Real-Time Relighting of Human Faces with a Low-Cost Setup

Nejc Maček

Technische Universiteit Delft

TU Delft

Real-Time Relighting of Human Faces with a Low-Cost Setup

by

Nejc Maček

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Friday June 25, 2021 at 10:00 AM.

Student number:	4929446	
Programme:	MSc Computer Science	
Affiliation:	Computer Graphics and Visualization,	
	Intelligent Systems,	
	Faculty of Electrical Engineering, Mathematics and Computer Science	
Thesis committee:	Dr. R. Marroquim,	TU Delft, Thesis advisor
	Prof. Dr. E. Eisemann,	TU Delft, Chair
	Prof. Dr. J. van Gemert,	TU Delft
Daily supervisor:	B. Usta,	TU Delft

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft

Preface

This work is a master thesis written in affiliation with Computer Graphics and Visualization group at Delft University of Technology. The work aims to facilitate the use of relighting in common applications by simplifying the acquisition process, required to perform it. We introduce a novel, low-cost acquisition technique that can be performed at home. Furthermore, We describe a novel interpolation method to mitigate acquisition imperfections. Finally, we provide a proof-of-concept for dynamic relighting by applying facial movement to static relit images.

First and foremost, I would like to thank my thesis advisor, Dr. Ricardo Marroquim, and my daily supervisor, Baran Usta, for their continuous guidance, encouragement, and an endless amount of support throughout the duration of the project. Furthermore, I would like to thank Prof. Dr. Elmar Eisemann for his assistance and useful ideas and hints that helped shape the final result. I would also like to thank Prof. Dr. Jan van Gemert for expressing his interest in the work and joining the thesis committee. A special thanks goes to everyone who volunteered in the face-capturing experiment and help me to obtain results that made this thesis possible.

Nejc Maček
Delft, June 2021

Contents

1 Introduction

With the world becoming a global village, long distances are cut short using various online communication and streaming platforms and solutions. Especially video conferencing has recently gain in popularity and, therefore, developers have been busy trying to improve their software and deliver more features to outrun their competitors. One such feature that is often used is virtual backgrounds. It allows a user to change the background of their camera feed and replace it with a new, virtual one, be it a picture or a video. However, this is achieved using masking, by detecting the contour around the user's head and body, cropping them out and compositing them on top of the new background. This undermines user experience as the physical lighting on the user remains unchanged and causes a mismatch compared to the new, virtual background, as seen in Figure 1.1. While this is just one of many use-cases, our work aims at narrowing this gap.

Figure 1.1: Left: Original image. Right: Lighting mismatch between the background and the subject from the left image that has been composited onto the new background.

Modifying the illumination of an image or video after its acquisition is known as *relighting*. In this work, we specifically focus on relighting human faces. In order to infer the appearance of a face under given arbitrary lighting conditions, we need to predict the light transport function, or how environment light reflects off of the face. There are two main trends for performing relighting: image-based and learning-based. The former relies on sampling the light transport function by capturing the appearance of the face under various view and/or light directions [1, 10, 18, 27, 55]. With enough samples it is possible to reconstruct a 3D face model and interpolate the sampled reflectance information to simulate novel lighting conditions. High-end devices, such as Light Stages, use many cameras and hundreds of light sources in a dome-like structure. These custom-made devices serve as benchmarks but target professional users, such as the movie industry, and not the general public. Learning-based approaches [31, 34, 38, 43, 49, 54, 58] are a recent alternative for tackling portrait relighting and have gained popularity. Nevertheless, many methods require training data from Light Stage-like devices and do not easily generalize. The other methods rely on pre-processing large datasets of facial images for training but produce results that lack the visual richness of natural portraits.

We propose a method that considerably lowers the requirements of image-based approaches. We

target bringing such acquisition methods to the masses by relying only on a smartphone flash, a camera (possible another smartphone) and a reflecting sphere. The acquisition process is manually performed by moving the flashing smartphone around the subject, and lasts no longer than one minute. The rest of the process requires minimal user intervention. Our method produces light reflectance information per pixel that can be used for relighting. Furthermore, we demonstrate how our method can be applied for real-time relighting applications, such as video streaming.

Outline The remainder of this work is organised as follows. In Chapter 2 we explain core concepts related to light transport, reflectance, and perception, as well as some technicalities for concepts referenced throughout the work. Then, existing research in the field as well as state-of-the art is detailed in Chapter 3.

Chapter 4 details the acquisition technique used for relighting, starting with the initial inspiration for relighting itself in Section 4.1. While we eventually deviate from the described relighting method, we inherit some concepts of its image-capture acquisition technique, along with image alignment problems. Mitigation attempts thereof are detailed in Section 4.2. While some are more and some less successful, all findings contribute to introducing a novel acquisition method in Section 4.3.

Next, Chapter 5 provides an overview of the creation and refinements of reflectance maps, which are constructed from the captured data and used for relighting. First, their structure and construction are outlined in Section 5.1. Then, Section 5.2 describes how samples that are missing are estimated. As this process yields many artefacts, refinements that account for shadows and varying light intensity are accounted for in Sections 5.3 and 5.4, respectively. Additional adjustments are introduced in Section 5.5 and, finally, an extension of reflectance maps beyond the existing samples is given in Section 5.6.

The constructed reflectance maps are used for relighting and this process is described in Chapter 6. Section 6.1 describes how this is done for static images. Next, optimisations that make the process run in real-time are described in Section 6.2. To extend relighting to a dynamic setting and perform it on videos, Section 6.3 describes a proof-of-concept approach for achieving this.

We present and analyse relighting results in Chapter 7 and list limitation in Chapter 8. Finally, we conclude with a brief summary and list potential future directions in Chapter 9.

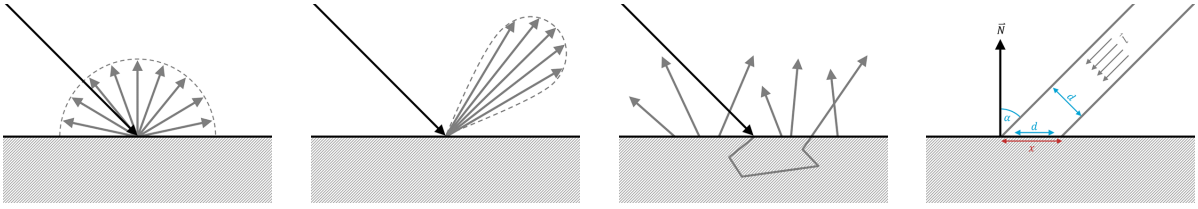
2

Background

In this chapter, we cover core concepts that will be the basis of research in the remained of the thesis. We first summarise concepts related to light transport and human perception that lay foundation for understanding relighting in Section 2.1. Then, in Section 2.2 we distinguish linear and gamma space that affect our relighting computations. We proceed to define coordinate spaces that we use as well as conversions between them in Section 2.3. Finally, Section 2.4 details how environment illumination is described using environment maps, which are used to define the lighting of relit images.

2.1. Light Transport and Reflectance Functions

To understand relighting, we first have a look at human sight, perception, and light traversal. People perceive the world around us as receptors in our eyes react to light (photons), emitted from a light source, that bounced from surrounding objects and into our eyes. In computer graphics, such a traversal of light in a scene is described using a light transport function. We are specifically interested in how different materials reflect light, which is described with a reflectance function, as this is what affects their appearance. In simple case for opaque surfaces, this can be captured with a bidirectional reflectance distribution function (BRDF). In traditional implementations, the function consists of two isotropic components, namely diffuse and specular, depicted in Figures 2.1a and 2.1b, respectively. The former, diffuse, measures how incoming light scatters homogeneously in all directions, whereas the latter, specular, measures intensity of light as it reflects from the surface in the direction of incoming light reflected over the normal of the surface. A reflectance function that is solely based on the diffuse component is called Lambertian.



(a) Diffuse reflection. (b) Specular reflection. (c) Subsurface scattering. (d) Foreshortening.

Figure 2.1: Figures 2.1a, 2.1b, and 2.1c represent different reflection and scattering effects. Figure 2.1d represents foreshortening where surface patch x appears smaller (as large as d) from the perspective of light direction $\mathbf{l}$. Here, $d = x \cos \alpha$.

The BRDF model is an approximation and is incomplete for complex real-world materials, especially for human skin. While human skin does reflect some light directly off of itself, which can be considered a specular reflection, a lot of light is refracted inside the skin and then re-emitted in different directions. If this happens close to the surface of the skin, we can approximate it as a diffuse reflectance, sometimes also referred to shallow scattering in the case of skin. However, human skin consists of several layers of biological tissue, which allow the light to traverse further into the skin and travel for a certain distance before being re-emitted [2, 19, 39]. This effect is known as subsurface scattering and is depicted in Figure 2.1c. As an extension to BRDF, it forms bidirectional scattering-surface reflectance

distribution function or BSSRDF [28]. This phenomenon drastically affects the appearance of skin. It mostly manifests in red tones that can be most easily be observed where a light shines through a thin layer of body tissue such as ears or fingers. Another case can be observed where a shadow is cast over a patch of skin. Light from the lit area can traverse through the skin and be emitted in the shadow area, causing the part of the shadow region close to the shadow boundary to appear somewhat red. While these effects may sound marginal, people are very perceptive of even the smallest inaccuracies in appearance of faces, making this effect an important consideration especially applications such as the movie or game industry.

Regardless of the reflectance function used, appearance of a surface is still subject to the amount of incoming light (i.e. the number of photons that hit the surface). This is primarily dependent on the orientation of the surface in relation to the light source, measured by the angle between the surface's normal and a vector from the surface toward the light source. A greater angle causes a patch of the surface to appear smaller as seen from the perspective of the light source. This effect is known as foreshortening and affects light intensity on the surface. In other words, for greater angles, the same amount of light is spread over a larger area of a surface, which means that an individual patch of the surface receives a smaller amount of light, making it appear darker. This effect is depicted in Figure 2.1d. The factor that describes the loss of light intensity is the cosine of the measured angle.

2.2. Gamma Correction

Relighting requires many light computations to be performed, however, these cannot always be directly performed on images that we capture with a camera. Light is additive, thus, combining light emitted from multiple source is as simple as summing the individual contributions. However, human perception of light is logarithmic; we are better at distinguishing different shades of darker colours than brighter. Thus, there is a discrepancy between a linear scale on a physical and perceptual level. Most modern images use 8-bit colours and therefore only have 256 distinct levels for each of the RGB channels. If these values were encoded in (physically) linear space, the perceived difference between two darkest levels would be great, yet the brightest levels would barely be distinguishable. To overcome this issue, colours are encoded using a perceptually linear scale. We say that images are encoded in gamma space. When displaying these images, a digital monitor would then convert the colours using a logarithmic mapping from gamma space back to linear space, such that they would be perceived correctly by the end user. The conversion between gamma and linear space is called gamma correction. In its simplest form, the conversion is computed as $c_L = c_G^{\gamma}$ or, conversely, $c_G = c_L^{\gamma^{-1}}$, where c_L and c_G are the same colour in linear and gamma space, respectively, and $\gamma = 2.2$ is the gamma correction factor. As this is a non-linear operation, we cannot simply add values in gamma space to combine light source as we would in linear space. We first need to convert images to linear space, then perform any computations, and then convert them back to gamma space. We perform this operation any time we deal with light computations. However, some image formats, such as EXR and HDR, already store images in linear space. When using these formats, no conversion to or from linear space is needed.

2.3. Coordinate Spaces

Typical applications in computer graphics operate in Euclidean 3D space, however, the orientation of the space often varies in different use-cases and applications. To avoid ambiguity, we give clear definitions of coordinate spaces that we use and detail conversions between them. Our Euclidean 3D space is defined such that the X axis increases rightward, Y upward, and Z toward the camera. This orientation is used consistently throughout this work.

The direction of any vector in this 3D space, anchored at the origin, can be expressed using spherical coordinates (θ, ϕ), also referred to as longitude and latitude, respectively. Here, $\theta \in [-\pi, \pi]$ represents rotation round the Y axis, according to the right-hand rule, from $-X$ toward $+X$, with the extremes aligned toward the negative Z axis. ϕ represents the polar angle or inclination from the XZ plane, where negative values face in the direction of negative Y axis, and positive toward positive Y. A graphical visualisation of the coordinate system can be seen in Figure 2.2 The coordinates only represent direction, no radius or distance is given. If distance is assume to be one, however, then the vector can be thought of as pointing from the origin to a unique point on the unit sphere with the direction defined

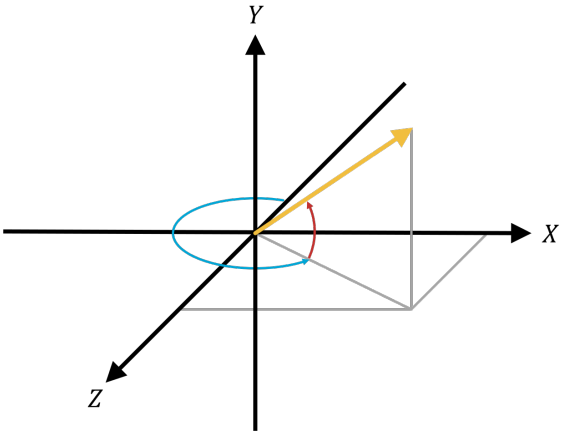


Figure 2.2: 3D coordinate space with X, Y, and Z axes. A direction vector (yellow) can be represented using its latitudinal (red) and longitudinal (blue) coordinates.

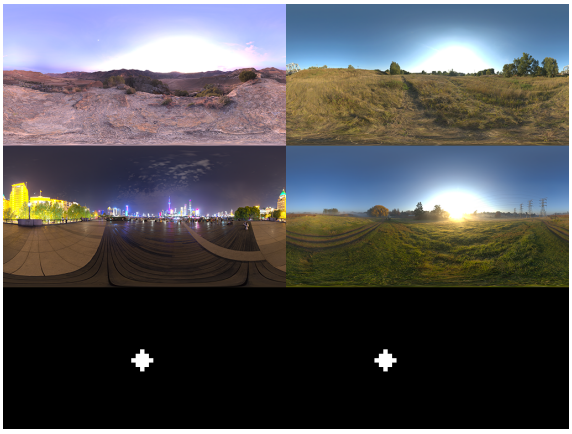


Figure 2.3: Example environment maps. Top two rows represent examples of high-resolution real-world environment maps. The bottom row represents two low-resolution environment maps representing a single small light source that illuminate a scene from in front (bottom left image) and from the viewer's left (bottom right image).

by (θ, ϕ). Conversion from a unit vector $\mathbf{v} = (x, y, z)$ to spherical coordinates is computed as

$$\phi = \arcsin y$$
$$\theta = \arctan2(x, z),$$

where $\arctan2(y, x)$ is a 2D function that returns the angle $\delta \in [-\pi, \pi]$ of the line through point (x, y). Conversely, spherical coordinates can be converted to a 3D unit vector using

$$X = \sin\theta \cos\phi$$
$$Y = \sin\phi$$
$$Z = \cos\theta \cos\phi.$$

2.4. Environment Maps

Our relighting process involves evaluating a set of reflectance function to compute the appearance of relit face. However, reflectance functions do not describe the incoming light, which is given in form of environment maps.

Environment maps are high-dynamic-range images of size $w \times h$ (in our case 64×32). Each pixel of an environment map represents light that illuminates a scene from a certain direction or small area. The environment map can be thought of as a texture for a 3D sphere at an infinite distance from the origin. The colours mapped from the environment map to the sphere then shine light onto the scene, illuminating it. Each point on the map therefore also represents a point on the sphere. Thus, environment maps can be represented in two coordinate systems. The first is using its image-space coordinate system, represented as (x, y) where $x, y \in [0, 1]$ and x increases horizontally and y vertically, and the origin in the top-left corner. These image-space coordinates can be mapped to spherical coordinates (θ, ϕ), which we have already defined. The conversion to spherical coordinates is computed as

$$\theta = 2\pi(x - 0.5)$$
$$\phi = \pi(0.5 - y).$$

Similarly, polar coordinates can be converted back to image-space coordinate system as

$$x = \frac{\theta}{2\pi} + 0.5$$
$$y = 0.5 - \frac{\phi}{\pi}.$$

Example environment maps can be seen in Figure 2.3.

