumination for 100 iterations.

The result after dense alignment can be seen in the figure. As before, the final mesh overlay, geometry, decomposed illumination and albedo is shown. Since our illumination model assumes Lambertian material of the face, the specular highlights on the forehead create higher source of photometric error, circled in red. Furthermore, since the 2D landmarks on the chin are detected slightly too low, the face is fitted to the shadowed region, which is another source of photometric error, circled in yellow.

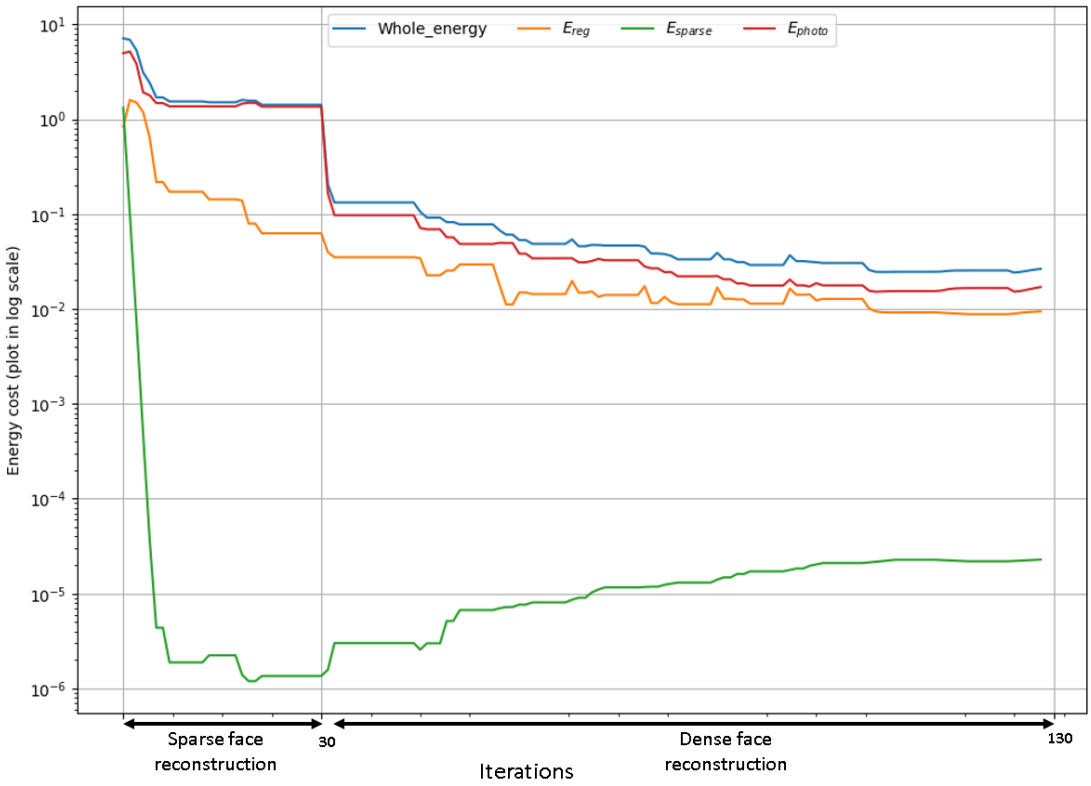


Figure 5.5: Energy costs of the optimizer iterations for face reconstruction.

Figure 5.5 shows the decrease in various energies (in log scale) of the optimizer iterations for the face reconstruction. We use a fixed number of iterations in all our experiments, as these energies consistently stopped decreasing up-to a small threshold beyond which changes in parameter values were negligible as seen in the

figure. For sparse face reconstruction, we minimize using E_{sparse} and E_{reg} which go down considerably in the first 30 iterations. Next we add and solve E_{photo} term, and this reduces the whole energy substantially.

Note that while the E_{sparse} term increases when E_{photo} is introduced, the whole energy still reduces, as the optimizer focuses more on dense vertex-to-pixel color alignment. This is run for 100 iterations, where we re-sample the vertices which have to be projected for the dense E_{photo} term at every 10^{th} iteration. This explains a slight increase in energy before it goes down again at every 10^{th} iteration starting from the 30^{th}.

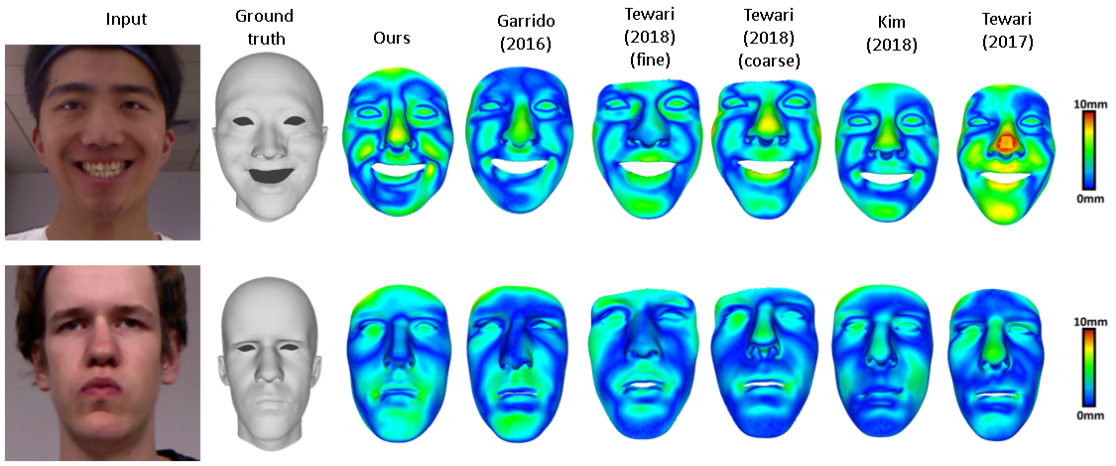


Figure 5.6: Two examples from the FaceWarehouse (Cao *et al.*, 2014) data-set for the geometry errors of the reconstructed meshes.

5.2.2 Quantitative evaluation

To quantify our method and compare our results with other methods which do face reconstruction, we performed evaluations on the FaceWarehouse (Cao *et al.*, 2014) data-set. It consists of 180 ground truth facial geometry meshes for respective monocular RGB images. Since it contains data from nine different people with varying facial shapes, each with 20 different expressions, it validates the accuracy of our method for reconstructing multiple facial shapes and expressions. The comparisons are done on the reconstructed geometry. The errors reported are the mean of corresponding vertex-to-vertex distances (in mm) between the reconstructed mesh vertices and the ground truth mesh vertices.

Since the given ground truth mesh had a different topology than the reconstructed one, we use a pre-computed set of dense correspondences between the topologies

of our reconstructions and the ground truth using a non-rigid refinement step, same as Tewari *et al.* (2018). Following the same protocol as Tewari *et al.* (2018), we then use these correspondences to align the meshes rigidly using Procrustes algorithm (Hurley & Cattell, 2007), and the error is computed as the mean closest point error.

Table 5.1 shows that our errors on the face geometry are comparable to the state-of-the art methods. For Garrido *et al.* (2016), the numbers correspond to their coarse-scale reconstructions. While Garrido *et al.* (2016) still slightly outperform our method, the difference is not significant and possibly even within the range of hyper-parameter tuning. Our results are very similar to the current learning-based approaches for face reconstruction which use the same linear 3DMM model, even outperforming Tewari *et al.* (2018) (coarse), Tewari *et al.* (2017), and Kim *et al.* (2018b). We come close to the quality of Tewari *et al.* (2018)(fine) even when they learn an additional corrective model on top of the 3DMM used.

Table 5.1: Reconstructed face geometry error comparisons on the FaceWarehouse dataset. Errors reported in mm.

	Ours	Garrido *et al.* (2016)	Tewari *et al.* (2018) (Fine)	Tewari *et al.* (2018) (Coarse)	Kim *et al.* (2018b)	Tewari *et al.* (2017)
Type	Optimization	Optimization	Learning	Learning	Learning	Learning
Mean	1.94	1.7	1.82	2.03	2.12	2.4
SD	0.37	0.27	0.37	0.52	0.42	0.62

5.3 Evaluation of eye-ball reconstruction

In this section, we evaluate the reconstructed and tracked eye-ball positions and orientations from a sequence of monocular RGB images. The qualitative evaluation performed on the eye-ball reconstruction is an ablative study of the algorithm similar to the previous section. The quantitative study, on the other hand, focuses on eye-gaze estimation and draws a detailed comparison to the current state-of-the-art regarding this problem.

5.3.1 Qualitative evaluation

Similar to face reconstruction, the effect of the individual eye energy terms in an ablative sense can be seen in Figures 5.7, 5.8 where we show intermediate results between the optimizer iterations.

Figure 5.7 visualizes the eye-ball positions and orientations after each step. Eye-balls positions are initialized by placing them behind the average face mesh before face reconstruction. Once we have solved for the face, we first couple the eye-ball

behind the eye-lids using $E_{coupling}$ term along with regularizers, while solving for the eye-size and eye-ball centers with respect to the face for 10 iterations.