3

Related Work

Skin's appearance and reflection properties are a vital consideration for any artificial representation of skin and have, therefore, been studied extensively. Skin reflectance was found to vary significantly from individual to individual and is subject not only to exceptional skin conditions like albinism, but also to regular variations in biologically conditioned factors. Examples include levels of melanin and hemoglobin [2, 29], age, moisture (water content), environment humidity, time of day, skin treatment prior to observation [25], sweat, blood flow [19, 55], emotion [29], sun exposure, genetics, health, substance use [22], and even external factors such as skin stretch [25, 37] and make-up. In general, however, skin was found to exhibit highly anisotropic specular and scattering components that change based on the incidence angle of incoming light, with angles perpendicular to the surface exhibiting more scattering and angles parallel to the surface exhibiting more specularity [25]. Subsurface scattering also received a lot of attention as it greatly changes the perceived colour of the skin [19, 51, 55]. Researches managed to capture and reproduce the effects listed above using custom rendering methods. However, they relied on measurements conducted with lasers [25], polarised light filters [10, 20, 42], or other custom in vivo techniques with specialised equipment [29, 55], all of which are not suitable for a simple setup that we aim for. These measurements also revealed that skin appearance varies for different parts of the face [55], which makes generalising skin's reflection function even more difficult and would require complex models to accurately estimate.

Some relighting methods have adapted to these problems by employing so-called one-light-at-a-time (OLAT) approaches [10, 17, 26, 27, 55]. These image-based techniques sample the light transport function by recording a subject's face in a carefully controlled lighting environment. The function can then be used to perform relighting.

Debevec et al. [10] proposed the first Light Stage to extract reflectance fields for face in an OLAT manner. Two fixed cameras and a light source attached to a gantry that rotates around the subject's face are used for capture. Moreover, polarising filters separate diffuse and specular components. Figure 3.1 (left and center) shows this setup. A reflectance function is inferred per pixel to perform relighting. Newer Light Stage versions involve more cameras and a larger number of fixed light sources in a dome-like structure, allowing for increasingly denser and precise capturing of human faces [9]. Such dome-based setups are commonly used in the entertainment industry and can be extended to performance capture [40] and the full human body [11].

Lighter versions of the Light Stage lower the setup requirements by focusing on diffuse and specular maps instead of reflectance fields. Single-shot, or quasi-instantaneous, methods are particularly attractive since they avoid the issues of head movement [15, 16, 20, 21, 32, 41]. Nevertheless, such systems still require several cameras and special lighting, such as polarised illumination patterns. However, we aim at a wide-spread use of face appearance capture, mostly relying only on common devices, such as a pair of smartphones.

Reflectance fields, even when acquired with such domes, are considered sparse for many applications and require interpolation [14]. Recently, multi-view OLAT data has been used as training sets for many machine learning-based approaches to increase the lighting resolution [50, 57]. Xu et al.'s deep neural network for relighting requires only five images with different light directions as input [56]. Yet,

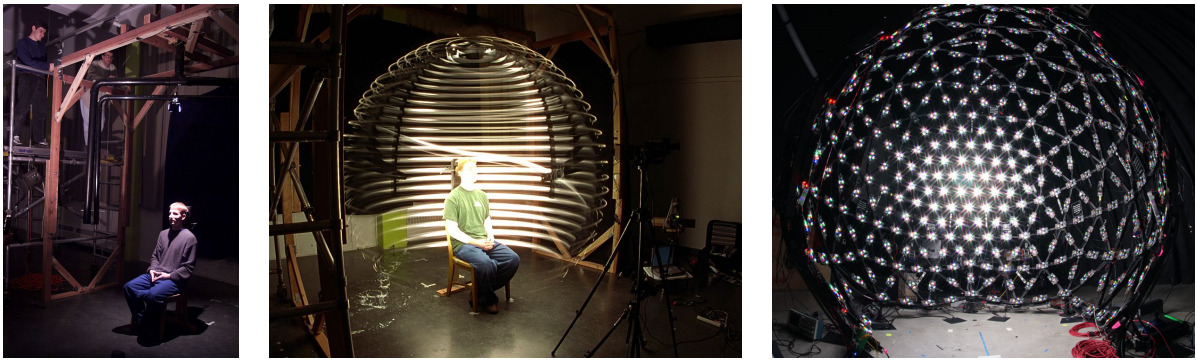


Figure 3.1: Left and center: Light stage gantry and acquisition timelapse of the Light Stage used by Debevec et al. [10]. Right: Light Stage used by Meka et al. [34].

their training is based on renderings with pre-defined BRDFs, which cannot capture the complexity and broad variation of human faces.

Using a neural-network trained with Light Stage data, Meka et al. [34] proposed a method that only requires two images per subject, but these two images need to be acquired under special gradient lights obtained via Light Stages as seein in Figure 3.1, right. Other methods use such data to train a network that accepts a single portrait as input [31, 38, 49, 54]. Nevertheless, apart from requiring challenging data for training, these networks have issues when extrapolating data.

To lower training-data requirements, one can fit a 3D mesh to single portraits to synthesise data [43, 58] or perform relighting between two portraits [46]. Nevertheless, such methods cannot generalise illumination conditions and mostly rely on a simple Lambertian model for the face, which cannot match the quality of directly sampling the light-transport function.

Light Stages have also been used for digitalisation of cultural artefacts [26], although alternative approaches with a simpler setup have gained much attention later. Namely, Reflectance Transformation Imaging (RTI) [35], which internally uses Polynomial Texture Maps (PTM) [33]. This is another OLAT method that uses a fixed camera to record several images of an object, each illuminated from a different direction. RTI provides a framework for viewing the scanned objects while changing illumination conditions, enhancing specular highlights, visualising normal maps, and more. While this method was designed for capturing static objects, it served as an initial inspiration for exploring options of using it to capture human faces instead.

4

Acquisition Method

Face relighting requires full representation of the reflectance function, which changes across the face [55] and differs for each individual. To capture this, we need an efficient acquisition method that allows us to record the this function. We take inspiration from Reflectance Transformation Imaging (RTI) [35] and investigate the capabilities of the approach in Section 4.1. While RTI-like acquisition of capturing faces is simple and efficient, it exhibits many artefacts. Most notably, misalignment between the captured images due to head movements during capture. While we eventually deviate from RTI and base our capture and relighting method around Light Stages [10], we still face the same issues that occurred during RTI. Thus, we first describe several tentative solution to misalignment in Section 4.2. Because not all of the listed methods are suitable for the problem at hand, we learn how to adjust the acquisition setup to overcome the issue. The final version of the acquisition setup is described in Section 4.3, along with methods to extract useful data from the captured footage.

4.1. Reflectance Transformation Imaging

RTI offers a simple relighting technique that operates on reflectance function of objected inferred from data, capturing during a simple acquisition technique using a single camera, a portable light source, and a reflective sphere. While normally used for static objects, we wish to use RTI's simple acquisition technique and adapt it for human faces. Thus, in this section, we explore the applicability of this method.

In its original form, RTI uses Polynomial Texture Maps (PTM) [33] to compute an approximate reflectance function based on multiple images of a selected artefact. Capturing is done using a professional camera. Each of the captured images is recorded under different lighting conditions generated by a flash or other similar light source with directional light oriented towards the object. The light source is moved at a constant distance around the object as pictures are being taken. A reflective sphere is added to the scene and exhibits a specular highlight of the light source. Highlight's location on the sphere in the captured images is used to compute the direction of incoming light. A detailed process, as well as the software for performing RTI is provided by Cultural Heritage Imaging (CHI)[1].

Contrary to the original approach, we use this method to capture human faces at home. We ask our subject to sit still in front of a plain-coloured background, preferably matte black. It was found that a headrest helped our subject keep their head still. We use a smartphone camera instead of a professional camera and a directional table lamp instead of a camera flash. The smartphone is fixed in a holder made of cardboard since a tripod was not available. For the reflective sphere, CHI recommends to use a black porcelain sphere, however, for simplicity, we use red Christmas tree ornaments which can be easily purchased. The process is done in a darkened room with window shades down, although the room was not completely dark. Apart from the subject, at least one other person needs to be present to move the light source and remotely trigger the camera shutter as the light source is being moved around. The acquisition process takes up to a few minutes.

Afterwards, we remove images that are not usable. This may happen if the light source is too far to the side and casts dark shadow over half of the face. The remaining images are fed to the RTI software to produce results. Our first observation is that auto-stabilisation, auto-focus, white-balancing,

[1] http://culturalheritageimaging.org/Technologies/RTI/

automatic exposure adjustments, and similar settings introduce undesirable shaking and unpredictable changes in colour; therefore, they should be disabled, if possible. This is especially pronounced for front-facing cameras, which may, on some smartphones, produce pictures of prohibitively low quality, especially in dark environments. Furthermore, taking pictures with the default camera app significantly increases the quality of images as opposed to grabbing an active frame from the live video feed. Second, we captured several datasets with different amount of ambient illumination. It was found that RTI reconstructs the same amount of detail regardless of whether there is a small amount of ambient illumination present. In such a case, however, the inclination of normals in the normal map was less pronounced (i.e. the normal maps were flatter).

Self-shadowing effects (e.g. when a nose casts a shadow over a cheek) affected the estimated normals and caused seams on smooth surfaces. Furthermore, normals in concave areas, most notably around ears, failed to be estimated correctly due to the lack of light illuminating the inside of ears when the light source was on the other side of the head. These artefacts happen because RTI expects a convex object where normals can be reliably estimated based on the captured colours and light directions. Faces, however, are not convex and contain many concavities. We tested the RTI software against the ground truth with renders of a ray-traced 3D head model. Indeed, the software failed to reconstruct concave areas as seen in Figure 4.1. It also reproduced great level of detail, down to the level of skin pores. Unfortunately, this was not the case for captured images, where normals appeared blurry, significantly decreasing their quality.

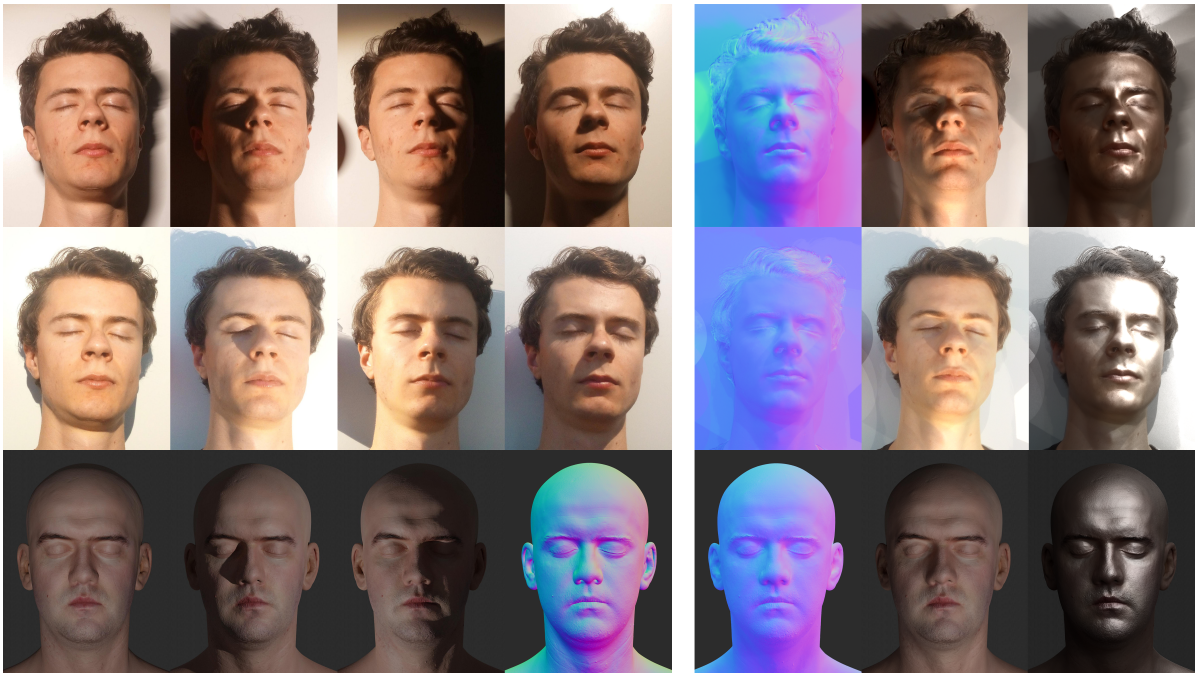


Figure 4.1: RTI input files and results. Each row represents a dataset with different features (top to bottom): little ambient illumination, ambient illumination, synthetic renders. The first four columns represent example input image with light source in front, to viewer's right, left, and top-right, respectively. The last three images represent constructed normal maps, example relighting with light source to the bottom-right, and specular enhancements for light source top-right. Notice how normal maps in the first row produce stronger normals as opposed to the second. The image in the third row fourth column represents ground truth normal maps for the synthetic case. Additionally, right-facing normals in the right ear (viewer's left) cannot be reproduced.

While RTI provided a good initial insight into relighting using image-based approaches, its primary drawback are twofold. First, it suffers from misalignment artefacts and, second, it cannot be used for relighting under arbitrary environmental lighting. Thus, we eventually decided to opt for a more elaborate approach similar to Debevec et al. [10], that uses a spatially-varying encoding of reflectance functions for each pixel of the face, which is further explained in Chapter 5. This approach allows us to perform relighting under arbitrary illumination. However, it is also an image-based technique and, therefore, requires images of faces under varying illumination, just like RTI. Thus, the problem of misaligned images still persists. In the following section, Section 4.3.3, we analyse various methods for stabilisation to overcome this. While not all the described methods perform stabilisation successfully,

we learn how to refine the acquisition method in a way, that allows us to stabilise images after all. The new acquisition method is our novel contribution and is described in Section 4.3. It is a hybrid approach that is imitates the aforementioned Light Stage, but with the simplicity of an RTI-like setup.

4.2. Alignment of Images

Subjects that we capture in our acquisition process cannot always stay completely still, causing misalignment and decreased quality. Even though we use a headrest, small movements cannot be avoided. Figure 4.2 visualises these misalignment issues and potential improvement. To help keep head movements minimal, research in face capture techniques clearly shows a trend in decreasing the acquisition time, going from few minutes in early face capture stages to a few seconds [15, 16, 20, 21, 32, 41] or even a single-shot capture for very detailed face models [3]. On the other hand, some researchers state that their subjects had no problems staying still for a few minutes [10]. However, the resolution of their captured images seems forgivingly small, making it difficult to notice small head movements at all. This also relates to our acquisition technique. A short capture with 9 images was able to produce much better results in terms of alignment than a dataset with 36 images that took longer to capture. Unfortunately, 9 images are not sufficient to accurately capture lighting details over the whole face, and minor head motion was present nonetheless. Thus, alignment of images remains an issue that cannot be completely solved by our acquisition method alone. Instead, we explore options for aligning source images as a post-processing step.

Figure 4.2: The image on the left represents the average of unaligned captured images. The right image represents the average of the same images after (manual) alignment. Notice how blur artifacts are much less pronounced in the right image.

4.2.1. Dense Optical Flow

The first attempt at stabilising was to use dense optical flow [12]. Dense optical flow tracks the movement of pixels across several frames of a video by comparing their local neighbourhoods. We deem the images that we capture to be close to a video as the face is mostly still, only the lighting changes. To minimise the error of tracking movement across several frames, we designate one frame, that is lit from in front, as an anchor frame. We then compute dense optical flow to every other frame, as if they were consecutive frames of a video. The result is a movement vector for each pixel of each frame (apart from the anchor frame, which we are trying to align other frames to), which can be seen in Figure 4.3a. Unfortunately, lighting differences appear to be too significant for dense optical flow to perform well enough to be used for stabilisation.

We identify several issues. By definition, optical flow assumes that pixel intensities of a moving part do no change. In our case, however, movement is detected because specular highlights, especially on the hair, move as a result of illumination changes and not head movements. Similarly, shadows caused by self-shadowing effects are also tracked as they move over the caused because of illumination changes. Furthermore, if a whole area of the face is in shadow and pixel intensities in that area change drastically, it makes tracking almost impossible. Even though we expected these problems to occur, it seems that they are more significant than expected. Thus, we conclude that dense optical flow is not an appropriate tool for stabilisation.

(a) Visualisation of movement in dense optical flow.

(b) Best tracking location for template matching.

(c) Good features to track with and without a mask.

(d) Feature tracking using sparse optical flow.

Figure 4.3: Different stabilisation techniques. Figure 4.3a shows pixels of a face moved according to dense optical flow estimations, leaving behind empty space coloured black. Figure 4.3b shows which areas of the face are easiest to track using template matching. Figure 4.3c displays how masking affects detection of good features to track. Figure 4.3d shows tracking using sparse optical flow. The red indicators failed to track, yellow indicators are outliers, and green indicators were used for rigid registration.

4.2.2. Rigid Registration

The next approach was to track a handful of points on the face and use their movement to compute stabilisation. To track these points, we could add fiducial makers onto the face during capture, but for simplicity and to keep the footage clean, we did not wish to do so. Instead, we tracked selected facial landmarks (i.e. distinguishable points on the face) in post-processing, that is, we detect those landmarks in each individual image separately. As a result, we obtain a set of facial landmarks, each with a slightly different location on each image.

The points that we track give enough information to align the images to one another. We choose to approximate head movement using image-space translation and rotation combination. In other words, we assume one image can be aligned to another by simply translating and rotating it by a certain amount. We always align two images at a time. Let $\mathcal{P} = \{\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_n\}$ and $\mathcal{Q} = \{\mathbf{q}_1, \mathbf{q}_2, ..., \mathbf{q}_n\}$, where $\mathbf{p}_i, \mathbf{q}_i \in \mathbb{R}^2$ be the two sets of points for two images we are trying to align. We wish to find a rotation matrix R and translation vector $\mathbf{t}$, such that the transformed points of $\mathcal{P}$, computed as $\mathbf{p}'_i = R\mathbf{p}_i + \mathbf{t}$, would be as close to their counterparts in $\mathcal{Q}$. We formalise this disparity as a least squares error whose minimisation yields our best solution:

$$(R, t) = \underset{R,\mathbf{t}}{\operatorname{argmin}} \sum_{i=1}^{n} ||(R\mathbf{p}_i + \mathbf{t}) - \mathbf{q}_i||^2.$$

We obtain the results if we equate its derivative to zero and solve for R and $\mathbf{t}$. This well-known problem is called rigid registration; the reader is referred to other sources for implementation details [48].

This approach can operate on any set of at least three points for the images. How we obtain the points is subject to another method. In the following section, we test the accuracy of the method by manually selecting and tracking points. We test an automated tracking method in Section 4.2.4 and an automated point detection method in Section 4.2.5, until finally finding a suitable solution for both problems in Section 4.2.6.

4.2.3. Manual Alignment

We can use rigid registration using manually identified and placed landmarks over every frame. We call this approach manual alignment. Eventually, we wish to automate this process, however, this method is sufficient to validate the accuracy of rigid registration.

To test the approach against ground truth, we take an image, and rotate and translate it by a set amount. Then, we perform manual alignment on the original image and the transformed image and see if the ground truth and estimated transformations match. Our results over several iterations show that, for a source image with resolution of 1591×1846 pixels, the average errors over 10 iterations did not exceed 2.5 pixels for translation and $0.5°$ rotation. The error is deemed very low, especially considering

that the resolution of the images was high and even small facial landmarks were spanning over multiple pixels, which made it difficult to locate their position precisely. During the experiment, regardless of the resolution of the image, the landmarks were always chosen with a per-pixel accuracy. We did not observe significant improvements or decline in accuracy for a higher number of facial landmarks used.

Furthermore, we tested the method on datasets that we captured for the initial RTI tests. All images were aligned to a selected anchor frame, with frontal illumination. Points that we were unable to locate due to shadows or other lighting effects were skipped in the process. We performed visual inspection by comparing an average of original images with what of aligned images. The latter was observed to be significantly less blurry, which implies that images were no longer misaligned. This validation method has proven to be very effective and is therefore used when no ground truth is available. Results of this method can be observed in Figure 4.2.

4.2.4. Template Matching

While using rigid transformation using manually located points performs well, we wish to use an automated system for performing the alignment. Thus, we aim to find a method that could track the same points selected on a so-called anchor frame across the remaining frames. We first resort template matching as implemented by OpenCV [5].

In general, template matching is used to find the location of an image part in the target image that it was taken from. The image part of size $s \times s$ is called template and is compared to patches of the target image until a best match is found. We estimate that this method is robust enough to allow searching for the template location in an image that is not the same, but similar to the target image. Thus, if our template is a patch from around a selected facial landmark in the anchor frame, then we can find the location of that landmark in other frames as well. Performing this operation for all facial landmarks, we obtain two sets of points, one for each image, which can be used to perform rigid registration.

We used OpenCV's implementation of template matching using `TM_CCOEFF_NORMED` similarity measure, defined as

$$\text{similarity} = \frac{\sum_i (T_i I_i)}{\sqrt{\sum T_i^2 \sum I_i^2}},$$

where T_i represents a pixel in a template and I_i its matching pixel in the patch of the target image. The template matching algorithm does not take rotation into account, but as we are only comparing small patches, the expected rotation is estimated to be small and negligible within these patches. After the points are tracked, rigid registration may still yield a noticeable amount of rotation, which is desirable.

Since our footage contains strong shadows that can make certain areas of the images almost completely black, template matching is likely to fail in such cases. For example, two images where the light source is on the left and the right will be difficult to align because of dark shadows on different sides of the face. For this reason, we only align images with similar lighting conditions. The images are manually arranged in a tree, such that the root node had most frontal lighting and the leaf nodes had least frontal lighting. The leaf nodes were separated such that only images with similar lighting conditions were siblings of some node with slightly more frontal lighting. When performing pairwise alignment, rigid registration was limited to facial landmarks that did not lie in shadows on any of the two contributing images. Alignment was propagated throughout the tree. Each child node was aligned to its parent, and the image represented by the child node was transformed accordingly. The process recursively repeated until all nodes were aligned.

As template matching compares a neighbourhood around a selected point in an image, we suspect that points with no notable facial features around them, e.g. a point on the forehead, would be difficult to track. To validate this, we perform a brute-force tests. We take an image of a face, transform it by a small amount and perform template matching between then for every pixel[2]. This gives us a set of new points on the transformed image, which we compare with the ground truth. We use the normalised distance between the estimated transformed points and actual transformed points to visualise the error. Figure 4.3b shows this visualisation, which confirms our hypothesis that only points around noticeable facial features are good for tracking.

[2]In practice, due to high resolution of the image, we only processed pixels in every nth row and nth column to avoid excessive processing, yet still obtain reasonable results.

This guides us to manually select points that we use for aligning images in our dataset, which are also visualised in Figure 4.3b. Using these initial points, we run our algorithm for several template sizes $s \in \{16, 24, 32, 48, 64\}$. Unfortunately, the results were, upon visual inspection by averaging, more blurry than the original ones. This means that stabilisation failed and facial landmarks were not successfully tracked. As with dense optical flow, there was a significant change in lighting that affected the appearance of selected facial landmarks to such an extent that tracking template matching was not possible, even though we already limited ourselves to comparing images with similar lighting conditions.

4.2.5. Detecting Facial Landmarks

Instead of choosing facial landmarks manually, we employ an automated system for detecting them. We resort to machine learning frameworks as they may be able to predict location of facial landmarks that are hidden due to shadowing effects [6, 23, 24, 30]. Due to its accessibility, we opt for using approach suggested by Bulat and Tzimiropoulos [6]. The authors already provide pre-trained models, which are simple to use. The algorithm returns, for a given image, a set of facial landmarks around the eyes, eyebrows, nose, mouth, and a path from the ears, along the jaw toward the chin. The algorithm can capture these landmarks from a single RGB image where a face is present under any orientation. Even if the face is facing sideways, the algorithm is capable of estimating the landmarks on the occluded side of the face, which can be useful for detecting faces where half of the face is in dark shadow.

We ran the algorithm on one of our datasets with almost no ambient illumination. Unfortunately, the algorithm performed failed to detect facial landmarks for all images. These results were once again subject to the effects of hard lighting effects. While the algorithm was trained on images that included shadows, they never were as strong as in our case, causing the algorithm to fail where parts of faces were almost completely black.

We test the method on another dataset, captured with a noticeable amount of ambient illumination, where all facial features were still clearly recognisable, even if in shadow. However, even for this dataset, the consistency of estimations of facial features was not coherent. This was most prominent for landmarks that stretch from the chin towards ears. This was no surprise given that the texture on the chin is homogeneous and therefore makes it difficult to determine exact position of landmarks, especially if the appearance of the chin varies due to changing lighting conditions. In conclusion, it seems that the algorithm is simply not accurate enough for fine alignment of images. Examples of failed alignment for this dataset can also be seen in Figure 4.4.

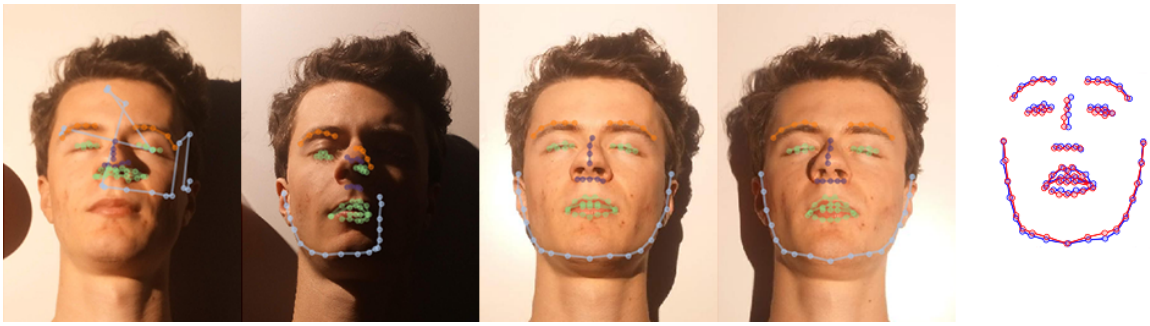


Figure 4.4: Results of detecting facial landmarks. The first two images represent failure to detect facial landmarks for a dataset with little ambient illumination. The following two images represent correctly, but inaccurately estimated facial landmarks. Note how points at the sides of the faces do not match with the actual contour of the face. The last, rightmost image shows misalignment of facial landmarks from the preceding two images.

Other researches who used this framework also stumbled across the same issue [58] and therefore used a more exact version of landmark detection by Kazemi and Sullivan [30]. Although future work should explore using this option for facial landmark detection with shadows, we eventually resorted to using sparse optical flow.

4.2.6. Sparse Optical Flow

Current approaches this far proved not to be working for alignment of images with different lighting conditions as lighting varied too much from one frames to another. To overcome this, we record a video of the face while the light source is being moved around. We estimate that having many frames with

small and gradual light changes between consecutive frames will make alignment easier as opposed to capturing only individual images with great changes in lighting.

Tracking To provide useful tracking data to rigid registration, we used sparse optical flow [4]. This method takes a set of points in a frame and tracks their movement to another frame. The method expects the movement of tracked points to be minimal, however, we use a pyramidal implementation. This means that tracking is first executed on down-scaled version of the images and then again at the original resolution. We use three pyramidal levels in total. This allows the movement to be somewhat bigger, which is useful since we want the algorithm to work on high-resolution images as well, where even small facial features may be spanning over several pixels.

The points to track are detected automatically using a corner-detection algorithm [45], which yields good features to track. These features are placed where the gradient of an image changes significantly, which happens in what we perceive as corners or borders. Consequently, homogeneously textured areas are avoided, which is consistent with our observations in template matching. A mask is used to restrict feature detection only on the face itself. The effects of this can be seen in Figure 4.3c. Points are mostly placed around the eyes, nose, and mouth. To automate the creation of such a mask, we use a face detection algorithm [52] which gives a bounding box of the given face. A mask is generated such that a circle is inscribed into the bounding box and scaled down by a factor of 0.6, such that it covers the aforementioned facial features.

Suppose we have a video of n frames, where F_i is the ith frame. Face and points to track are only computed on the first frame of a video, giving a set of points P_0. For all the following frames, sparse optical flow is used to track the features from frame F_i to its following frame F_{i+1}. In other words, given a set of points P_i, running sparse optical flow yields a set of tracked points P_{i+1}. These two sets can be used to perform rigid registration and obtain a transformation T, which is a combination of rotation and translation. Sparse optical flow may fail to track all the points correctly. Those that failed to be tracked are not used for rigid registration. Instead, they are transformed according to T to obtain their expected location in frame F_{i+1}, which is then becomes a part of the set P_{i+1}, that can, in turn, be used to perform tracking on the following frame F_{i+2}.

Removing Outliers Often times, sparse optical flow does not fail completely, but may track points to an invalid location that is far off of where it should be. Such outliers contribute equally to computing rigid transformations as any other point, negatively affecting our stabilisation algorithm. We detect and remove these using RANSAC [13]. This operation is applied after sparse optical flow detects new points P_{i+1}. It is an iterative process; first, it only selects three random pairs of matching points from P_i and P_{i+1} and uses them to compute a rigid transformation T'. Then, for each remaining pair of points $p_{P_i} \in P_i$ and $p_{P_{i+1}} \in P_{i+1}$, an error value is computed as $||T' p_{P_i} - p_{P_{i+1}}||$. If the error value is below a threshold, the pair of points is added to a set S along with the three initial pairs used to compute T'. The pairs of points in S are used to compute a better rigid transformation T^*. An error measure of T^* is computed as a mean squared error (MSE) of the pairs of points in S:

$$\text{MSE} = \frac{1}{|S|} \sum_{(p_{P_i}, p_{P_{i+1}}) \in S} ||T^* p_{P_i} - p_{P_{i+1}}||^2.$$

If MSE is below a set threshold, then the model T^* is returned as the best fitting model. Otherwise, the processes is repeated by sampling three new random pairs of points. After a set number of iterations, the model with the lowest error is returned. Each iteration increases our confidence in finding the most suitable transformation. We only require 13 iterations to obtain the best model with a 99.5% confidence assuming at most 30% of the points are outliers[3]. When a model is obtained, pairs of points that are not in S are considered outliers. These are treated the same as points that failed to track. An example visualisation is depicted in Figure 4.3d.

Stabilisation of Recordings The algorithm was tested on a video, where a light source was moving slowly around the face. To enhance visibility and improve the tracking, we take a footage with a sufficient degree of ambient illumination for facial features to be clearly visible, even if in shadow. The moving

[3]The reader is referred to the original paper for details about this statistical analysis [13]

light source caused self-occlusions and cast shadows which moved over the face as the light source was moved. Unfortunately, tracking points that came in contact with such a shadow were often tracked along with it instead of being fixed at a landmark, rendering stabilisation impossible.