This is not the best fit, as seen in the results after coupling, where the eyes-balls protrude outside the face mesh and the eye-ball centers are still off. But this gives a nice initialization for the next stage, where the E_{pupil} term is introduced, to further solve for eye-centers by aligning the detected and estimated pupil positions of the frontal gaze. The results after pupil alignment to frontal gaze can be seen in the 2^{nd} column from the right. Then for another image of the same person, we solve for eye-ball rotations using E_{pupil} for 10 iterations, while keeping the eye-centers and eye-size fixed.

The results after pupil alignment to rotate the eye-balls can be seen in the last column. Note that in general the detected pupil positions are not exactly at the center of the colored iris, as can been seen in the figure. This can introduce some errors as we depend on them to fit the eye-balls.

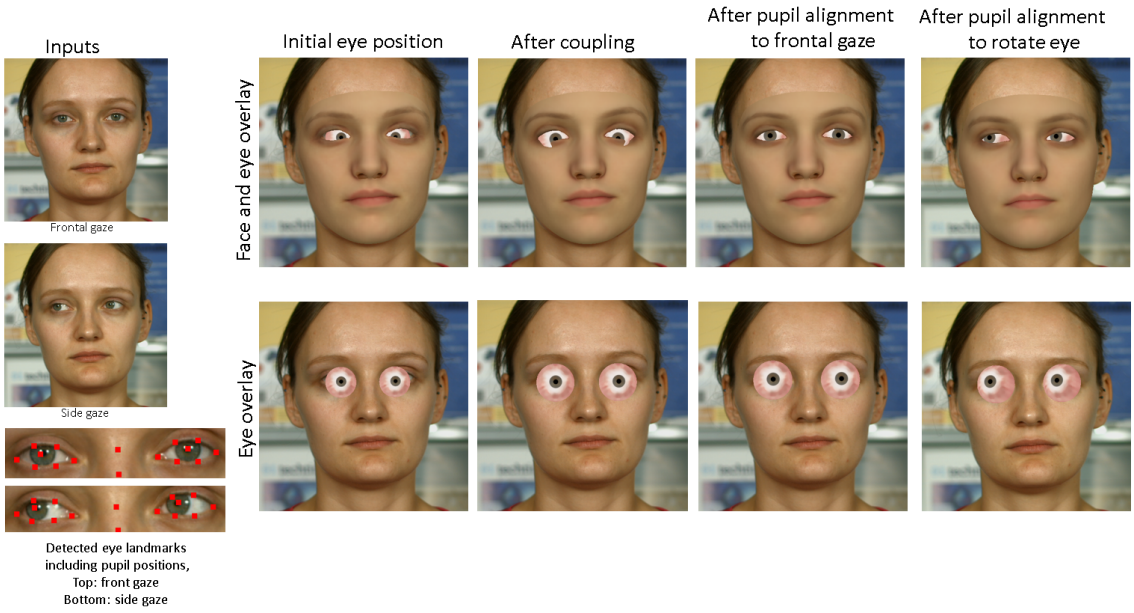


Figure 5.7: Intermediate qualitative results for eye-ball reconstruction between the optimizer iterations.

Figure 5.8 shows the decrease in various energies (in log scale) of the optimizer iterations for eye-ball reconstruction for a frontal-gaze image. As for the face, we use a fixed number of iterations in all our experiments, as these energies consistently stopped decreasing up to a small threshold beyond which changes in parameter values were negligible as seen in the figure. At first, we only solve using the $E_{coupling}$ term and the regularizer terms to couple the eye-balls behind the eye-lids for 10

iterations. Then the E_{pupil} term is added to the whole energy to align the estimated and detected pupil positions, while keeping the eye-rotations fixed. This is run for 10 more iterations.

Note that while the $E_{coupling}$ increases in these iterations, the whole energy goes down. As the reconstructed eye-lids geometry is not completely reliable to hold the eye-balls, the increase in $E_{coupling}$ is expected. But $E_{coupling}$ is still required to hold the eye-balls behind the eye-lids, without which the eye-ball centers would be free to translate in the z direction, while the pupil positions still align in 2D.

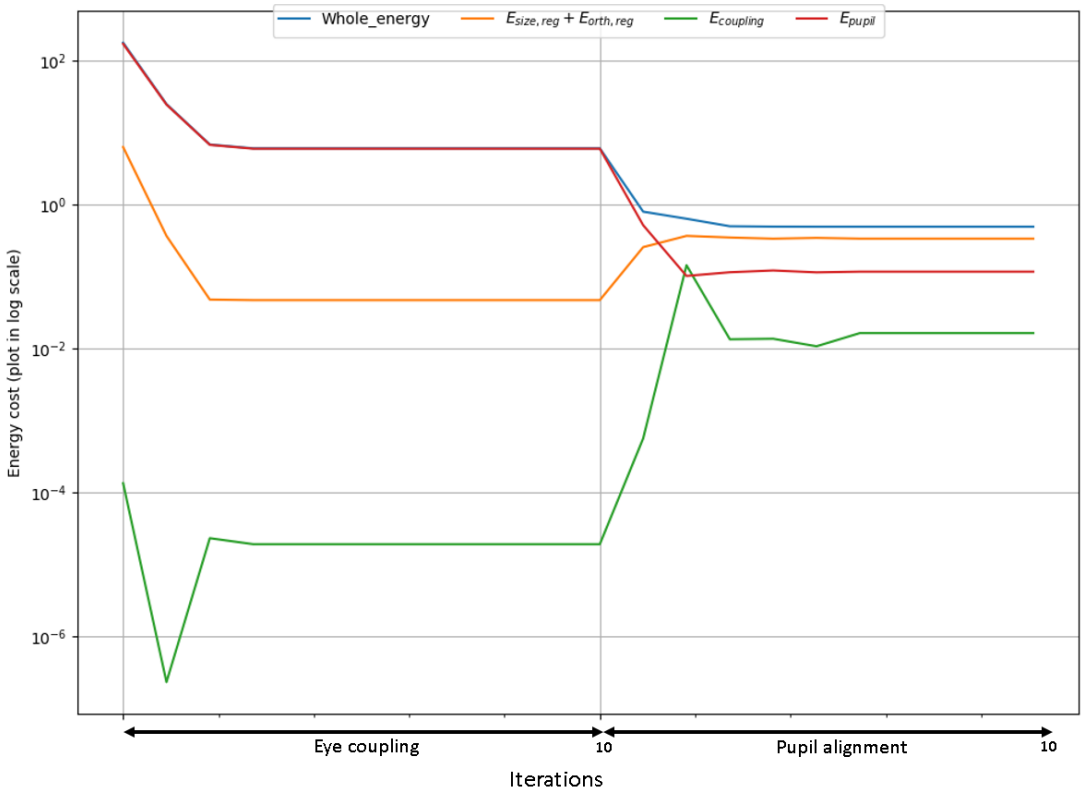


Figure 5.8: Energy costs of the optimizer iterations for eye-ball reconstruction.

5.3.2 Quantitative evaluation

As the eye-balls are rotated by aligning detected 2D pupil position and the mesh 3D pupil position, the predicted rotation gives us the eye-gaze direction. This eye-gaze direction can be estimated by rotating a unit vector pointing along the optical axis of the eye by the estimated rotation of each eye $R_{L|R}(\theta, \alpha, \nu)$, as described in Chapter 4, Table 4.2.

While our focus was on coupling the eye-balls to the face mesh, eye-gaze can be inferred from the eye-ball orientation, as we explained above. While there exist

more robust methods which entirely focus on estimating the eye-gaze directly, we provide some quantitative evaluations for 3D eye-gaze direction errors, to quantify the quality of the fitted eye-ball orientations. There is a lack of a ground truth data which directly provides face meshes and coupled eye-balls, as it is hard to capture hidden complete eye-balls directly, and non-intrusively using cameras. Thus, we use the estimated eye-gaze as a quantitative metric to assess the fidelity of our eye-ball reconstruction.

In the rest of this subsection, we explain the evaluation in detail, provide an overview of the data-set used, and discuss the results along with relevant inferences.

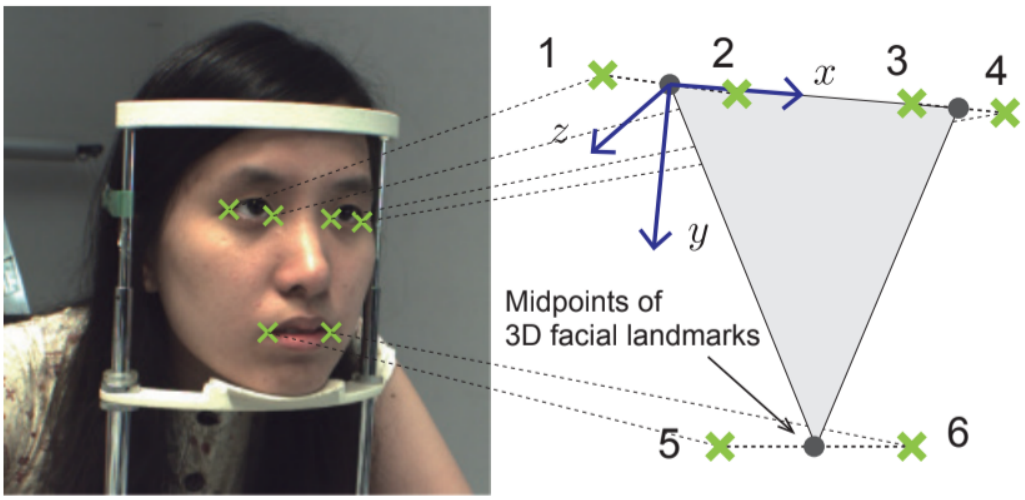


Figure 5.9: Set-up for UTMultiView (Sugano *et al.*, 2014) data-set. A person's head was held stationary while 160 different gaze directions were recorded. The green crosses depict the 3D positions of six facial landmarks recorded.