Considering all the previous results, we conclude that our method is not suitable enough for tracking of points under changing illumination. Therefore, we research the possibilities of changing the capture method such that tracking and stabilisation would be performed under constant illumination. This is further discussed in Section 4.3.1. To facilitate such stabilisation, we test our tracking algorithm on a video with constant illumination.

The algorithm performed well for a video captured with a webcam under constant illumination. The video features a subject moving their head left, right, up, and down at an approximately constant distance away from the camera, while also rotating the face around the axis of the line between the subject's head and the camera. The algorithm executed successfully for a 1920×1080 video and equally well for the resolution of 640×480. Most tracking points were distributed at expected locations around eyes, nose, eyebrows, and mouth and tracked correctly. Then, for each frame, the acquired rigid transformation was inverted and applied to the same frame to stabilised it in respect to the first frame, where the faction landmarks were first detected. If tracking worked correctly, the face should no longer move in the stabilised video.

This, indeed, was the case after performing a qualitative assessment. Best results were achieved when the subject was looking directly at the camera when moving their head around the space. If the subject did not look directly at the camera, this would result in transformation of the head in 3D space that could not be fully captured by an image-space rotation when projected to the perspective of the camera. Moreover, although outside of the scope of our application, this method—with a surprising degree of accuracy—also managed to track and stabilise faces with glasses, moving the head closer to or away from the camera, and overcome problems such as blinking or partial occlusions of the face.

For another test, we captured a video of a subject sitting still, to see how we can stabilise smaller head movements. This acquisition method resembled that used for RTI, except there was no light used so that lighting was consistent throughout the recording. Since estimated head movements are small in such a setting, we require a high-resolution footage of at least $1080x1920$ to capture them. The algorithm managed to successfully track facial features in this footage as well, successfully stabilising it. We verified this by averaging frames of both the original and the stabilised footage, and we observed that the latter was noticeably less blurry.

Accuracy A qualitative analysis indicated that the algorithm performs well when video frames are consistent. The analysis was conducted on an image sequence created by taking a single image of a face, sized 1920×1080, and duplicating it 85 times. Each frame following the first was translated by up to 6 pixels in any direction and rotated by up to 2° in respect to the previous frame. We then applied our stabilisation algorithm. Three error metrics were computed for each frame. The first two measure the distance between the ground truth and estimated translations vectors and rotation angles. The third encapsulates both transformations at once by measuring the average distance between tracked points in the current frame and tracked points detected in the first frame but transformed with the ground truth transformation of the current frame. After conducting the test, we observed that the rotation error stayed below 0.1° at all times, translation error did not exceed 0.3 pixels, and the point distance error metric reach at most 3 pixels. Unfortunately, the algorithm performed sub-optimally for translation and rotations of greater magnitude[4]. Nevertheless, our subjects always attempt to remain still during the acquisition period, therefore, stabilisation for large facial movements is irrelevant.

4.3. Face Capture

To perform relighting, we require a set of images where faces are illuminated from different directions. However, our acquisition process suffers from misalignment, which can only be improved if illumination is constant. As described in the previous sections, traditional tracking and stabilization methods fail under illumination changes.

To address this issue, we present a novel acquisition approach, where we capture a video in which our subject is exposed to a flashing light. This allows us to capture frames with alternately illuminated

[4] One could study the effects of increasing the number of pyramidal levels of the tracking algorithm to improve this, if need be.

face as well as with constant lighting conditions. The former can be used for relighting, while the latter are used for stabilisation.

We detail this acquisition process in Section 4.3.1, and describe how to separate frames that illuminate faces and those where lighting is constant in Section 4.3.2. The application of the stabilisation algorithm is discussed in Section 4.3.3, whereas Section 4.3.4 explains how to obtain the direction of the flashing light. Finally, Section 4.3.5 elaborates on an unsuccessful attempt to remove indirect illumination from the capture frames.

4.3.1. Acquisition Setup

Our setup consists of a camera, which can be an everyday smartphone, mounted on a tripod or any other holder (a simple version can be made of cardboard). The camera is directed at a subject, who should sit as still as possible, ideally leaning on a headrest to minimise head movement. A reflective sphere for detecting light direction should be stably mounted in the background. The sphere could be a simple round and reflective object found at home, such as a piece of garden decor or Christmas tree bulb ornament. An example setup is demonstrated in Figure 4.5.

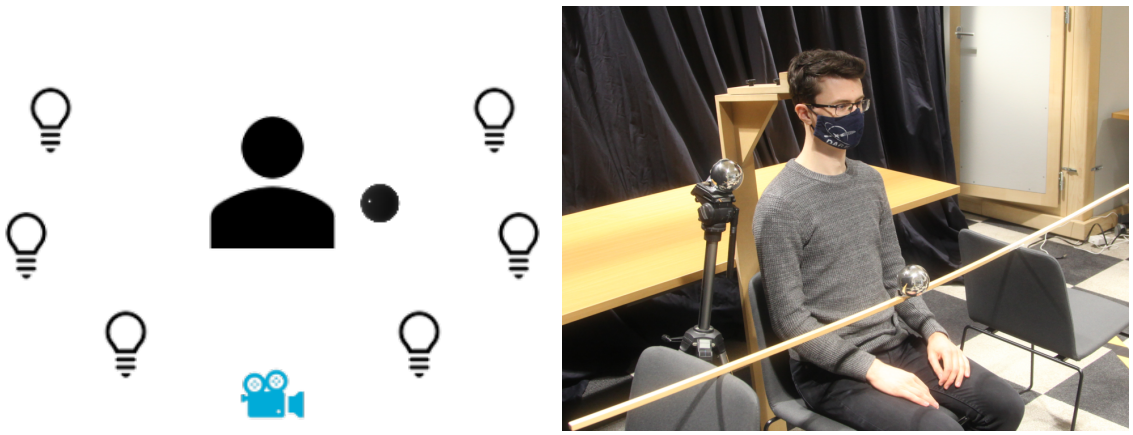


Figure 4.5: Left: Schematic acquisition setup. A camera captures the subject's face a flashing light source is manually moved around the subject. A simple reflective sphere is placed near the subject to retrieve the light direction from the images. Right: Studio acquisition setup. We placed two reflecting spheres for testing purposes, but we only require one.

Any automatic enhancement by the camera, such as auto-stabilisation, auto-exposure correction or similar should be disabled, if possible. This will allow for capturing raw footage, without any unpredictable and undesirable adjustments. The reflective sphere should be in the camera view, but it should not occlude the face. Furthermore, the sphere should not cast shadows on the face when we add a light source and the face should not cast shadows on the sphere. We find that positioning the sphere to the side of the face or below it normally works well.

During the acquisition, the subject is illuminated by a light source, flashing at a frequency of 5 Hz. This light source can be another smartphone with a flashlight[5]. The acquisition should be held in a dark environment such that illuminations by the light source are clearly visible on the face. On the other hand, the environment should not be completely dark, as we will use footage without an extra light source for stabilisation; therefore, all facial features should still be clearly visible without it.

We obtain illumination data by recording the subject at 30 FPS while using a flashing light source. The light source should be moved around the subject in a way, that uniformly samples as many points in the hemisphere in front of the subject. The light source should be at a constant as possible distance from the head, but always facing toward the subject. In our experience, repeated up-and-down movements while gradually moving from the left of the subject to the right worked well. We kept the light source about one meter away from the face. The acquisition process need not be long, one minute is more than sufficient. The result of this process is a video of the subject's face, which is, in some frames, illuminated, and in the others, not.

[5]We used application Strobe Light by Brandon Stecklein from Google Play Store.

4.3.2. Bright/Dark Frame Extraction

As we record at 30 FPS and flash at 5 Hz, each flash spans over approximately six frames, three of which are where it illuminates the scene and three where it is turned off. However, the camera shutter and the flashing light are not synchronised, therefore, not all frames are completely lit or dark. For the same reason, we cannot simply take every third frame and consider it the brightest or the darkest. Instead, we present a robust method of extracting the brightest frames where the face is illuminated, called bright frames, and the darkest frames where it is not, called dark frames.

The idea of the method is to compute average brightness of each frame and then find local minima, which are the dark frames, and local maxima, which are the bright frames. Thus, we find the minima by applying a sliding window of three frames. If the middle frame's brightness is minimal, we classify that frame as a dark frame. We find bright frames in a similar manner.

Due to errors in capturing, the directional flashing light may not always be facing toward the face. As this decreases the quality of our dataset, we wish to exclude these frames. We estimate that in such a scenario, the brightness of the bright frame will be comparably low to all others. We therefore include an amplitude check and discard any frames where the amplitude in brightness between the preceding and the following bright or dark frame is below a certain threshold.

We expect the average distance between consecutive bright and dark frames to be around three frames (i.e. every third frame is either bright or dark). Since the brightness may fluctuate, we account for a margin of two extra frames, one for the error of the bright frame, and one for the dark frame. But if two consecutive bright and dark frames are more than five frames apart, this indicates and outlier due to an acquisition error. Hence, we remove both frames.

With these steps, we can robustly extract bright and dark frames as shown in Figures 4.6 and 4.7. We do, however, observe several drawbacks in the process. First, the average brightness of dark frames is inconsistent and fluctuates. This is presumably due to automatic exposure or brightness adjustments of smartphone cameras that cannot be always turned off. This variation has not proven to be significant and does not affect stabilisation, which we discuss in the following section. Thus, we do not consider this issue further. Second, we note that the brightness of bright frames may vary too, but this is expected as different parts of faces are highlighted for different illumination directions. Regardless, we can still argue that the brightness may not be consistent because the person moving the light source may not have always held it at the same distance away from the face, resulting in artefacts due to light fall-off. Because we cannot rely on the average brightness of the frame in this case, this issue would require a more sophisticated solution such as adjusting the brightness of the frame based on perceived brightness of a control object placed in the scene. This did not yield significant artefacts when relighting and therefore remains a point for future research.

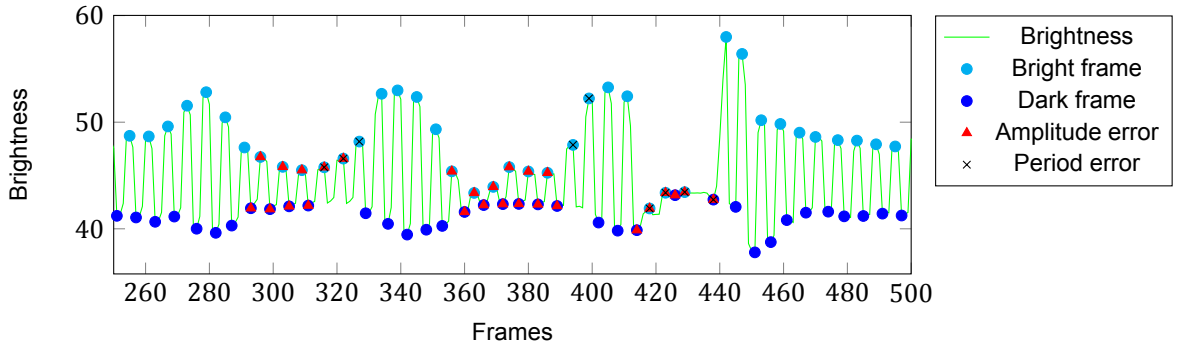


Figure 4.6: Brightness of a subset of captured frames. Approximately every third frame was classified as either bright or dark. Several frames were filtered out due to an invalid amplitude or period.

We also experimented with recording at different frame rates and flashing frequencies. If the flash frequency is higher than one sixth of the recording frame rate (as in our case), then the captured frames are somewhere in between bright and dark, and if we extract them using the same algorithm, the brightness of the extracted bright and dark frames is inconsistent. We could flash at a lower rate than presented here, but this would not improve the results and would increase acquisition time, which is undesirable. Finally, we do not recommend recording at 60 FPS, which some smartphones are capable of, because, due to a dark environment, the camera cannot capture enough light to produce

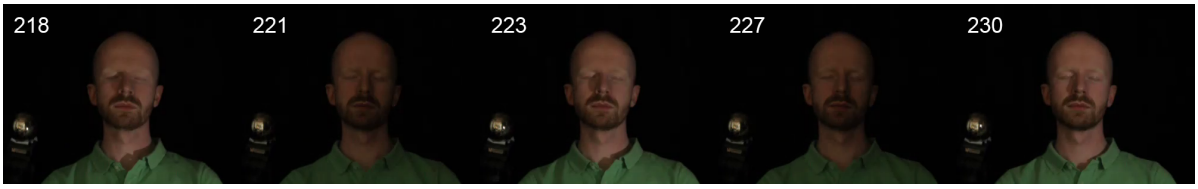


Figure 4.7: A sequence of extracted bright and dark frames with the frame numbers above. Note how the spacing between them is not constant.

images of sufficient quality.

4.3.3. Stabilisation of Bright and Dark Frames

With the extracted dark frames, it is simple to run the stabilisation algorithm since lighting conditions are stable enough. However, we eventually require bright frames to be stabilised as well. To achieve this, for every bright frame, we simply take the transformations of two neighbouring dark frames and interpolate between them. The interpolation takes into account the spacing between these frames, that is, the number of non-bright and non-dark frames between the three selected frames. The computed transformation is then applied to bright frames in order to stabilise them.

To test the accuracy of stabilisation ran on every sixth frame, we employed the following test. We recorded a video of a face with consistent lighting conditions and ran the stabilisation algorithm for every frame. Then, we only kept every sixth frame, and ran the stabilisation again. We computed translation and rotation errors for every sixth frame between the two methods. Translation error stayed below 1.2 pixels and rotation error never exceeded $0.35°$ for a 540×960 video. With this, we deem the approach suitable. The interpolation of in-between frames could be problematic if translation or rotation would rapidly change direction, but since our subjects try to stay still, the error there is assumed to be negligible.

4.3.4. Light Direction Extraction

Our relighting approach, fully described in Chapters 5 and 6, requires direction of incoming light to be known for each bright frame. This direction is computed with the help of the reflective sphere used during acquisition. It exhibits specular highlight where the light from the flash source bounces off of it and into the camera.

We ask the user to locate the sphere in one of the captured frames. We obtain its radius r, measured in pixels, and its center point O. We assume the sphere is fixed in place so that its location and radius can be reused throughout the captured video. If this is not the case (e.g. due to automatic camera stabilisation which causes the footage to move), then the footage needs to be stabilised in respect to the sphere. This can be done using external software or the sphere's location needs to be corrected manually for every frame. Note that this process needs to be executed before stabilisation is applied.

The highlight on the sphere can span over several pixels. As our data is not in high dynamic range and subject to hardware imperfections, we cannot simply take the location of the brightest pixel and consider it the location of the highlight. Instead, we apply thresholding over 0.85 over all RGB channels to obtain a set of pixels within the bounds of the sphere that the highlight consists of. An average of the pixels' location is computed, weighted by their individual brightness computed as the average of the RGB channels. The result is considered to be the highlight location in image-space. Figure 4.8 shows an example visualisation. It is important that the sphere does not reflect any bright objects in the environment other than the flashing light source, as this could affect the method.

Our method of computing the incident light direction is similar to that of Mudge et al. [36]. We first normalise the highlight location H to a $[-1, 1]$ range for both X and Y axis, relative to the center of the sphere and its radius. We then compute a normalised 3D vector $\mathbf{v} = (x, y, z)$ that describes the light direction:

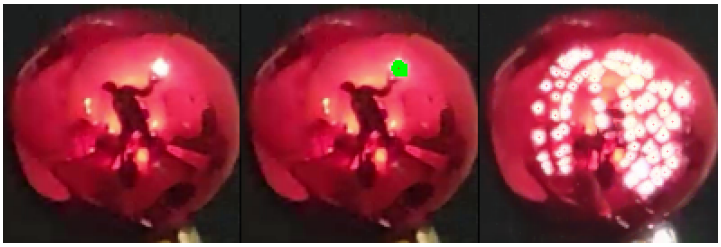


Figure 4.8: Highlights are detected in reflective spheres to exact incidence light direction. The leftmost image shows an example of a captured sphere. Center image shows thresholding applied (pixels in green) and highlight center (red cross). The rightmost image is a visualisation of all sampled light directions in the dataset.

$$
\begin{aligned}
p &= \arctan 2\, H \\
q &= 2 \arcsin |H| \\
z &= \cos q \\
x &= \cos p * \sin q \\
y &= \sin p * \sin q.
\end{aligned}
$$

The computed vector can be converted into any representation of choice, as described in Chapter 2.