Evaluations are conducted on the popular UTMultiView (Sugano *et al.*, 2014) data-set, which provides ground truth 3D gaze directions for monocular RGB images. The UTMultiView data-set contains monocular RGB face images of 50 people captured from eight different camera views, which essentially also capture side views of faces along with frontal views. 160 different 3D gaze directions were recorded for each person by making the person look at a known point on a target screen, as seen in Figure 3.10, in Chapter 3. The gaze direction is the line joining the eye-corner center and the target screen position.

This data-set provides 3D positions of six facial landmark positions in their world coordinate system as seen in Figure 5.9. These were used to find a rigid transformation to align these points to our estimated 3D facial landmarks using the

Top left	Top	Top right
Left	Frontal	Right
Bottom left	Bottom	Bottom right

Figure 5.10: Nine gaze clusters for the ground truth gaze directions for each person. Since the gaze directions were collected by making people look at a target point on a screen, this can be seen as the nine clusters on the target screen at which people looked at.

Procrustes algorithm. This transformation was used to transform their ground truth eye-gaze direction to our face coordinate system, in which we ran our evaluations.

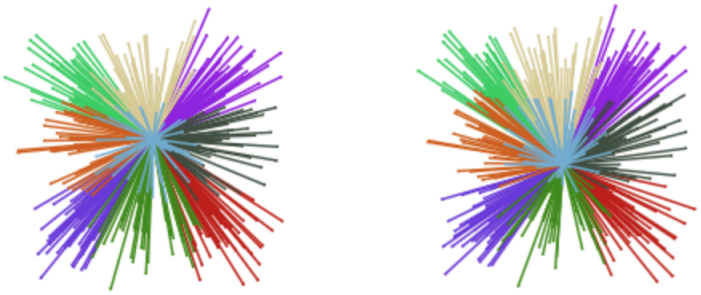


Figure 5.11: Left and right ground truth eye-gaze directions for nine clusters. It depicts one arrow per person per gaze. Each arrow is a 3D unit vector projected and seen from top.

We selected a subset of images for each person by clustering the gaze directions into nine gaze clusters as in Figure 5.10. The Figure 5.11 shows the distribution of gaze directions chosen for each person, which contains one eye-gaze direction per person per gaze. Each color represents a different gaze cluster, and the direction for a particular cluster for a person was chosen randomly from all the directions which

fell in that cluster.

The eye-gaze error is the angle formed between the predicted and ground truth 3D eye-gaze directions. Note that these gaze directions are independent of the face rotations, since they are calculated in the face model space. Hence the comparison is consistent across multiple people with different face poses, and it is decoupled from the face pose.

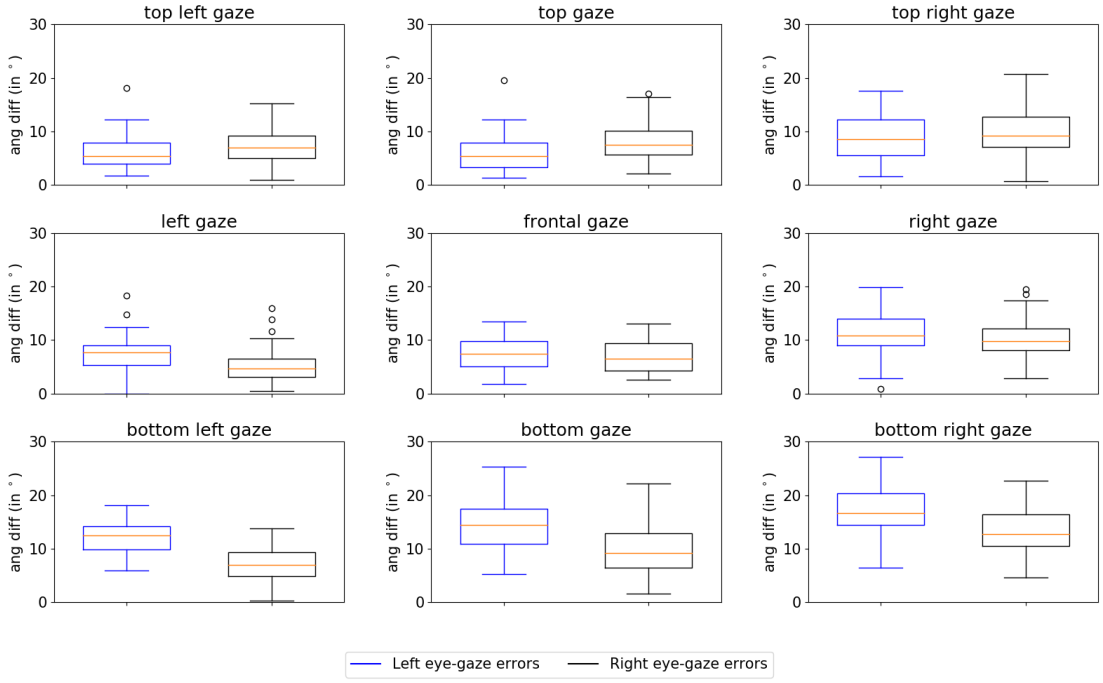


Figure 5.12: Box plots for each eye-gaze cluster. Our errors reported in degrees.

Since the multi-view camera setup provides multiple angles of the face, we show our evaluations using one of the views of the cameras which provides a frontal view of the person. Nine eye-gaze directions are chosen from each of the nine eye-gaze clusters. A frontal gaze image is chosen for th calibration step, where the ground truth eye-gaze vector is closest to a frontal-vector in the eye model space. The mean, median and standard deviations (sd) for eye-gaze errors are reported in Table 5.2.

Individual region-specific box plots for the same eye-gaze errors on all the 50 people can be seen in the Figure 5.12. Errors are reported for both the eyes, and are close to around 10° for each region. Figure 5.15 shows the ground truth gaze (red) and predicted gaze (green) for one person, which are quite close.

Figure 5.13 visualizes person-specific ground truth and predicted left and right eye-gaze directions and errors for one person, selected randomly from the same evaluation data-set. Nine gaze directions for each cluster are shown, and color

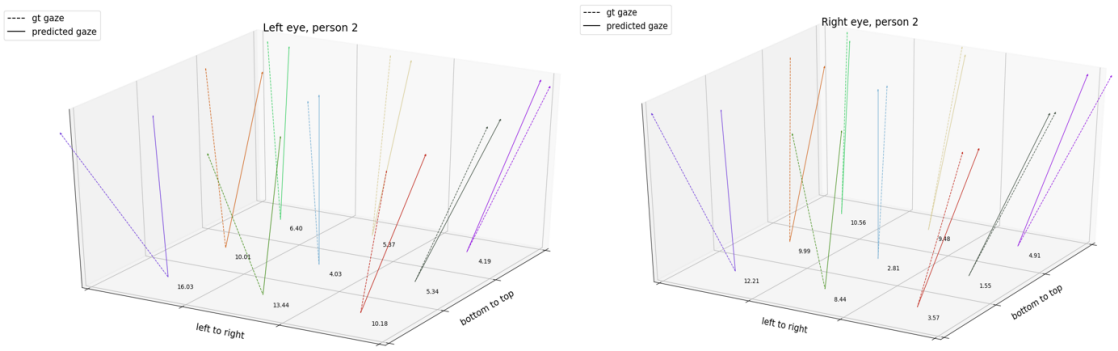


Figure 5.13: Person specific ground truth and predicted left and right eye-gaze directions and errors. Nine gaze directions, one for each eye-gaze cluster. The gaze errors are reported in degrees.

Table 5.2: Eye-gaze errors (in degrees) on UTMultiView data-set

	left eye	right eye	both eyes	frontal gaze	top gaze	bottom gaze
mean	10.11	8.71	9.41	8.08	7.70	12.45
median	9.62	7.76	8.54	7.71	7.29	12.44
sd	5.32	4.48	4.96	4.02	4.01	5.27

coded according to Figure 5.11.

The upper eye-lid is capable of more articulation. This leads to most of the eye being covered when the gaze is directed downwards and thus the gaze estimation in such cases is prone to higher degrees of error as evident from the plot for bottom gaze, in Figure 5.12. Table 5.2 also summarizes errors for top, frontal, and bottom gaze. There is about 4°-5° between the bottom and top/frontal gaze. This is because the 2D pupil detected is more un-reliable, as it depends on the features from the partially-visible circular iris, which gets hidden more by the upper eye-lid for a bottom gaze. This does not occur in the case of top/frontal gaze as the lower eye-lid is relatively stationary while the upper eye-lid moves upward to reveal the eyes.

A similar trend can be seen in Figure 5.14 which shows cumulative error distribution (CED) for each cluster. A smaller area is covered under the plots for bottom gaze. A larger area under the curve (AUC) implies better prediction and lesser eye-gaze error.

To evaluate our results on eye-gaze prediction, we use make use of *UTMultiView* data-set (Sugano *et al.*, 2014). This and other data-sets which are used by the methods we compare to are described below:

1. UTMultiView data-set (Sugano *et al.*, 2014) — provides most real eye move-

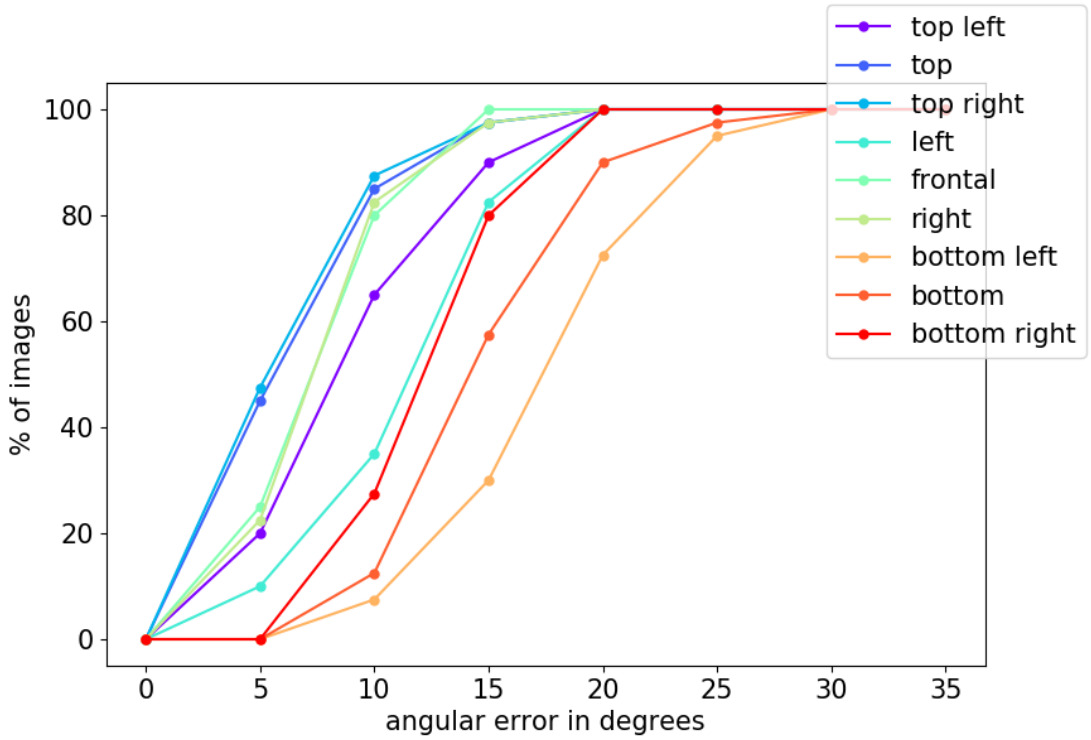


Figure 5.14: Cumulative error distribution (CED) curve on eye-gaze errors for different eye-gaze clusters. The x-axis depicts the angular error in degrees while the y-axis depicts the percentage of images where the error is smaller than the eye-gaze error. A larger area under the curve (AUC) implies better prediction and lesser eye-gaze error.

ments for faces. 50 participants were constrained by head-clamped device with similar scene lighting for all participants.

2. Columbia dataset (Smith *et al.*, 2013) — Like UTMultiView, 56 participants were constrained by head-clamped device with similar scene lighting for all participants. Results on this data-set can be compared to ones obtained on UTMultiView, as they have been captured with similar constrains.

3. EyeDiap data-set (Funes Mora *et al.*, 2014) — Eye movements for faces from 16 participants who where not constrained by head-clamped device. Similar scene lighting for all participants. A harder data-set that the above two, as the head-pose varies.

4. SynthEyes data-set (Wood *et al.*, 2015) — Provides synthetically rendered single eye-regions which capture more variations in lighting and facial appearance.

5. MPIIgaze (Zhang *et al.*, 2018) — A very challenging real image data-set which

Figure 5.15: Our qualitative eye-gaze results on UTMultiView.

captures multiple users with various head poses, eye-gaze directions and lighting variations.

Table 5.3: Comparisons for eye-gaze estimation errors (in degrees)

Method	Method type	Mean	Median	Tested on data-set
Ours	optimization-based	9.42	8.54	UTMultiView
Wood *et al.* (2016a)	optimization-based	8.87	7.54	Columbia
Wood *et al.* (2016a)	optimization-based	9.44	8.63	EyeDiap
Sugano *et al.* (2014)	learning-based(CNN)	6.5	-	UTMultiView (trained on the same)
Wood *et al.* (2015)	learning-based(CNN)	11.12	-	UTMultiView (trained on synthEyes)
Zhang *et al.* (2015)	learning-based(CNN)	13.9	-	MPIIGaze (trained on UTMultiView)
Zhang *et al.* (2015)	learning-based(CNN)	10.5	-	EyeDiap (trained on UTMultiView)
Zhang *et al.* (2018)	learning-based(CNN)	9.8	-	MPIIGaze (trained on UTMultiView)

Table 5.3 lists the eye-gaze errors for various methods. We get comparable results for the estimated gaze-error, while our method works on faces without requiring any explicit eye-gaze training data. The eye-gaze extracted is also decoupled from the face pose, which is not possible for some learning-based approaches like Wood *et al.* (2016b).

To compare our approach to learning-based methods, we chose errors reported on cross data-set and cross-person settings, as learning-based methods can give very low errors if trained on the same data-set and they may be biased towards the type of variations in a data-set and might not generalize well.

Note that MPIIGaze data-set captures a lot of variations in person identities, head pose and illumination conditions, so our error values are not directly comparable when the test set is MPIIGaze. This is evident from the errors reported by Zhang *et al.* (2015), which is higher on MPIIGaze than EyeDiap data-set.

Wood *et al.* (2016a) is a model-fitting optimization-based method, which is tested on two data-sets Columbia and EyeDiap (described in Chapter 3, Section 3.3). Since these data-sets have similar characteristics as UTMultiView, the eye-gaze errors are roughly comparable. Both ours and Wood *et al.* (2016a) are optimization-based methods which are not dependent on a person, lighting conditions, and are head pose invariant.

Our method computes eye-gaze directions for both eyes together, while Wood *et al.* (2016a) computes them on one eye at a time. We also reconstruct the full face mesh of the person including various expressions. While we require a frontal gaze image to find eye-centers and then compute eye-gaze directions for further images, Wood *et al.* (2016a)'s method directly computes the eye-gaze direction from a single image. This was possible as the eye-center position and the eye-ball size was already a part of their eye-region morphable model, while we had to couple the eye-ball positions while jointly optimizing for the face and the eyes.

Chapter 6

Conclusion and outlook

This chapter summarizes the work of this thesis. Further on, it also delves into some limitations of this work along with a discussion about future avenues of research on this topic.

6.1 Conclusion

In this thesis, we proposed a novel optimization-based approach to reconstruct and track a 3D face mesh along with coupled eye-balls from monocular RGB images. We demonstrated that it is possible *couple* eye-balls to existing face morphable models directly, while fitting them to monocular images in an analysis-by-synthesis approach.

At first, a frontal gaze image from the image sequence was used to calibrate for eye-ball positions and size. Along with this, rigid face pose, non-rigid facial shape and expression deformations, and facial color formed by albedo and scene illumination were recovered. On the rest of the images, tracking was performed to recover eye-ball rotations and changing face pose and expressions.

A non-linear and non-convex problem was formed and the energy terms were formulated in a least-squares sense. This was then solved by a multi-step process. Energy terms were introduced one-by-one in an incremental fashion, such that the result of the last smaller problem was used as an initialization for the next larger problem.

Finally, we demonstrated our results by qualitative and quantitative analysis. We obtained high-quality face reconstructions comparable to the state-of-the-art. We further showed qualitatively that the eye-balls consistently fit behind the eye-lids and rotate in a natural manner to follow the eye-gaze. To quantify our eye reconstruction,

we evaluated on the eye-gaze retrieved from the eye-ball orientations. Our eye-gaze errors are comparable to the state-of-the art methods operating in similar conditions, even when we did not directly set to find eye-gaze from images. We also saw that the design of the multi-step approach was crucial to solve the highly non-convex problem, without which the optimizer gets stuck in local minima.