This process can reliably extract light direction from images of a reflective sphere, however, the method is not without issues. First, based on a test with a synthetically rendered sphere, the average error of the method is around four degrees and it mostly depends on how accurately the user estimates the location of the sphere. Fortunately, the sphere is only estimated once, thus, the error, if present, is consistent for all the extracted highlights. Second, the method assumes that the light source and camera are far away and that the incident light angle does not change for different points in a frame. This is not entirely correct, but cannot be accounted for with the current acquisition setup. For future work, one could study the usage of two reflective spheres, calibrated to the scene, and using the disparity between the extracted light directions to compute 3D location of the light source. Yet another problem is that highlights at the side of the reflective sphere are processed using a 2D image-space averaging technique, whereas, due to the curvature of the sphere and projection of the camera, the extracted angle may not be correct. However, this error would get more prominent for highlights at the edges of the sphere, but as our sampling is limited to the frontal hemisphere, highlights always have a more central position; thus, we estimate this to be negligible.

4.3.5. Ambient Subtraction

Ambient illumination and the intensity of the flashing light are uncontrolled in our method. To improve the results, we wish to subtract ambient illumination from our bright frames. Because ambient lighting is captured in dark frames, this operation appears to be as simple as subtracting a dark frame from a bright frame[6]. This works well for synthetic, rendered cases. For captured images, however, this method does not work. It results in a blue faces, visualised in Figure 4.9. The reason for this is that the camera performs adjustments that generally improve image quality, such as automatic white-balancing and exposure correction. However, these adjustments tinker with the balance of colours, which are then no longer calibrated across all our frames. This is especially prominent since we use a flashing light, which forces the camera to adjust to new lighting conditions every few frames. The effects of this can be observed in the aforementioned finding of how average brightness of the dark frames is not consistent; the drop in brightness of a dark frame is correlated to the average brightness of the preceding bright frame.

This issue could be mitigated using more sophisticated hardware, but we limit ourselves to a low-cost setup that can be performed at home. We also attempted to solve this in post-processing. We attached a white piece of paper to the scene with the subject. Before subtraction, we scale each of the RGB channels of an image, such that the average colour of the paper is a shade of gray. This correction is performed for both the dark and the bright frames involved in the subtraction. While this

[6] Subtraction is done in linear space.

Figure 4.9: The difference between a bright (far left) and a dark (center left) frame does not approximate well the contributions of direct/indirect illumination due to uncontrollable self-adjustments of the smartphone camera under rapid illumination variation (center right), even with simple colour adjustments (far right).

did improve the results slightly, as seen in Figure 4.9, it did not solve the problem. We recognise that this is a very simple method, that would need to be improved in order to function sufficiently well. Due to time limitations, we resort to simplicity and use the extracted bright frames as they are.

5

Constructing Reflectance Maps

The extracted and stabilised bright frames provide sufficient data to capture our subject's appearance, however, it needs to be restructured before it can be used for relighting. To obtain a useful data structure that encodes reflectance information of individual points on the face, we rely on so-called reflectance maps. This chapter explains in detail what information reflectance maps hold and how they are constructed.

5.1. Construction

Authors of the first Light Stage already used reflectance maps to encode the reflectance functions for different points on the face that they captured [10]. Our approach is similar, except that, instead of a rotating gantry, we manually move the (flashing) light source around to sample reflectance information of the face. Thus, we also opt for usage of reflectance maps.

We implement them as 32-bit floating-point RGB images of size 64×32 that share the same coordinate space as environment maps. One reflectance map is constructed for every point on the face and captures light reflectance information for that specific point. Each point in a reflectance map is associated with a light direction and describes the colour of reflected light in the viewing direction if a white light source was present at the associated light direction.

We populate reflectance maps with colours sampled from the stabilised bright frames. Each pixel in these frames represents a point on the face, which, in turn, is represented by one of the reflectance maps. In total, we have as many reflectance maps as is the resolution of bright frames, one for each pixel. In practice, we crop and resize the these frames to 256×256 or 150×200 pixels, primarily to limit memory consumption. Furthermore, each such frame represent the appearance of the face under some known and extracted light direction, which—if mapped to the coordinate space of reflectance maps—falls within a certain associated pixel. Therefore, each pixel in a captured frame can be mapped to a linked reflectance map's pixel of the associated light direction. With this mapping, we transfer pixel colours from captured frames to reflectance maps, which results in sparsely sampled reflectance maps. A conceptual visualisation of this process can be seen in Figure 5.1.

Just as our captured bright frames, our reflectance maps, too, capture not only the BRDF of the face, but also all effects of indirect illumination such as ambient illumination, self-shadowing, subsurface scattering and global illumination effects [10]. These can be clearly observed for a synthetic, render model with dense sampling, seen in Figure 5.1.

While dense sampling our reflectance maps would be ideal, we normally only have 150-200 sampled light directions due to limitations of the acquisition process. To overcome this, we interpolate the colours of our samples to approximate a dense sampling. This process is described in Section 5.2, but yields many artefacts with shadows, which we address in Section 5.3. Further improvements to interpolated colours are described in Sections 5.4 and 5.5, while Section 5.6 lists methods of extrapolating existing colours to extend the sampled region.

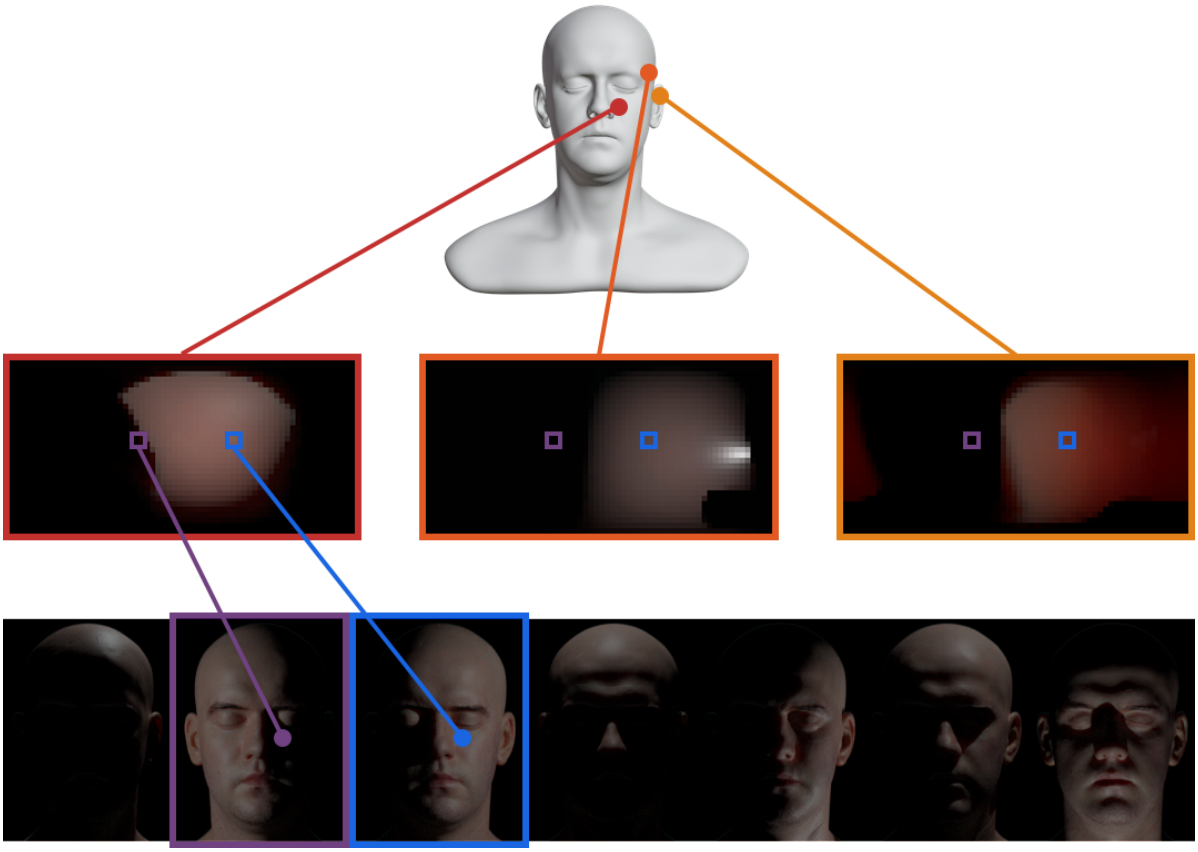


Figure 5.1: Schematic representation of reflectance map construction. Three reflectance maps (middle row) represent each a point on the face (top): cheek (left, red), temple (center, orange), and ear (right, ochre). Note the white specular reflections on the temple and subsurface scattering on the ear. Colours in reflectance maps are sampled from captured bright frames (bottom row). For clarity of visualisation, only two frames are highlighted: purple (second from left) with light source on viewer's left and blue (third from left), with light source on viewer's right. For the same reason, only sampled points from the left (red) reflectance map are linked to the two captured frames.

5.2. Interpolation

To estimate missing samples in the reflectance maps, we resort to a Delaunay triangulation of the sampled light directions, performed in image-space. This yields a set of triangles with sampled colours at their vertices. Since we do not capture lights direction from behind the face, we do not have to worry about triangles wrapping around the map's borders. Using barycentric interpolation, we can colour non-sampled pixels within the triangles.

To mitigate the difference between linear edges in image-space and projected edges in equirectangular projection, we use Spherical Barycentric Interpolation [7]. With this approach, vertices of the triangles, defined by light directions, are connected via the shortest path on their unit sphere. Similarly, barycentric coordinates are now calculated using areas of the sphere's surface rather than areas of a 2D image-space triangles. The improvement of this method over linear image-space interpolation is shown in Figure 5.2.

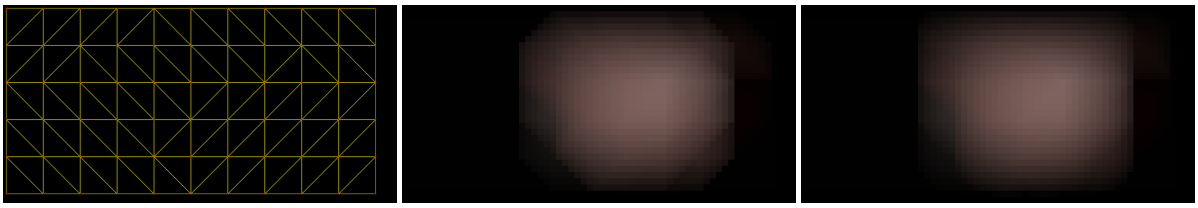


Figure 5.2: Comparison of reflectance map interpolation methods for a synthetic case with grid-like sampling (left to right): triangulation, linear barycentric interpolation, spherical barycentric interpolation. Notice how the latter exhibits smoother transitions.

We also considered interpolating missing samples using the Poisson equation, however, the method performs only in image-space. Hence, it lacks spherical adjustments that we found to improve image quality.

5.3. Visibility Maps

Pure barycentric interpolation exhibits visible artefacts in form of ghost shadows [56] when used for relighting. On images relit with a points light source, we would expect the shadow to be clear, with a distinct border between the shadowed and lit areas. Instead, we observe multiple such borders of different intensities. This happens when interpolating the colours of two samples in a reflectance map, one dark, sampled in shadow, and one bright, sampled under direct illumination (lit). The pixels between them follow gradient mid-tones, as if the shadow slowly faded away, whereas they should exhibit a hard boundary between the lit and shadowed areas instead.

The idea to solve this issue is to not use such a linear gradient but, instead, identify a point between the two pixels where the lighting changes, i.e. a point where the surface point is no longer visible from the light source. Thus, instead of a gradient, we formulate a step function, with two colours, the one of the lit pixel and the one of the pixel in shadow. This allows us to define precise shadow boundaries and avoid ghost shadows. In the rest of the section, we discuss technical implementation details of the approach.

To obtain light directions, for which specific surface points are no longer occluded by the light source, we fit a 3D model to the face [23, 24] and compute a visibility map for each reflectance map. Visibility maps are greyscale images that share the same size and coordinate system as reflectance maps. They contain a one if the light directions corresponding to the pixel are completely visible, a zero if they are blocked, and in-between values for partial visibility. As the 3D model does not span over the entire face, this approach can be used only on parts of the face which the model covers. For the rest, we fall back to using plain spherical barycentric interpolation. To avoid visible seams caused by the two different approaches, we smooth the contributions of the 3D model around the edges.

Visibility maps allows us to improve the interpolation when one or two of the triangle vertices are not visible. To simplify, we first consider a binary visibility map - one can imagine thresholding the values with 0.5. Without loss of generality, we consider that v_0 represents the direction of a non visible light source while lights at v_1 and v_2 are visible. For every pixel inside the triangle that represents a non visible light direction we use the value of v_0, while visible directions interpolate between $v1$ and $v2$ using spherical barycentric coordinates. This process is depicted in Figure 5.3, left.

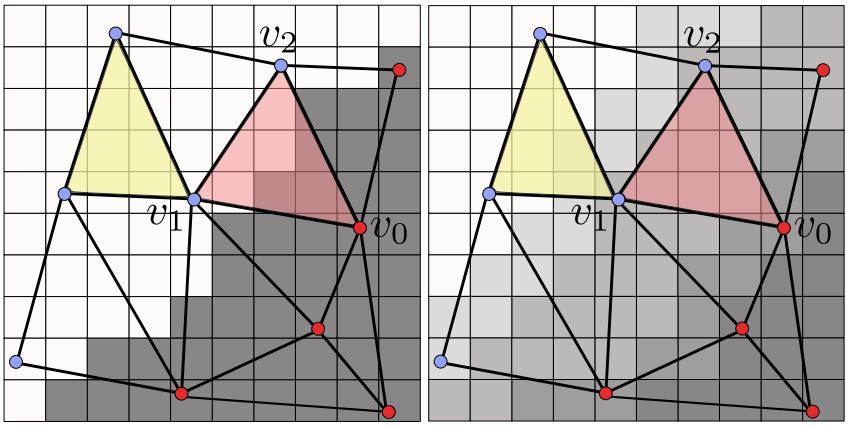


Figure 5.3: Using visibility to interpolate inside a triangles. Visible, not visible, and partial visible directions are marked as white, dark gray, and intermediate gray values, respectively. Left: Reflectance map with five visible light directions (blue vertices) and five invisible (red vertices). The yellow triangle uses regular interpolation. For the red triangle, white pixels will mostly rely on blue vertices, while gray pixels rely on the red vertex. Right: In practice, we use a smoothed visibility map and generalize this interpolation scheme.

This approach assumes that the 3D model perfectly matches the face. However, since it is just an approximate model, we expect some light directions to be wrongly classified. Considering Figure 5.3, left, if direction v_0 was wrongly classified as being visible, the red triangle would be interpolated ignoring the visibility map and lose the sharp transition. If a neighboring reflectance map would classify the

samples correctly, this issue would be accentuated.

To mitigate this problem, we smooth the original non-binary visibility map with a 3×3 box filter. Further, we propose an interpolation between both interpolation modes: not using the visibility map (regular barycentric interpolation) and taking the visibility map into account as described above. If all vertex visibilities are the same, we just use regular interpolation, while a large difference of visibility on the vertices implies that we resort to the visibility map. Specifically, we use the difference between the minimum and maximum as interpolation weight between the two methods. Please note that working with the difference means that in some cases there might not be any vertex with a visibility below 0.5. Instead, we always assume that the minimum value indicates being in shadow, the maximum lit. The visibility-map colour is computed for both cases where the third vertex is classified as lit and in shadow, and interpolated based on the relative distance between itself and the minimum and maximum. Figure 5.3 illustrates this concept, and Figure 5.4 shows an example of a real reflectance map and the visibility maps.

Smoothing the visibility map leads to a smoother interpolation but also prevents some sharp shadows from being detected. Nevertheless, since we cannot fully solve the misclassifications with an approximated 3D model there remains a trade-off.

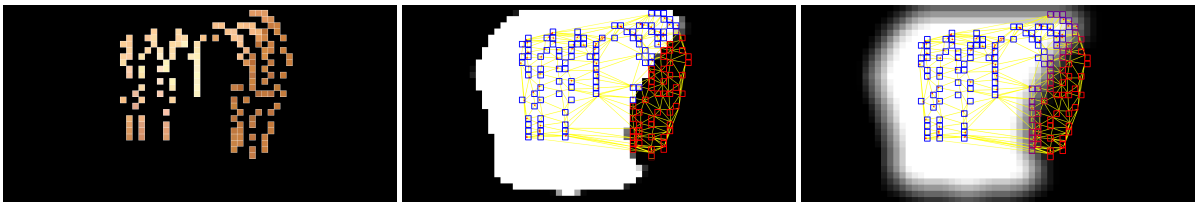


Figure 5.4: Comparison of visibility maps. Left: sampled light directions. Center: sharp visibility map. Right: smoothed visibility map. The outline colour of samples corresponds with the corresponding values in visibility maps.

5.4. Light Adaptation

The current interpolation method currently does not respect lighting changes when interpolating colours within a triangle. To improve on this, we adjust the brightness of the colour within interpolated triangle vertices according to the angle of incoming light.

More specifically, we first use the 3D model to estimate surface normals. Since the model does not span over the whole captured face, we smooth the contribution of the model on the borders to avoid any visible seams. Then, as each pixel in reflectance maps represents incoming light direction, we can compute the cosine of the angle between the normal and each light direction. This so-called cosine factor corresponds with the percentage of the amount of light received by a patch of surface due to foreshortening. To account for this effect, we adjust the interpolation within reflectance maps by compensating for this factor. That is, we divide vertex colours by the corresponding cosine contributions, interpolate, and then multiply by the cosine factor of the interpolated position.

We do assume that the direct illumination is dominant. This also holds for ambient light, since the natural solution of subtracting the dark from the bright frame was not successful.

The previous solution requires a refinement for when any of the cosine factors is negative (or clamped to zero), i.e. the light direction is behind the surface. Here, the correction would be meaningless. Nevertheless, it does not mean that the captured colour is necessarily black, as indirect illumination is recorded as well. Such local occlusions are important to handle because realistic skin appearance requires capturing subsurface scattering effects. Nevertheless, if we use a similar approach as described for the visibility map, we could obtain negative values and even cause discontinuity artefacts. Instead, we distinguish three separate cases. First, if all three vertices have a negative cosine factor, we just perform interpolation as described before. The second case covers two vertices with negative cosine factor. The third will be one negative cosine factor. As these cases are complex, we start with the intuition.

The idea is to use the value at the vertex with negative cosine factor to approximate the indirect contribution at the other vertices. Specifically, we remove the indirect illumination, perform the cosine correction for the direct illumination, and then add back the indirect component that is slowly faded out towards the two lit vertices. To simplify, let us consider a 1D case of interpolating between vertex v_0

with only indirect illumination and v_1 that has both direct and indirect contributions. We use the colour at v_0 as an approximation of the indirect component at v_1. Hence the contribution of the direct component to a point p becomes:

$$c_{\text{dir}} = (c_1 - w_0 c_0)\frac{\delta_p}{\delta_1} + w_0 c_0, \tag{5.1}$$

where c_i, w_i and δ_i are, respectively, the colour, barycentric weight, and cosine factor at vertex i. Finally, we can compute the colour at point p as $c_p = w_1 c_{\text{dir}} + w_0 c_0$.

We can now extend this idea to 2D to interpolate inside the triangles. If two vertices have negative cosine factors, say v_0 and v_1, we can first interpolate their indirect contributions as $c_{\text{ind}} = (w_0 c_0 + w_1 c_1)$ and their combined weight as $w_{\text{ind}} = w_0 + w_1$. The contribution of the direct illumination can then be defined similarly to Eq. 5.1, with c_p computed as before:

$$c_{\text{dir}} = (c_2 - w_{\text{ind}} c_{\text{ind}})\frac{\delta_p}{\delta_2} + w_{\text{ind}} c_{\text{ind}}. \tag{5.2}$$

For the final case of v_0 and v_1 having a positive cosine factor, we first interpolate between v_1 and v_2 and compute $c_p = (w_1 + w_2)c_{\text{dir}} + c_0 w_0$, where the direct contribution c_{dir} is given by:

$$c_{\text{dir}} = \left[\frac{w_1(c_1 - w_0 c_0)}{\delta_1} + \frac{w_2(c_2 - w_0 c_0)}{\delta_2}\right]\frac{\delta_p}{w_1 + w_2} + w_0 c_0. \tag{5.3}$$

Visually, light adaptation helps brighten or darken reflectance maps to make them appear smoother and more believable. This is especially helpful in areas where visibility maps change and a single colour is stretched over a larger area. Figure 5.5 shows these effects and compares described interpolation methods with ground truth data for a synthetic example.

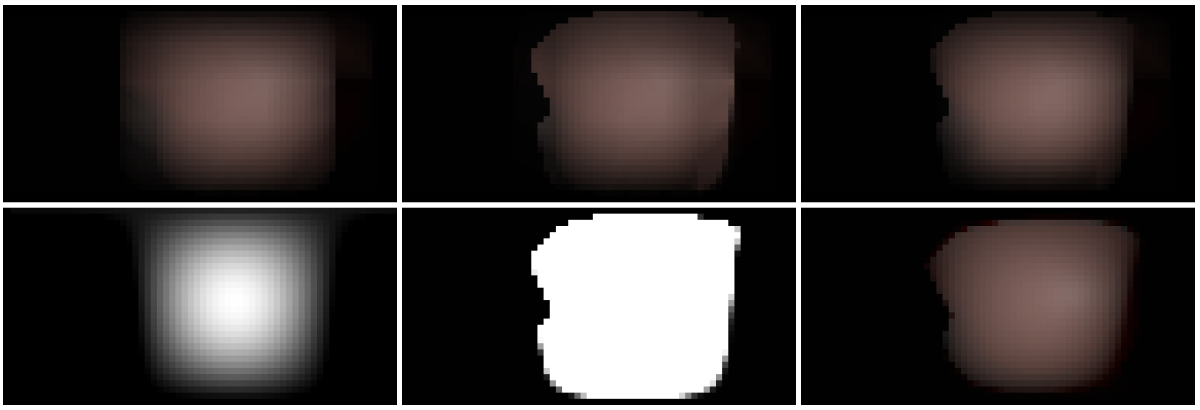


Figure 5.5: Comparison of reflectance map construction methods for a synthetic case. Top row (left to right): spherical barycentric interpolation, added sharp visibility maps, added light adaptation. Bottom row (left to right): visualisation of cosine weights, visibility map, ground truth.

5.5. Adjustments

We apply gamma correction and all calculations are done in linear sRGB space. Similar to Debevec et al. [10] we normalise reflectance maps after they are computed. Normalisation accounts for the fact that in an equirectangular projection, pixels at the poles represent smaller areas. We multiply the pixel intensity by the sine of its corresponding latitudinal coordinate, such that values at the poles are close to zero, whereas the values at the equator remain mostly unchanged.

5.6. Extending Sampled Area

As the current sampled area is limited to the frontal hemisphere, we investigated methods that would extrapolate reflectance information outside of the sampled region. To do so, we describe several attempts of adding additional samples to our interpolation.

The intuition behind the first approach was to stretch the sampled colours in lit region (according to visibility maps) to fill the rest of the region. We attempted to achieve this by adding additional synthetic black samples around the lit region. These points would be included in the triangulation and interpolation using visibility-maps, which would cause the colours of the lit vertices to fill most of the lit region. More specifically, the points were added to reflectance maps, on locations which the corresponding visibility maps indicated to be in shadow, yet still next to lit areas. If the sampled region already covered a shadowed area where new points would be added to, then these points would just be added a constant distance away from the sampled region instead. To allow interpolating over the whole lit region if it stretches along a border, we also add points outside the reflectance map. As the number and location of extra points depends on visibility maps, triangulation also needed to be performed for each reflectance map individually.

When tested, this method performed poorly, causing threefold errors. First, triangulation varies per reflectance map, diminishing consistency across neighbouring reflectance maps, which resulted in visible seams. Second, some triangles connected only the added black points, rendering their contribution void. Third, some undesired triangles appeared in shadow areas, stretching the effects of sampled colours inconsistently over it.

To mitigate the effects of inconsistent triangulation, we add triangles solely based on the sampled points. More precisely, to obtain the newly added points, we move the points on the convex hull of the sampled region by a set distance away from the center of the sampled region. Since the extra points are no longer limited to the shadow region, we instead only limit their placement within the lit region. Thus, when moving the points away from the center, we only do so for as long as they are still within the lit region.

This method performed well for individual reflectance map, however, it was limited to the areas of the face which visibility maps could be computed for. This caused visual seams when relighting.

Due to all limitations, the final approach consisted of forcing a linear interpolation of colour within triangles consisting of extra black points. The extra points were added at a set distance around the sampled region just as in the previous method, but without limiting them to lit regions. This method produced results without seams, and its results can be seen in Figure 5.6. However, this is only an approximation of reflectance beyond the sampled region, which can be useful for catching some light from side angles. On the other hand, the effects of the approximated regions can be observed in terms of static highlights that fade away instead of moving along with the light. Future research should investigate possibilities of improving the 3D model to capture the whole face and enhancing reflectance estimations beyond the sampled region.

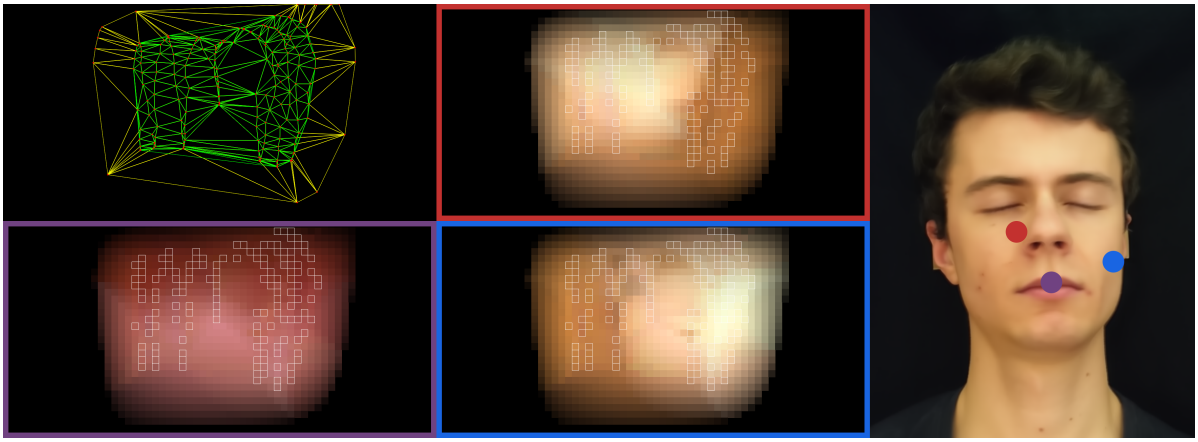


Figure 5.6: Top left: triangulated light source samples (green) with extra points (yellow). Other images: reflectance maps for different points on the face (right). Reflectance maps (displayed here in linear space) were created using spherical barycentric interpolation, a smoothed shadow boundary, light adaptation, and normalisation.

6

Relighting

With a reflectance map per pixel, we can easily perform relighting on static images. After describing this method in Section 6.1, we list optimisation steps to make the algorithm run in real-time in Section 6.2. Finally, we describe a proof of concept for relighting non-static footage using a reenactment approach in Section 6.3.

6.1. Relighting of Static Images

We combine environment maps with reflectance maps and by doing so convolve incoming light by our reflection function to obtain a relit image. For ease of computation and to avoid aliasing artefacts [10], environment maps should be of the same size as reflectance maps. Each pixel of an environment map describes incoming light from a specific direction, whereas each pixel of a reflectance map represents light reflected from a point on the face, assuming incoming white light from the associated direction. Thus, by multiplying the two, we obtain the colour of the reflected environment. Since light is additive, we can perform this operation for every pixel of environment and reflectance maps, and sum their contributions to obtain the colour of environmental light reflected from all directions of a point on the face. This process can be written as

$$O(r, e) = \sum_{l \in \mathcal{L}} r_l e_l, \tag{6.1}$$

where $O(r)$ represents the total reflected light for reflectance map r and environment map e, $\mathcal{L}$ a set of pixels for reflectance and environment maps, and r_l and e_l the sampled pixel therein. Since we have one reflectance map for each point (pixel) of the face, we can repeat this process for every reflectance map to compute the appearance of the entire face under novel illumination given by the environment map. This is the relighting equation and is the core method of producing images under novel illumination.

6.2. Optimisations

To make relighting run faster, we formulate Equation 6.1 as a matrix multiplication problem, to which well-known optimisations can be applied. Suppose first, that our captured frames are of size $w_f \times h_f$ with s_f pixels in total, and our reflectance and environment maps are of size $w_m \times h_m$ with a total of s_m pixels. In our tests, described below, $w_f \times h_f = 150 \times 200$ and $w_m \times h_m = 64 \times 32$.

In short, relighting consists of multiplying each reflectance map with an environment map while summing individual results to obtain the output image. The convolution of reflectance and environment maps can be simplified to a dot product between two vectors of size s_m, where each contains all pixels of the respective reflectance or environment map, flattened to a one-dimensional array. The order of the pixels within the array is not relevant, but needs to be consistent over all reflectance and environment maps. Stacking vectors of each reflectance map together, we formulate a matrix A of size $s_f \times s_m$, where each row represents a single, flattened reflectance map. The environment map is made into a single-column matrix B of size $s_m \times 1$. Multiplying A and B yields a matrix of size $s_f \times 1$, where every row represents a relit pixel of the output image, arranged in the same order as reflectance maps in A.

This formulation separates reflectance data from incoming illumination, which makes relighting for novel illuminations as simple as substituting B and recomputing $A \times B$. This is especially useful for making animations where the reflectance data does not change, but environment lighting does.

We perform a benchmark test for three different implementations of relighting. The first is a parallelised but naive CPU implementation where we loop through all pixels of each reflectance map and perform the computation. The second and third approaches use the matrix multiplication method on the CPU and the GPU, respectively. For optimised CPU-based matrix-multiplication implementation, we use OpenCV library [5]. The library also supports CUDA for GPU execution. All tests were averaged over 10 iterations, performed on a 8-core Intel i7-6700HQ 2.60GHz CPU and NVIDIA GeForce GTX 950M GPU with latest CUDA drivers. Results are summarised in Table 6.1. We observe that the GPU implementation runs in real-time and achieves a speed-up of about 32 times compared to the naive CPU implementation and 3 times compared to the optimised CPU implementation. Since we used a commodity GPU, we estimate that the latest GPU hardware would run even faster, and, due to having more memory, allow for higher resolution images to be used as well.

implementation	**runtime [ms]**	**runtime [FPS]**
naive CPU	1,486.80	0.87
matrix multiplication CPU	161.12	8.55
matrix multiplication GPU	33.57	27.82

Table 6.1: Benchmark test for different implementations of relighting.