6.2 Limitations and outlook

The problem of reconstructing faces and eyes directly from monocular images is a very challenging and ill-problem problem. While we started with a challenging scenario of using monocular images inspired by lightweight methods, multi-view images have not been used to reconstruct coupled eye-balls. These could give more robust results as there is less depth ambiguity.

We largely depend on the facial deformations of the learned 3DMM model we use. This introduces an ethnic bias on the structure of the face recovered (seen in Figure 5.15), as the used 3DMM was formed from Caucasian face scans, while both our face and eye evaluation sets are mostly on Asian faces. This specially effects the quality of the eye region. Moreover, this learned 3DMM only encodes linear deformations. Recent work by Tewari *et al.* (2018), Tran & Liu (2018) have incorporated additional non-linear correctives on top of this 3DMM used to get better quality results. They train with in-the-wild-images which can remove this bias by learning from more diverse faces.

While we get low eye-gaze errors, comparable to the state-of-the art methods performing under similar conditions, we elaborate on some sources of eye-gaze error. Contrary to the methods which directly regress for the eye-gaze direction, our method focuses on coupling the eye-balls consistently behind the eye-lids, solving for eye-ball centers and orientations. This is a more restrictive setting, where the optimizer focuses on mesh reconstruction while the eye-gaze is an implicit output.

Our eye model also has some simplified assumptions, one of which ignores the angle difference between the optical axis and the visual axis. This can introduce an error up to 5° (Majid *et al.*, 2013). Most methods solve only for one eye-gaze direction at a time, while we solve for both the eyes together. The optimizer best fits both the eyes jointly for natural plausible rotations. If the eye center is not positioned rightly for one of the eyes, this can introduce more error. Finally, we depend on the detected pupil positions, a wrong detection because of strong eye specularities can introduce error in the fitting, and further in the predicted eye-gaze direction.

Methods which recover the eye centers with respect to the face often involve a person-specific calibration step, wherein they make people gaze at different directions, keeping the head still. We simplify this by an easier calibration step which just requires a frontal gaze image of the person. This calibration image is easy to obtain in any image sequence and can work on in-the-wild videos. The detection of this frontal gaze image can easily be automated. This can be done by training a classifier (Thies *et al.*, 2016) on images to identify if the person is looking in front or not. Or by using an eye-gaze tracker to detect a frontal eye-gaze.

The eye centers are estimated at the position which is explained by our energy terms, and may not be anatomically correct. However, we demonstrate that these estimated centers and rotations result in eye-gaze predictions which perform similarly compared to other eye-gaze estimation methods.

While we solve for eye-ball pose and geometry, the energy can be extended to solve for the eye-ball appearance. Similar to the dense face reconstruction (as defined in Chapter 4, Equation 4.17), a dense eye reconstruction term $E_{e,photo}$ (like in Wood *et al.* (2016a)) can be used to retrieve eye model parameters for eye-ball appearance:

$$E_{e,photo}(\mathbf{x}_e) = \frac{1}{N_{visible}} \sum_{i=1}^{N_{visible}} \rho(I(\mathbf{p}_{e,i}(\mathbf{x}_e)) - \mathbf{c}_{e,i}(\mathbf{x}_e)) \tag{6.1}$$

where $\mathbf{c}_{e,i} \coloneqq \mathbf{c}_{e,i}(\boldsymbol{\tau}, \boldsymbol{\kappa})$ represents eye-ball vertex color, and $\mathbf{p}_{e,i} \coloneqq \mathbf{p}_{e,i}(s, s_{iris}, s_{pupil})$. Since the eye-ball is partially visible, the $N_{visible}$ vertices which project on to the convex polygon formed by landmarks around the eye-lids, can be considered. This will also give use better alignment, correct iris and pupil size (s_{iris}, s_{pupil})in addition to the appearance information.

Other circle fitting approaches like circular hough transform (Xie & Ji, 2002), integro-differential operator (Daugman, 1993), RANSAC model fitting (Zhang *et al.*, 2010; Wood & Bulling, 2014) could be directly used to align eye-model iris area to the iris area on the image, after segmenting this area.

The optimization problem is formulated as a set of energy functions which have to be minimized to solve for a set of parameters. We used *Ceres* (Agarwal & Mierle, 2012), a C++ library for modeling and solving large and complex non-linear problems on the CPU, as a proof of concept. However, the nature of the problem is inherently suited for parallel computation as it is formulated on thousands of vertices of a 3D mesh which are independently deformed. This leaves a lot of room to explore the option of a GPU based approach for these highly parallel computations.

Appendix A

Computational details

A.1 Transformations in euclidean space

A.1.1 Rotation in euclidean space

A rotation vector $\mathbf{r} = (V, H, T) \in \mathbb{R}^3$ is used to express three angles in radians which describe the amount of rotation in a 3D euclidean space about origin. To rotate a point $\mathbf{X} = (x, y, z) \in \mathbb{R}^3$ about origin by an angle V about x-axis, H about y-axis, T about z-axis in a right-handed system, rotation matrices are used.

Let,

$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(V) & \sin(V) \\ 0 & -\sin(V) & \cos(V) \end{bmatrix}$$

$$R_y = \begin{bmatrix} \cos(H) & 0 & -\sin(H) \\ 0 & 1 & 0 \\ \sin(H) & 0 & \cos(H) \end{bmatrix}$$

$$R_z = \begin{bmatrix} \cos(T) & \sin(T) & 0 \\ -\sin(T) & \cos(T) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The rotation matrix which describes rotation about z-axis, then y-axis and then x-axis is given by $R = R_x R_y R_z$ where each element in R is given by:

$$R_{11} = \cos(H)\cos(T)$$

$$R_{12} = \cos(H)\sin(T)$$

$$R_{13} = -\sin(H)$$

$$R_{21} = \sin(V)\sin(H)\cos(T) - \cos(V)\sin(T)$$

$$R_{22} = \sin(V)\sin(H)\sin(T) + \cos(V)\cos(T)$$

$$R_{23} = \cos(H)\sin(V)$$

$$R_{31} = \cos(V)\sin(H)\cos(T) + \sin(V)\sin(T)$$

$$R_{32} = \cos(V)\sin(H)\sin(T) - \sin(V)\cos(T)$$

$$R_{33} = \cos(H)\cos(V)$$

Hence position of the rotated point can be seen as a simple matrix-vector multiplication $\mathbf{X_{rotated}} = R\mathbf{X}$. The rotation R forms a special orthogonal group:

$$SO(3) \coloneqq \{R \in \mathbb{R}^{3\times 3} : det(R) = \pm 1 \wedge RR^T = \mathbf{1}\}$$

A.1.2 Translation in euclidean space

$\mathbf{t} = (t_x, t_y, t_z) \in \mathbb{R}^3$ represents translation which changes the position of a point.

A.2 Partial derivatives

A.2.1 Multivariate gradient vector

Gradient $\nabla r(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}^n$ over a vector $\mathbf{x} \in \mathbb{R}^n$ with respect to a function $r(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$ is defined as

$$\nabla r(\mathbf{x}) = \left(\frac{\partial r(\mathbf{x})}{\partial \mathbf{x}_1}, \frac{\partial r(\mathbf{x})}{\partial \mathbf{x}_2}, ..., \frac{\partial r(\mathbf{x})}{\partial \mathbf{x}_n} \right)^\top \tag{A.1}$$

where $\frac{\partial r(\mathbf{x})}{\partial \mathbf{x}_i}$ are the partial derivatives of r with respect to $\mathbf{x}_i$.

A.2.2 Jacobian matrix

If a vector valued function $\mathbf{r}(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}^m$, which takes as input vector $\mathbf{x} \in \mathbb{R}^n$ and produces output as vector $\mathbf{r}(\mathbf{x}) \in \mathbb{R}^m$, then it's Jacobian is a matrix $\mathbf{J}_r \in \mathbb{R}^{m \times n}$ of first order partial derivatives, such that

$$\mathbf{J}_r(\mathbf{x}) = \begin{pmatrix} \frac{\partial r_1(\mathbf{x})}{\partial x_1} & \frac{\partial r_1(\mathbf{x})}{\partial x_2} & \cdots & \frac{\partial r_1(\mathbf{x})}{\partial x_n} \\ \frac{\partial r_2(\mathbf{x})}{\partial x_1} & \frac{\partial r_2(\mathbf{x})}{\partial x_2} & \cdots & \frac{\partial r_2(\mathbf{x})}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial r_m(\mathbf{x})}{\partial x_1} & \frac{\partial r_m(\mathbf{x})}{\partial x_2} & \cdots & \frac{\partial r_m(\mathbf{x})}{\partial x_n} \end{pmatrix} \tag{A.2}$$

A.2.3 Hessian matrix

If a scalar function $r(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$, which takes as input vector $\mathbf{x} \in \mathbb{R}^n$ and produces output as scalar $r(\mathbf{x}) \in \mathbb{R}$, then Hessian $\mathbf{H}_r \in \mathbb{R}^{m \times n}$ is a square matrix made up of second order partial derivatives, such that

$$\mathbf{H}_r(\mathbf{x}) = \begin{pmatrix} \frac{\partial^2 r(\mathbf{x})}{\partial x_1^2} & \frac{\partial^2 r(\mathbf{x})}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 r(\mathbf{x})}{\partial x_1 \partial x_n} \\ \frac{\partial^2 r(\mathbf{x})}{\partial x_2 \partial x_1} & \frac{\partial^2 r(\mathbf{x})}{\partial x_2^2} & \cdots & \frac{\partial^2 r(\mathbf{x})}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 r(\mathbf{x})}{\partial x_n \partial x_1} & \frac{\partial^2 r(\mathbf{x})}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 r(\mathbf{x})}{\partial x_n^2} \end{pmatrix} \tag{A.3}$$

A Hessian matrix can also be written as $\mathbf{H}_r(\mathbf{x}) = \mathbf{J}_r(\nabla r(\mathbf{x}))^\top$.