Nevertheless, we expected the GPU implementation to perform much faster than it did Compared to the optimised CPU implementation. While we assumed the bottleneck lies in the transfer of memory between the CPU and the GPU, we performed an additional test to validate our hypothesis. We measured the time of 100 iterations of multiplying two 32-bit floating-point matrices of size 1000×1000 in three different scenarios. In the first, the operation is performed on the CPU. In the second, the operation is performed on the GPU, but the memory of one matrix is uploaded to the GPU before each iteration and downloaded to machine memory (i.e. RAM) after each operation. The third operation is also performed on the GPU, but memory is reused over all iterations, such that no memory transfer is present.

The results, seen in Table 6.2, show that memory transfer causes a slow-down by three times. This means that reflectance maps should be cached in the GPU for maximum performance, so that only environment maps and relit images need to be transferred. It should be noted, however, that performance increments in the two tests are not comparable because of difference matrix sizes and because the first test required computation for three separate channels. We also observed that when we perform any GPU computations for the first time, there is a several-second delay, presumably due to initialisation of GPU kernels. Thus, the first iteration was ignored in the above tests.

implementation	**runtime [ms]**
CPU	130,754
GPU with continuous memory upload	3,162
GPU with cached memory	1,001

Table 6.2: Memory transfer bottleneck benchmark results.

6.3. Dynamic Relighting using Reenactments

Utilising the GPU, we are able to relight static images with any environment lighting in real-time. To extend applications and allow dynamic relighting, with animated facial movements, we resort to using a reenactment approach [47]. Reenactments take a driving video of one's face and transfer head and facial movements to a target image of another face, such that the appearance of the target face does not change, but its movement mimics the face in the driving video. This machine-learning approach can not only transfer head movement and rotation in 3D space, but also details such as facial deformations due to different expressions or closing and opening eyes and mouth. With this method, we are able to perform relighting on a static image, and then add artificial facial movement on top. The source of this artificial movement is a recorded video, or even a live camera feed for more capable computers.

Because reenactments are a post-processing image-based technique, head rotation does not change the lighting on the head; instead, it warps it, as if the lighting was drawn onto the texture of the face. In other words, lighting for a relit image is computed for a static pose, therefore, it no longer matches the expected lighting after the head is rotated using reenactments. This mismatch is fairly noticeable and undermines the quality of relighting.

To solve this issue, we apply an inverse rotation to the light source when performing relighting, such that after head rotation with reenactments, the lighting is oriented as we would expect it to be. To achieve this, we need an estimation of head rotation before it is actually applied using reenactments. As this rotation comes from a driving video, we run a learning-based gaze estimation technique [23, 24] to obtain head rotation around X, Y, and Z axes. This rotation is inverted and applied to the environment map used to perform relighting. Afterwards, reenactments are applied on the relit image using the same driving video, such that the estimated gaze and transformation of the head by reenactments match. This makes lighting on the head consistent with its 3D rotation, as seen in Figure 6.1.

Figure 6.1: Reenactment results for relit images with gaze estimation. Driving source images are on the left and the following columns represents results for different environment maps. Notice how specular highlights in the third and shadows in the fourth column change as the head is rotated. The first row also exhibits reenactment artefacts above the ear.

Rotation of an environment map in 3D space is not as simple as a regular 2D rotation, thus, we detail the technical steps to perform it. Such a 3D rotation of an environment map is analogous to rotating the sphere that the environment map maps onto with spherical coordinates, and projecting the rotated texture back to a new environment map. While rotation around the Y axis corresponds with a simple translation along the x axis in image-space of environment maps, rotations around X and Z axes are more complex. To perform this, we create a new environment map $\hat{I}$, where each pixel p is sampled from the original map I. The sampling is done such that $\hat{I}[p] = I[f(p)]$, where $f(p)$ represents a mapping that accounts for the desired rotation. $f(p)$ is computed by converting image-space coordinates of p to a normalised 3D vector in XYZ space, and then utilising quaternions[1] to rotate the vector. Then, the resulting vector is converted back to image-space coordinates. This sampling is done for the center point of each pixel of $\hat{I}$, which is sufficient, however, additional samples per pixel can be used to improve quality even further. To produce smooth results, we did not sample the nearest pixel in I. Instead, we interpolated contributions of the four pixels neighbouring the sampled point, which is known as bilinear interpolation.

[1] For details about quaternions, the reader is referred to supplementary material [44].

The described reenactments approach can, on capable personal computers, add dynamic movements to relit images at interactive rates. Unfortunately, the change in the shape of the face due to facial deformations cannot be accounted for by simply rotating the environment map. Instead, future work should investigate using per-frame estimated 3D models for approximating these details. Additionally, the reenactments approach has trouble estimating the appearance of the face for greater rotation angles where parts of the face were not visible in static images. Thus, future work should also research application of the latest state-of-the-art reenactment approaches [53] to relighting.

7 Results

We have introduced a relighting technique for relighting static images, extended with a reenactment approach for dynamic footage. In this chapter, we look at results of this method. We first inspect relighting results for captured faces and as well as a synthetic case. The latter allows us to validate our relighting results against ground truth. Next, we compare our approach to a learning-based relighting method [58]. Finally, we summarise the suitability of dynamic relighting using reenactments.

Relighting of Captured Faces We have captured several faces with the setup illustrated in Figure 4.5, multiple in a studio and two that were captured independently, at home. Figure 7.1 illustrates results for relighting with different environment maps. Upon visual inspection, our conclusions are as follows.

- Our method achieves believable results for relighting faces and can also capture hair and facial hair. Relighting may, however, fail for strong light sources coming from aside or from behind the face as these areas cannot be sampled.
- While most cases yield smooth results, some seams can be visible. These are artefacts caused by inaccurate 3D models and a trade-off between a sharp or blurred visibility maps.
- Our method performs well for people with different skin tones.

Variations in Sampled Light Directions As the flashing light source during acquisition is moved manually, we cannot guarantee uniform sampling. Nevertheless, our method is robust to different sampling patterns and densities. Figure 7.2 shows sampled light directions for three different acquisitions.

Comparison with Ground Truth To validate the accuracy of our interpolation method, we use a synthetic face model as the ground truth. More precisely, we ray-traced a 3D face model with support for subsurface scattering and global illumination using Blender [8]. For constructing ground truth reflectance maps, we rendered 2048 frames, one for each pixel of the reflectance maps. To imitate real-world acquisition, we sampled a subset of several rendered frames, selected based on samples recorded in one such acquisition. For the same reason, we used the approximate 3D model instead of the ground truth that was used for rendering. Comparison of ground truth and interpolated reflectance maps can be seen in Figure 7.3, and the relighting results are portrayed in Figure 7.4. While the relit faces appear believable, we notice two major artefacts. First, as sampled region only spans over a part of the frontal hemisphere, we cannot capture lighting effects from aside or from behind the face. Second, our 3D model is approximate and, consequently, visibility maps are inaccurate. This results in smoothed shadows cast by facial features, primarily the nose, which are sometimes shaped incorrectly.

To quantify the differences between interpolated reflectance maps and the ground truth, we create error heat maps. We introduce an error measure of two reflectance maps, computed by averaging squared distances between colours of all respective pixels in RGB space. As the contribution of pixels

in reflectance maps fades toward the poles, we computed a weighted average instead, with the sine of a pixels' latitudinal coordinate as the weight, similar to how this was done in Section 5.5. We formalize the equation as

$$\text{error}(r, \hat{r}) = \frac{\sum_{l \in \mathcal{L}} \sin(\text{lat}(l))[(r_l^R - \hat{r}_l^R)^2 + (r_l^G - \hat{r}_l^G)^2 + (r_l^B - \hat{r}_l^B)^2]}{\sum_{l \in \mathcal{L}} \sin(\text{lat}(l))}$$

where r and $\hat{r}$ are the two reflectance maps, $\mathcal{L}$ the set of all pixels in reflectance maps, $\text{lat}(l)$ the latitudinal coordinate of pixel l, and $r_l^{\cdot}$ one of RGB values of the colour of pixel l in r. A heat map is obtained by computing this error for every pixel of the face with an interpolated and ground truth reflectance maps.

We first validate our interpolation without extra points that would extend the sampled region as described in Section 5.6. Figure 7.5a shows the associated heat map where we clearly see that most errors appear at the side of the face. This is expected as our sampling only covers a part of the frontal hemisphere, whereas the points are capable of reflecting light from aside and from the back hemisphere as well.

Next, we limited our error computation within reflectance maps only to pixels that are contained in the convex hull of the sampled points. This approach allows us to focus not on which areas we have or have not sampled, but on which sampled areas are interpolated incorrectly. Results are visualised in Figure 7.5b. We observe three regions where most errors occur. One is on the forehead, where interpolation is suboptimal as normals and visibility maps are no longer available due to the shortcomings of the 3D model, which does not span over the whole face. Second is on dorsal of the nose, which exhibits notable subsurface scattering effects for non-frontal lighting. While our method distinguished between lit and shadowed areas, it was not designed to replicate heavy subsurface scattering effects in the shadowed areas. Finally, we notice bright seams on both sides of the nose, caused by the inaccurate 3D model, which, in turn, causes shadows cast by the nose to be incorrectly estimated. This complies with the errors observed previously in Figure 7.4.

Finally, we present another case aimed at understanding how artificially extending the sampled region by adding extra synthetic samples helps capture the light from beyond the sampled region. To visualise this, we created heat maps by comparing reflectance maps interpolated with and without extra points to the ground truth. The results are captured in Figure 7.5c. We observed that errors for interpolation with extra points are slightly lower and, thus, this method is somewhat effective. However, as the intensity of the extended sampled region fades out toward the edges, we still fall short of capturing light from aside and from behind the face in its full intensity. This explains why sides of the face still exhibit significant error.

Comparison with Other Methods We also compare our results with those from a learning-based method [58]. Since their work uses a Spherical Harmonics illumination model, we have converted environment maps to this representation. To have comparable results we then converted Spherical Harmonics representation back to an environment maps to account for the loss of high-frequencies. Figure 7.6 shows some comparisons between the two methods. Even though their method can produce stronger directional light, our method produces more natural relighting results with much less artefacts.

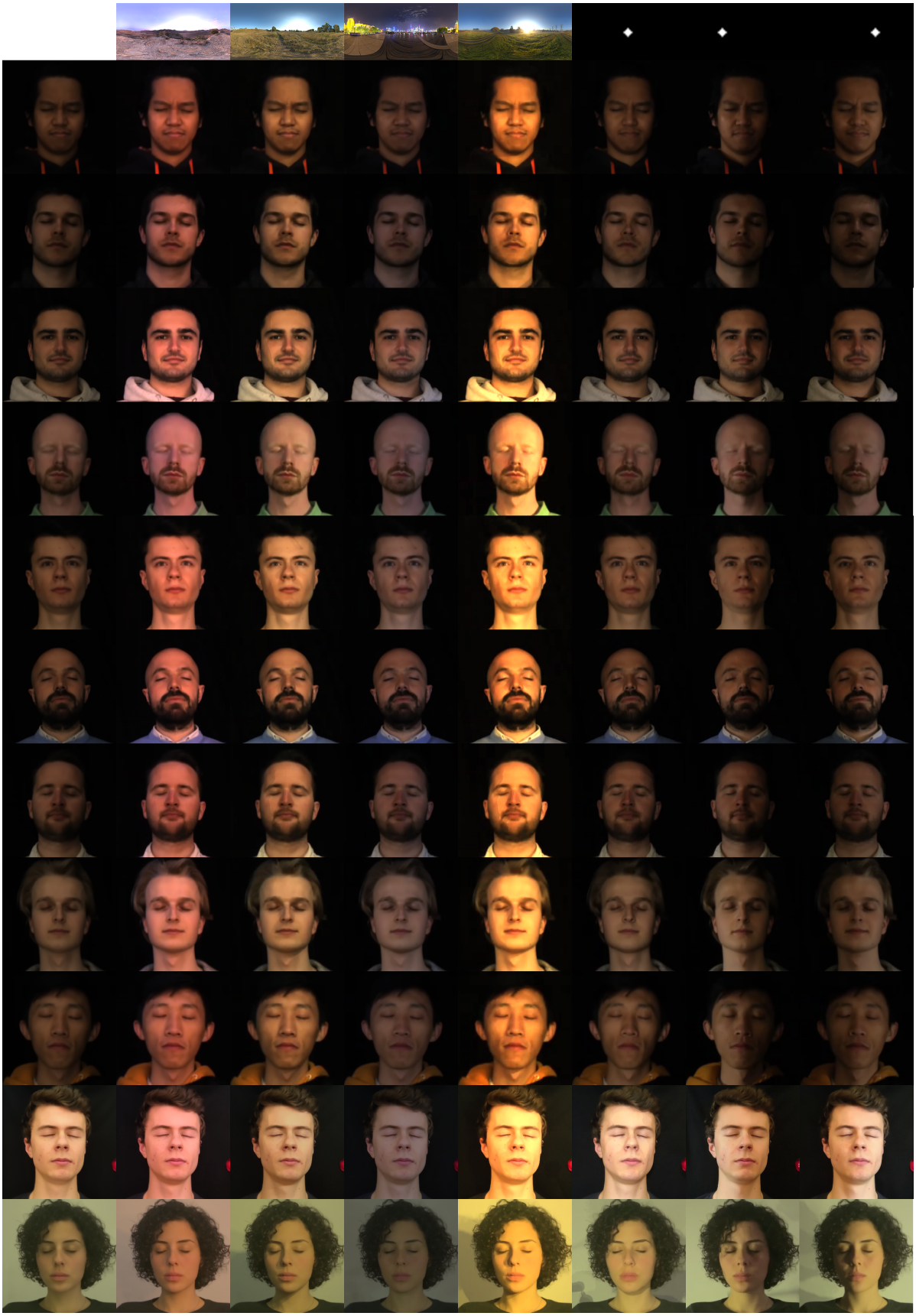


Figure 7.1: Left column shows a capture bright frame. The other columns show relighting results with the respective environment maps on top. The last two rows show independent acquisitions.

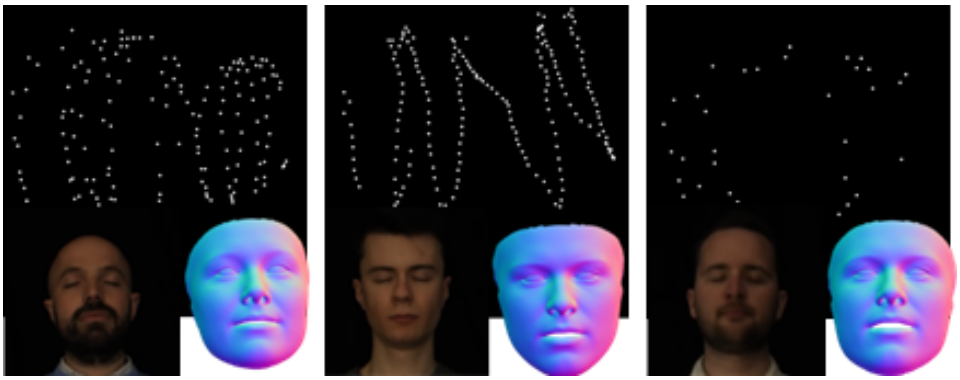


Figure 7.2: Light-direction sampling for three different acquisitions, including a frame and extracted normal map.