A.3 Normals

A.3.1 One-ring neighbourhood normals

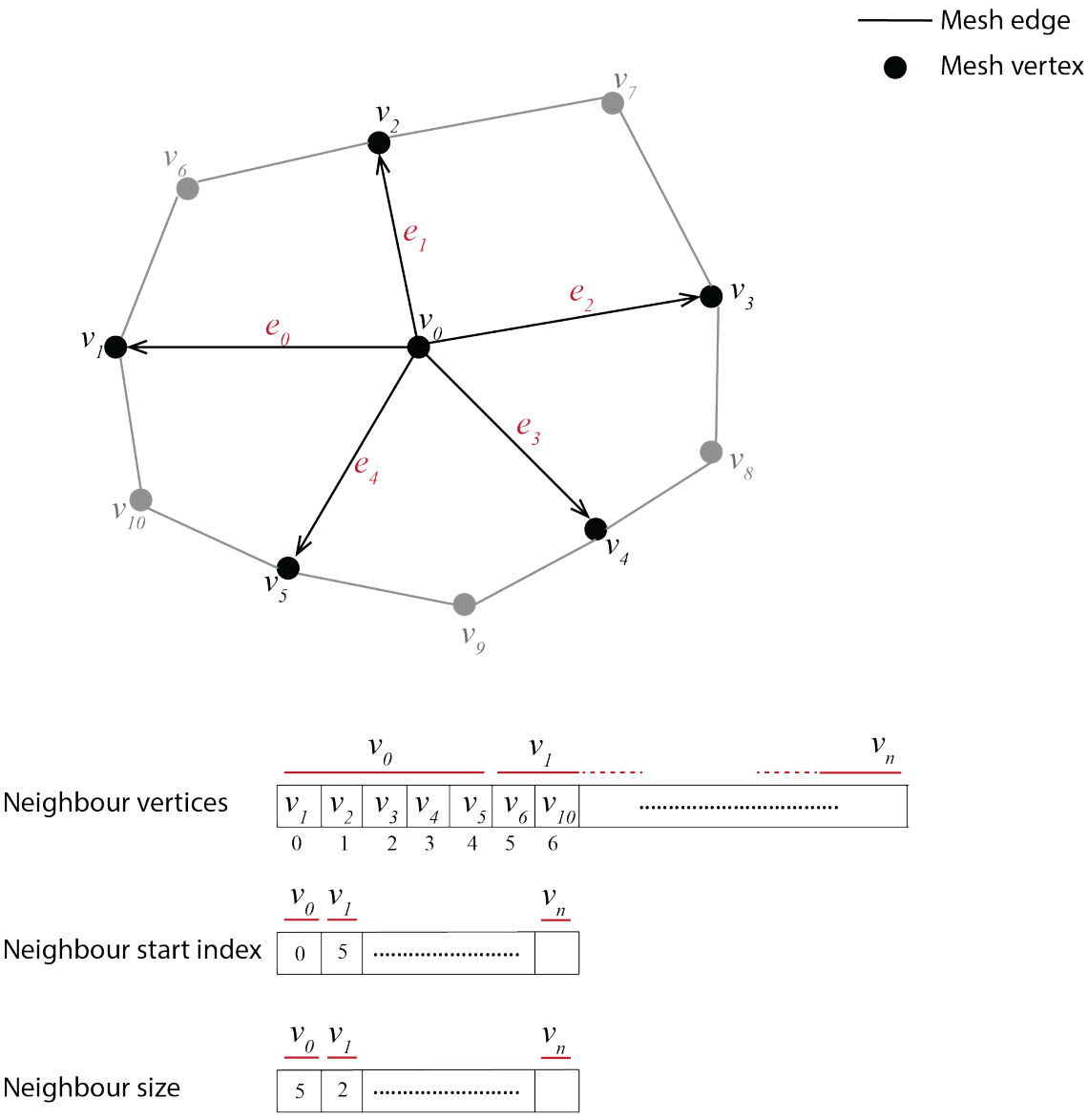


Figure A.1: Mesh's neighbour vertices and data-structures for normal computation using 1-ring neighborhood.

Let the vertex at index i in the mesh be $\mathbf{v}_i$, and its corresponding normal calculated using 1-ring neighbourhood normals be $\hat{\mathbf{n}}_i$. Then given neighbour vertex indices vector $\mathbf{N}_v$, neighbour start indices vector $\mathbf{N}_{start}$ and neighbour size vector $\mathbf{N}_{size}$, as seen in Figure A.1, we can calculate normals $\hat{\mathbf{n}}_i$ using Algorithm 1.

Algorithm 1 1-ring neighbourhood normals

Input: $\mathbf{v}_1 \dots \mathbf{v}_n; \mathbf{N}_v; \mathbf{N}_{start}; \mathbf{N}_{size}$
Output: $\hat{\mathbf{n}}_1 \dots \hat{\mathbf{n}}_n$

for $k \leftarrow 0$ to $n-1$ **do**
  $\mathbf{n}_t \leftarrow \mathbf{0}$
  $\mathbf{v} \leftarrow \mathbf{v}_{\mathbf{N}_v[k]}$
  for $i \leftarrow 0$ to $N_{size}[k]-1$ **do**
    $i1 \leftarrow i + \mathbf{N}_{start}[k]$
    $i2 \leftarrow (i+1)\%\mathbf{N}_{size}[k] + \mathbf{N}_{start}[k]$
    $\mathbf{n}_t \leftarrow \mathbf{n}_t + (\mathbf{v}_{i1} - \mathbf{v}) \times (\mathbf{v}_{i2} - \mathbf{v})$ ▷ area weighted normals
  end for
  $\mathbf{n}_k \leftarrow \frac{\mathbf{n}_t}{\mathbf{N}_{size}[k]}$
  $\hat{\mathbf{n}}_k \leftarrow \frac{\mathbf{n}_k}{\|\mathbf{n}_k\|_2}$
end for
return $\hat{\mathbf{n}}_1 \dots \hat{\mathbf{n}}_n$

A.3.2 Flat-sphere normals

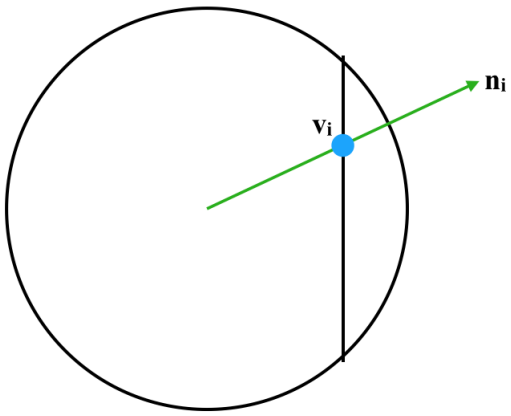


Figure A.2: Flat sphere's normal vector $\mathbf{n}_i$ at vertex $\mathbf{v}_i$

To avoid shading artifacts at the junction where a sphere is flattened, the normals of a flattened sphere can be calculated by assuming that it is a full sphere, for the vertices which lie on the fat area. This normal $\mathbf{n}_i$ can simply be represented as the direction joining the sphere center and the vertex $\mathbf{v}_i$. Figure A.2 illustrates the same.

Bibliography

Agarwal, S. & Mierle, K. (2012). Ceres solver. `http://ceres-solver.org`, accessed: 2018-09-10. Cited on 16, 72

Ahn, S.H. (2008). Opengl projection matrix. `http://www.songho.ca/opengl/gl_projectionmatrix.html`, accessed: 2018-09-10. Cited on 9

Alexander, O., Rogers, M., Lambeth, W., Chiang, M. & Debevec, P. (2009). The digital emily project. In *ACM SIGGRAPH 2009 Courses on - SIGGRAPH '09*, ACM Press. Cited on 37

Amberg, B., Romdhani, S. & Vetter, T. (2007). Optimal step nonrigid icp algorithms for surface registration. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, 1–8. Cited on 25

Arar, N.M. & Thiran, J. (2017). Robust real-time multi-view eye tracking. *CoRR*, **abs/1711.05444**. Cited on 34

Banf, M. & Blanz, V. (2009). Example-based rendering of eye movements. *Computer Graphics Forum*, **28**, 659–666. Cited on 27

Basri, R. & Jacobs, D.W. (2003). Lambertian reflectance and linear subspaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **25**, 218–233. Cited on 11

Beeler, T., Bickel, B., Beardsley, P., Sumner, B. & Gross, M. (2010). High-quality single-shot capture of facial geometry. In *ACM SIGGRAPH 2010 papers on - SIGGRAPH '10*, ACM Press. Cited on 28

Bérard, P., Bradley, D., Nitti, M., Beeler, T. & Gross, M. (2014). High-quality capture of eyes. *ACM Trans. Graph.*, **33**, 223:1–223:12. Cited on 26, 31

Bérard, P., Bradley, D., Gross, M. & Beeler, T. (2016). Lightweight eye capture using a parametric model. *ACM Trans. Graph.*, **35**, 117:1–117:12. Cited on 17, 26, 31

Bjõrck, A. (1996). *Numerical Methods for Least Squares Problems*. SIAM: Society for Industrial and Applied Mathematics. Cited on 12

Blanz, V. & Vetter, T. (1999). A morphable model for the synthesis of 3d faces. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '99, 187–194, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA. Cited on 1, 24, 25, 26, 29, 35, 37

Blignaut, P. (2013). Mapping the pupil-glint vector to gaze coordinates in a simple video-based eye tracker. Cited on 34

Cao, C., Weng, Y., Zhou, S., Tong, Y. & Zhou, K. (2014). FaceWarehouse: A 3d facial expression database for visual computing. *IEEE Transactions on Visualization and Computer Graphics*, **20**, 413–425. Cited on 2, 37, 59

Chen, J. & Ji, Q. (2008). 3d gaze estimation with a single camera without IR illumination. In *2008 19th International Conference on Pattern Recognition*, IEEE. Cited on 26, 31

Chen, J. & Li, W. (2005). Convergence of gauss–newton's method and uniqueness of the solution. *Applied Mathematics and Computation*, **170**, 686–705. Cited on 14, 15

Chowdhury, A., Chellappa, R., Krishnamurthy, S. & Vo, T. (2002). 3d face reconstruction from video using a generic model. In *Proceedings. IEEE International Conference on Multimedia and Expo*, IEEE. Cited on 28

Cook, R.L. & Torrance, K.E. (1982). A reflectance model for computer graphics. *ACM Trans. Graph.*, **1**, 7–24. Cited on 10

Cootes, T., Taylor, C., Cooper, D. & Graham, J. (1995). Active shape models-their training and application. *Computer Vision and Image Understanding*, **61**, 38–59. Cited on 23

Cristinacce, D. & Cootes, T.F. (2006). Feature detection and tracking with constrained local models. In *Procedings of the British Machine Vision Conference 2006*, British Machine Vision Association. Cited on 29, 45

Daugman, J. (1993). High confidence visual recognition of persons by a test of statistical independence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **15**, 1148–1161. Cited on 31, 72

Daugman, J. (2009). How iris recognition works. In *The Essential Guide to Image Processing*, 715–739, Elsevier. Cited on 26

DeCarlo, D., DeCarlo, D., Metaxas, D. & Stone, M. (1998). An anthropometric face model using variational techniques. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '98, 67–74, ACM, New York, NY, USA. Cited on 23

Dierkes, K., Kassner, M. & Bulling, A. (2018). A novel approach to single camera, glint-free 3d eye model fitting including corneal refraction. In *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications - ETRA '18*, ACM Press. Cited on 30, 31, 33

Edwards, G.J., Taylor, C.J. & Cootes, T.F. (1998). Interpreting face images using active appearance models. In *Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition*, 300–305. Cited on 23, 24

EL Haddioui, I. & Khaldi, M. (2012). Learner behaviour analysis through eye tracking. *International Journal of Computer Science Research and Application*, **02**, 11–18. Cited on 34

Fidaleo, D. & Medioni, G. (2007). Model-assisted 3d face reconstruction from video. In *Proceedings of the 3rd International Conference on Analysis and Modeling of Faces and Gestures*, AMFG'07, 124–138, Springer-Verlag, Berlin, Heidelberg. Cited on 28

Francois, G., Gautron, P., Breton, G. & Bouatouch, K. (2009). Image-based modeling of the human eye. *IEEE Transactions on Visualization and Computer Graphics*, **15**, 815–827. Cited on 30

Funes Mora, K.A., Monay, F. & Odobez, J.M. (2014). Eyediap database: Data description and gaze tracking evaluation benchmarks. Idiap-RR Idiap-RR-08-2014, Idiap. Cited on 67

Garrido, P., Valgaerts, L., Wu, C. & Theobalt, C. (2013). Reconstructing detailed dynamic face geometry from monocular video. In *ACM Trans. Graph. (Proceedings of SIGGRAPH Asia 2013)*, vol. 32, 158:1–158:10. Cited on 1, 24, 25, 29, 30, 35

Garrido, P., Zollhöfer, M., Casas, D., Valgaerts, L., Varanasi, K., Pérez, P. & Theobalt, C. (2016). Reconstruction of personalized 3d face rigs from monocular video. *ACM Transactions on Graphics*, **35**, 1–15. Cited on 29, 37, 43, 60

Ghosh, A., Fyffe, G., Tunwattanapong, B., Busch, J., Yu, X. & Debevec, P. (2011). Multiview face capture using polarized spherical gradient illumination. *ACM Transactions on Graphics*, **30**, 1. Cited on 28, 29

Hartley, R. & Zisserman, A. (2000). Camera models. In *Multiple View Geometry in Computer Vision*, 153–177, Cambridge University Press. Cited on 8, 28

Helmholtz, H.L.F.V. (1867). Handbuch der physiologischen optik. Cited on 3, 19, 33

Horner, D.G. (1999). *Oculomotor Functions and Neurology*. Cited on 19

Hurley, J.R. & Cattell, R.B. (2007). The procrustes program: Producing direct rotation to test a hypothesized factor structure. *Behavioral Science*, **7**, 258–262. Cited on 60

Ichim, A.E., Kadleček, P., Kavan, L. & Pauly, M. (2017). Phace: Physics-based face modeling and animation. *ACM Trans. Graph.*, **36**, 153:1–153:14. Cited on 23

Ishikawa, T., Baker, S., Matthews, I. & Kanade, T. (2004). Passive Driver Gaze Tracking with Active Appearance Models. Tech. Rep. CMU-RI-TR-04-08, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA. Cited on 31

Itti, L., Dhavale, N. & Pighin, F. (2004). Realistic avatar eye and head animation using a neurobiological model of visual attention. In B. Bosacchi, D.B. Fogel & J.C. Bezdek, eds., *Applications and Science of Neural Networks, Fuzzy Systems, and Evolutionary Computation VI*, SPIE. Cited on 26

Jeni, L.A. & Cohn, J.F. (2016). Person-independent 3d gaze estimation using face frontalization. In *2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, IEEE. Cited on 30

Jimenez, J., Danvoye, E. & von der Pahlen, J. (2012). Photorealistic eyes rendering,. *SIGGRAPH Talks, Advances in Real-Time Rendering.*. Cited on 26

Kar, A. & Corcoran, P. (2017). A review and analysis of eye-gaze estimation systems, algorithms and performance evaluation methods in consumer platforms. *IEEE Access*, **5**, 16495–16519. Cited on 1, 33

Kim, H., Garrido, P., Tewari, A., Xu, W., Thies, J., Niessner, N., Pérez, P., Richardt, C., Zollhöfer, M. & Theobalt, C. (2018a). Deep Video Portraits. *ACM Transactions on Graphics 2018 (TOG)*. Cited on 2

Kim, H., Zollöfer, M., Tewari, A., Thies, J., Richardt, C. & Theobalt, C. (2018b). Inversefacenet: Deep single-shot inverse face rendering from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Cited on 60

Lanitis, A., Taylor, C. & Cootes, T.F. (1995). An automatic face identification system using flexible appearance models. *Image and Vision Computing*, **13**, 393–401. Cited on 24

Lee, Y., Terzopoulos, D. & Waters, K. (1995). Realistic modeling for facial animation. In *Proceedings of the 22Nd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '95, 55–62, ACM, New York, NY, USA. Cited on 23, 30

Lewis, J.P., Anjyo, K., Rhee, T., Zhang, M., Pighin, F. & Deng, Z. (2014). Practice and theory of blendshape facial models. In S. Lefebvre & M. Spagnuolo, eds., *Eurographics 2014 - State of the Art Reports*, The Eurographics Association. Cited on 24

Majid, M., N Hoggan, R. & Muthappan, V. (2013). Angle kappa and its importance in refractive surgery. **6**, 151–158. Cited on 26, 71

Oyster., C. (1999). *The Human Eye: Structure and Function.*. Sinauer Associate, Inc. Cited on 25

Paysan, P., Knothe, R., Amberg, B., Romdhani, S. & Vetter, T. (2009). A 3d face model for pose and illumination invariant face recognition. In *2009 Sixth IEEE International Conference on Advanced Video and Signal Based Surveillance*, IEEE. Cited on 25

Pinscreen (2018). Pinscreen. inc. `https://www.pinscreen.com/`, accessed: 2018-10-10. Cited on 1

Ramamoorthi, R. (2006). Modeling illumination variation with spherical harmonics. In *Face Processing: Advanced Modeling Methods*, 385–424. Cited on 11, 29

Richardson, E., Sela, M. & Kimmel, R. (2016). 3d face reconstruction by learning from synthetic data. *CoRR*, **abs/1609.04387**. Cited on 30

Robinson, D.A. (1963). A method of measuring eye movement using a scieral search coil in a magnetic field. *IEEE Transactions on Bio-medical Electronics*, **10**, 137–145. Cited on 33

Run, L.V. & Berg, A.V.D. (1993). Binocular eye orientation during fixations: Listing's law extended to include eye vergence. *Vision Research*, **33**, 691–708. Cited on 20, 38

Sagar, M.A., Bullivant, D., Mallinson, G.D. & Hunter, P.J. (1994). A virtual environment and model of the eye for surgical simulation. In *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '94, 205–212, ACM, New York, NY, USA. Cited on 26

Saragih, J.M., Lucey, S. & Cohn, J.F. (2010). Deformable model fitting by regularized landmark mean-shift. *International Journal of Computer Vision*, **91**, 200–215. Cited on 29, 45

Schneider, A., Egger, B. & Vetter, T. (2018). A parametric freckle model for faces. In *2018 13th IEEE International Conference on Automatic Face Gesture Recognition (FG 2018)*, 431–435. Cited on 25

Shih, S.W. & Liu, J. (2004). A novel approach to 3-d gaze tracking using stereo cameras. *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics)*, **34**, 234–245. Cited on 33

Smilek, D., Birmingham, E., Cameron, D., Bischof, W. & Kingstone, A. (2006). Cognitive ethology and exploring attention in real-world scenes. *Brain Research*, **1080**, 101–119. Cited on 1

Smith, B.A., Yin, Q., Feiner, S.K. & Nayar, S.K. (2013). Gaze locking. In *Proceedings of the 26th annual ACM symposium on User interface software and technology - UIST '13*, ACM Press. Cited on 67

Sugano, Y., Matsushita, Y. & Sato, Y. (2014). Learning-by-synthesis for appearance-based 3d gaze estimation. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 1821–1828. Cited on ix, 34, 35, 63, 66, 68

Sumner, R.W. & Popović, J. (2004). Deformation transfer for triangle meshes. In *ACM SIGGRAPH 2004 Papers*, SIGGRAPH '04, 399–405, ACM, New York, NY, USA. Cited on 25

Świrski, L. & Dodgson, N.A. (2013). A fully-automatic, temporal approach to single camera, glint-free 3d eye model fitting [abstract]. In *Proceedings of ECEM 2013*. Cited on 31, 33

Tewari, A., Zollöfer, M., Kim, H., Garrido, P., Bernard, F., Perez, P. & Christian, T. (2017). MoFA: Model-based Deep Convolutional Face Autoencoder for Unsupervised Monocular Reconstruction. In *The IEEE International Conference on Computer Vision (ICCV)*. Cited on 1, 30, 45, 60

Tewari, A., Zollhöfer, M., Garrido, P., Bernard, F., Kim, H., Pérez, P. & Theobalt, C. (2018). Self-supervised multi-level face model learning for monocular reconstruction at over 250 hz. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Cited on 60, 71

Thies, J., Zollöfer, M., Stamminger, M., Theobalt, C. & Niessner, M. (2016). FaceVR: Real-Time Facial Reenactment and Eye Gaze Control in Virtual Reality. *arXiv preprint arXiv:1610.03151*. Cited on 72

Tran, L. & Liu, X. (2018). Nonlinear 3d face morphable model. In *In Proceeding of IEEE Computer Vision and Pattern Recognition*, Salt Lake City, UT. Cited on 71

Trucco, E. & Verri, A. (1998). *Introductory Techniques for 3-D Computer Vision*. Prentice Hall. Cited on 8

Wang, Sung & Venkateswarlu, R. (2003). Eye gaze estimation from a single image of one eye. In *Proceedings Ninth IEEE International Conference on Computer Vision*, IEEE. Cited on 30

Wang, C., Shi, F., Xia, S. & Chai, J. (2016). Realtime 3d eye gaze animation using a single RGB camera. *ACM Transactions on Graphics*, **35**, 1–14. Cited on 2, 32, 33

Wang, K. & Ji, Q. (2017). Real time eye gaze tracking with 3d deformable eye-face model. In *2017 IEEE International Conference on Computer Vision (ICCV)*, 1003–1011. Cited on 31, 32

Weise, T., Leibe, B. & Gool, L.V. (2007). Fast 3d scanning with automatic motion compensation. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE. Cited on 27

Weissenfeld, A., Liu, K. & Ostermann, J. (2009). Video-realistic image-based eye animation via statistically driven state machines. *The Visual Computer*, **26**, 1201–1216. Cited on 26

Wen, Q., Xu, F. & Yong, J.H. (2017). Real-time 3d eye performance reconstruction for RGBD cameras. *IEEE Transactions on Visualization and Computer Graphics*, **23**, 2586–2598. Cited on 26, 31

Whitehead, A. & Roth, G. (2004). Estimating intrinsic camera parameters from the fundamental matrix using an evolutionary approach. *EURASIP Journal on Advances in Signal Processing*, **2004**. Cited on 9

Wikipedia (2014). Spherical harmonics — Wikipedia, the free encyclopedia, wikimedia commons. `https://en.wikipedia.org/wiki/Spherical_harmonics`, accessed: 2018-09-14. Cited on 11

Williams, L. (1990). Performance-driven facial animation. In *Proceedings of the 17th annual conference on Computer graphics and interactive techniques - SIGGRAPH '90*, ACM Press. Cited on 28

Wilson-Pauwels, L., Stewart, P., Akesson, E. & Spacey, S. (2013). *Cranial Nerves, Function and Dysfunction*. 3rd edn. Cited on 18

Wong, A. (2004). Listing's law: clinical significance and implications for neural control. *Survey of Ophthalmology*, **49**, 563–575. Cited on 19, 20, 21

Wood, E. & Bulling, A. (2014). EyeTab. In *Proceedings of the Symposium on Eye Tracking Research and Applications - ETRA '14*, ACM Press. Cited on 30, 31, 72

Wood, E., Baltruaitis, T., Zhang, X., Sugano, Y., Robinson, P. & Bulling, A. (2015). Rendering of eyes for eye-shape registration and gaze estimation. In *2015 IEEE International Conference on Computer Vision (ICCV)*, IEEE. Cited on 67, 68

Wood, E., Baltrušaitis, T., Morency, L.P., Robinson, P. & Bulling, A. (2016a). A 3d morphable eye region model for gaze estimation. In *Computer Vision – ECCV 2016*, 297–313, Springer International Publishing. Cited on 2, 3, 26, 27, 32, 35, 38, 68, 69, 72

Wood, E., Baltrušaitis, T., Morency, L.P., Robinson, P. & Bulling, A. (2016b). Learning an appearance-based gaze estimator from one million synthesised images. In *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications*, ETRA '16, 131–138, ACM, New York, NY, USA. Cited on 26, 34, 68

Wood, E., Baltrušaitis, T., Morency, L.P., Robinson, P. & Bulling, A. (2018). GazeDirector: Fully articulated eye gaze redirection in video. *Computer Graphics Forum*, **37**, 217–225. Cited on 26, 32

Wood, E.W. (2017). Gaze estimation with graphics. Cited on 40

Wu, C., Bradley, D., Gross, M. & Beeler, T. (2016). An anatomically-constrained local deformation model for monocular face capture. *ACM Trans. Graph.*, **35**, 115:1–115:12. Cited on 23

Wu, H., Chen, Q. & Wada, T. (2004). Conic-based algorithm for visual line estimation from one image. In *Sixth IEEE International Conference on Automatic Face and Gesture Recognition, 2004. Proceedings.*, IEEE. Cited on 30

Xie, Y. & Ji, Q. (2002). A new efficient ellipse detection method. In *Object recognition supported by user interaction for service robots*, IEEE Comput. Soc. Cited on 31, 72

Zhang, L., Snavely, N., Curless, B. & Seitz, S.M. (2004). Spacetime faces. In *ACM SIGGRAPH 2004 Papers on - SIGGRAPH '04*, ACM Press. Cited on 27

Zhang, W., Zhang, T.N. & Chang, S.J. (2010). Gazing estimation and correction from elliptical features of one iris. In *2010 3rd International Congress on Image and Signal Processing*, IEEE. Cited on 31, 72

Zhang, X., Sugano, Y., Fritz, M. & Bulling, A. (2015). Appearance-based gaze estimation in the wild. *CoRR*, **abs/1504.02863**. Cited on 34, 68, 69

Zhang, X., Sugano, Y., Fritz, M. & Bulling, A. (2018). MPIIGaze: Real-world dataset and deep appearance-based gaze estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1–1. Cited on 67, 68

Zhao, X., Zou, X. & Chi, Z. (2012). A 3d gaze estimation method based on facial feature tracking. In *2012 International Conference on Computerized Healthcare (ICCH)*, IEEE. Cited on 34