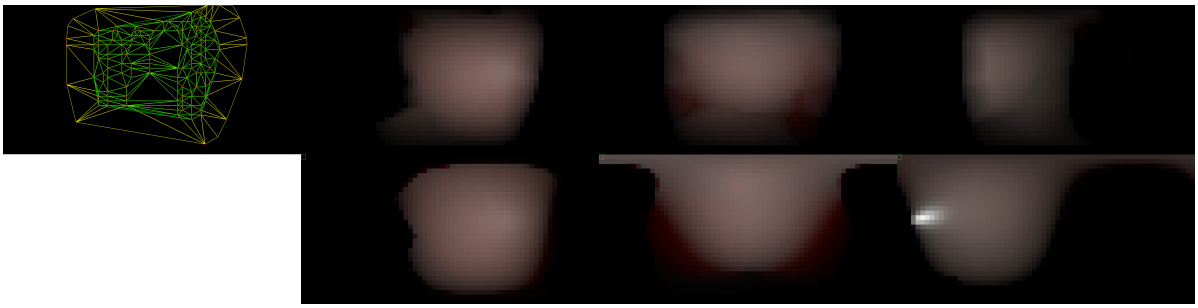
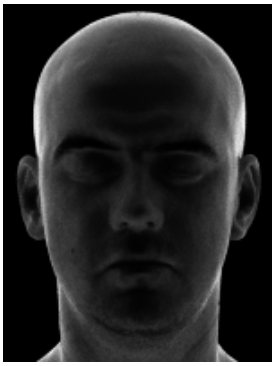
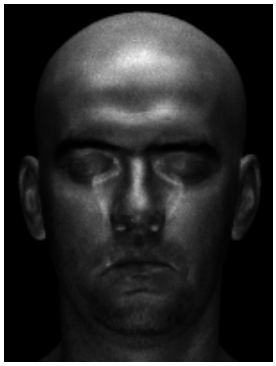


Figure 7.3: Comparison of interpolated reflectance maps versus the ground truth. The first row represents interpolated reflectance maps, the bottom row represents the ground truth. The columns represent (left to right): triangulation (green) with extra points (yellow) and reflectance maps for a point on the left (viewer's right) cheek, nose, right (viewer's left) temple. We cannot capture the specular highlight on the temple as it is outside of the sampled region.

Figure 7.4: Comparison of interpolated relighting results versus the ground truth. Each column represents relighting under a different environment map. The first row contains environment maps used, the middle row depicts interpolated relighting results, and the last row represents the ground truth. Notice how the top of the head is darker in interpolated results, especially in the leftmost column, as we cannot capture light that is outside of the sampled region. For the same reason, we fail to reproduce subsurface scattering on ears and specular highlight on the side as can easily be observed in the third column. Additionally, observe that shadow cast by the nose are sometimes soft or inaccurate, especially in the second and fourth column.



(a) Error heat map for sampled points, interpolated without extra points, computed over full area of reflectance maps.



(b) Error heat map for sampled points, interpolated without extra points, computed only for the sampled area of reflectance maps.

(c) Error heat maps for sampled points, interpolated with (right) and without (left) extra points, computed over full area of reflectance maps. To allow comparison, normalisation was computed jointly for both heat maps and not individually.

Figure 7.5: Heat maps measure error of interpolated reflectance map versus the ground truth. White areas represent greater error, black areas represent no error. For the purpose of visualisation, heat maps were normalised to range $[0, 1]$. Figures 7.5a, 7.5b, and 7.5c were normalised separately and therefore cannot be compared to one another directly.

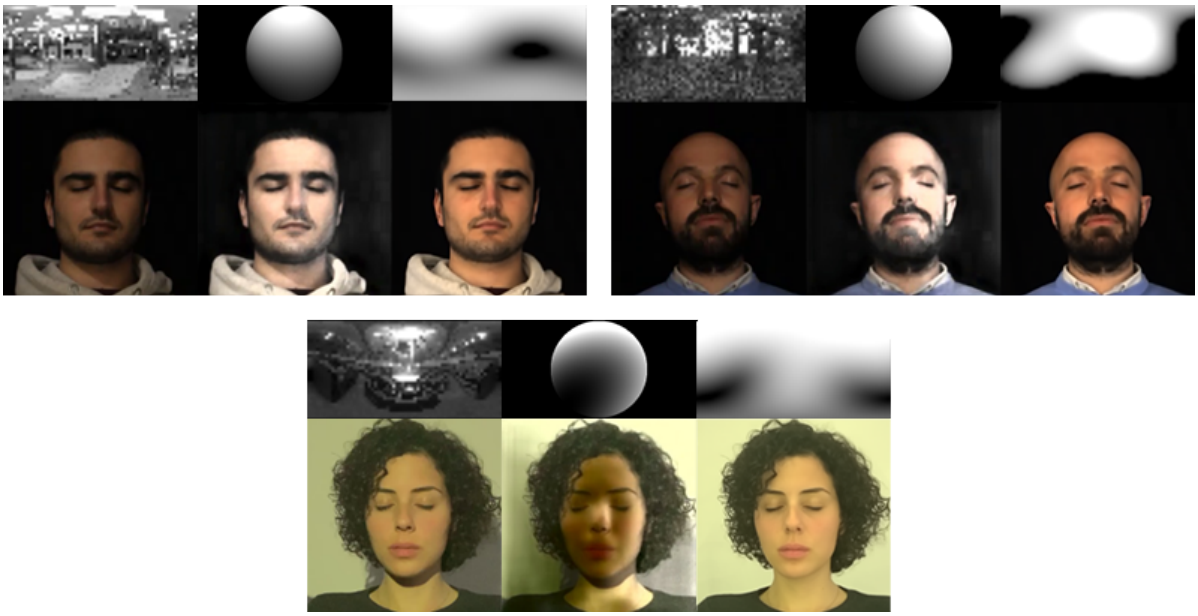


Figure 7.6: Odd rows (from left to right): Original environment map, its SH representation, SH representation reconverted into an environment map. Even rows (from left to right): Input frame from our capture session, frame relit using Zhou et al. [58] using the SH environmental light representation, and the subject relit with our method using the reconverted environment map.

8

Limitations

The results and comparisons described in the previous chapter demonstrate suitability of our relighting method. However, the method still suffers from several limitations that we list hereunder.

Acquisition Setup Without synchronization between the flashing light source and the camera in our acquisition setup, we need to consider a margin to ensure at least one dark frame without residual flash light. This imposes a limit on the flash frequency and, thus, acquisition time. Nevertheless, the current acquisition time of approximately one minute was not considered an issue by the subjects. A faster flash frequency and higher camera framerate could be used but this goes against our goal of acquisition simplicity. Our acquisition also includes a reflectance sphere, which is not something that is found in every home. However, we successfully tested our system with cheap stainless steel garden decoration spheres and Christmas balls that can easily be purchased.

Skin Reflectance While our captured frames record all aspects of a reflectance function, including indirect lighting due to subsurface scattering, we cannot accurately interpolate individual components. Light adjustments take foreshortening into account, but currently use a simplified model which does not differentiate between diffuse, specular, and indirect lighting conditions. Additionally, interpolation using visibility maps may reproduce sharp shadows with a good 3D model, but does not account for subsurface scattering that is present around the borders of the shadows. Moreover, certain parts of the face with much different reflections functions, such as eyes, if open, may require different interpolation methods [10].

Sampling Our setup make acquiring images from the back hemisphere challenging. While the actual back of the head is usually not interesting for relighting, we miss scattering effects on the ears and the side of the face. To compensate, we approximate these by artificially extending the sampled region by adding additional synthetic points. While this allows to reproduce some additional light from aside, it is not physically-based and fails to capture the reflectance function correctly.

Unstable Lighting Conditions Since we need some amount of ambient light to use dark frames for tracking, we cannot reproduce strong directional lighting. On the other hand, requiring a very dark environment for acquisition also makes its usage less attractive for general users. Again, given the trade-off, we opt for the easiest acquisition scenario.

In an attempt to mitigate these effects, we resorted to subtracting dark frames from bright ones. However, the method is not currently sufficient to counter limited hardware capability and perform the subtraction successfully.

Moreover, as the flashing light source is moved around manually, we can expect the distance between the light source and the face to slightly fluctuate. This may cause lighting on the face to vary in intensity due to light fall-off. Our method does not currently address this issue. A possible solution would be to include a matte object in the scene and adjust the brightness of capture frames according to the perceived brightness of the object. Another solution would be to use two reflective spheres,

positioned apart from one another, which would allow to compute not only direction but also distance to the light source due to disparity in highlights' locations. This remains an endeavour for future work.

3D Model Interpolation with visibility maps and light adaptation rely on the 3D model. The current method is still very approximate and does not reconstruct the whole face. As can be noted from the result, this causes seams to be visibility in some relit images. Nevertheless, we were able to produce convincing relighting results with our extremely simplified acquisition method.

Local Lighting Our relighting method can simulate any global illumination as given by environment maps, but falls short of estimating local lighting such as a light beam shone on one part of the face. While we could approximate this using different environment maps for the highlighted part of the face, this would not capture indirection illumination reflected from the highlighted region. For example, a light beam shone onto a cheek would cause the light to bounce to the side of the nose, but the method could not capture this. While there do exist image-based methods for approximating some of the global illumination effects such as subsurface scattering, they require a more advanced acquisition setup [51], which is contradictory to our goal of simplicity.

Hardware Limitations Although our relighting method runs in real-time on commodity GPU devices, it is still bound by hardware capabilities, especially in terms of GPU memory. Because we have one reflectance map for each pixel of the face, the amount of stored data raises quickly with increasing sizes of environment maps and captured frame. Even a simple crop to the sampled area of reflectance maps would increase capacity significantly, but this may not be sufficient for high-quality captures. Thus, the method would benefit from further compression or upsampling of relit image.

9

Conclusions

We have presented an end-to-end method for relighting faces under novel illumination. Our contributions are threefold. First, we present a novel acquisition method with a single camera, a flashing light source, and a reflective sphere. The setup imitates and offers benefits of Light Stages, yet, due to its simplicity, can be performed using accessible tools. This allows to capture face reflectance information at home, offering a trade-off between inaccessible professional equipment and learning-based approaches that operate with lower quality but no additional equipment.

Second, we introduce a novel technique for reflectance maps, interpolated via sparse samples. While existing techniques provide interpolation methods suitable for objects [14], our method focuses on interpolating samples for faces by considering shadow positions, light adaptation due to foreshortening, and sphericity. While the method could benefit from a better 3D model, it does not depend on any particular algorithm. Thus, when better alternatives become available, they can easily be used.

Our final contribution is a proof-of-concept for real-time dynamic relighting using reenactment approaches. This shows potential applications for communication and conferencing platforms as well as mobile applications that rely on capturing facial footage and compositing it on top of virtual backgrounds or even placing it in entire virtual environments. Our approach could help these applications match lighting in the recordings with that of the virtual counterpart.

Future research could investigate possibilities of extracting a 3D face model only from the captured images. While methods that require few samples of faces with varying illumination exist [18], the results exhibit artefacts and could benefit from multiple samples, known light direction, and the difference between direction and indirect illumination.

Another potential improvement is to find a robust calibration method to further mitigate camera imperfections. With this, we differentiate between direct and indirect lighting captured in bright and dark frames, allowing us to subtract the two to gain more contrasting images and improve light adaptation, which would, in turn, improve overall the quality of relighting. One could also research if this differentiation could allow capturing of subsurface scattering effects around self-cast shadows, which could then be used to approximate subsurface scattering effects on interpolated samples.

It would also be interesting to investigate further simplifications of the acquisition technique. For example, with additional improvements of the stabilisation method, one could ask a subject to hold their smartphone in a fixed position in front of them while turning around and recording changes in illumination on their face. Combined with the estimated light direction obtained via the smartphone's gyroscope, this could provide useful information for relighting with an even shorter and simpler acquisition.

Bibliography

[1] O. Alexander, M. Rogers, W. Lambeth, J. Chiang, W. Ma, C. Wang, and P. Debevec. The digital emily project: Achieving a photorealistic digital actor. *IEEE Computer Graphics and Applications*, 30(4):20–31, July 2010. ISSN 1558-1756. doi: 10.1109/MCG.2010.65.

[2] S. Alotaibi and W. A. P. Smith. A biophysical 3d morphable model of face appearance. In *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 824–832, Oct 2017. doi: 10.1109/ICCVW.2017.102.

[3] Thabo Beeler, Bernd Bickel, Paul Beardsley, Bob Sumner, and Markus Gross. High-quality single-shot capture of facial geometry. In *ACM SIGGRAPH 2010 Papers*, SIGGRAPH '10, New York, NY, USA, 2010. Association for Computing Machinery. ISBN 9781450302104. doi: 10.1145/1833349.1778777. URL https://doi.org/10.1145/1833349.1778777.

[4] Jean-Yves Bouguet et al. Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm. *Intel corporation*, 5(1-10):4, 2001.

[5] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

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

[7] Brian Cabral, Marc Olano, and Philip Nemec. Reflection space image based rendering. *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1999*, pages 165–171, 1999. doi: 10.1145/311535.311553.

[8] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. URL http://www.blender.org.

[9] Paul Debevec. The Light Stages and Their Applications to Photoreal Digital Actors. In *SIGGRAPH Asia*, Singapore, November 2012.

[10] Paul Debevec, Tim Hawkins, Chris Tchou, Haarm-Pieter Duiker, Westley Sarokin, and Mark Sagar. Acquiring the reflectance field of a human face. *Proceedings of the ACM SIGGRAPH Conference on Computer Graphics*, pages 145–156, 2000. doi: 10.1145/344779.344855.

[11] Per Einarsson, Charles-Felix Chabert, Andrew Jones, Wan-Chun Ma, Bruce Lamond, Tim Hawkins, Mark Bolas, Sebastian Sylwan, and Paul Debevec. Relighting human locomotion with flowed reflectance fields. In *Proceedings of the 17th Eurographics Conference on Rendering Techniques*, EGSR '06, page 183–194, Goslar, DEU, 2006. Eurographics Association. ISBN 3905673355.

[12] Gunnar Farnebäck. Two-frame motion estimation based on polynomial expansion. In *Scandinavian conference on Image analysis*, pages 363–370. Springer, 2003.

[13] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24 (6):381–395, 1981.

[14] Martin Fuchs, Hendrik P. A. Lensch, Volker Blanz, and Hans-Peter Seidel. Superresolution reflectance fields: Synthesizing images for intermediate light directions. *Computer Graphics Forum*, 26(3):447–456, 2007. doi: https://doi.org/10.1111/j.1467-8659.2007.01067.x. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-8659.2007.01067.x.

[15] G. Fyffe, P. Graham, B. Tunwattanapong, A. Ghosh, and P. Debevec. Near-instant capture of high-resolution facial geometry and reflectance. *Computer Graphics Forum*, 35(2):353–363, 2016. ISSN 14678659. doi: 10.1111/cgf.12837.

[16] Graham Fyffe and Paul Debevec. Single-Shot Reflectance Measurement from Polarized Color Gradient Illumination. *2015 IEEE International Conference on Computational Photography, ICCP 2015 - Proceedings*, 2015. doi: 10.1109/ICCPHOT.2015.7168375.

[17] Gaurav Garg, Eino-Ville Talvala, Marc Levoy, and Hendrik P Lensch. Symmetric Photography: Exploiting Data-sparseness in Reflectance Fields. *Eurographics Symposium on Rendering (EGSR)*, pages 251–262, 2006. ISSN 17273463.

[18] Athinodoros S. Georghiades, Peter N. Belhumeur, and David J. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE transactions on pattern analysis and machine intelligence*, 23(6):643–660, 2001.

[19] Abhijeet Ghosh, Tim Hawkins, Pieter Peers, Sune Frederiksen, and Paul Debevec. Practical modeling and acquisition of layered facial reflectance. In *ACM SIGGRAPH Asia 2008 Papers*, SIGGRAPH Asia '08, pages 139:1–139:10, New York, NY, USA, 2008. ACM. ISBN 978-1-4503-1831-0. doi: 10.1145/1457515.1409092. URL http://doi.acm.org/10.1145/1457515.1409092.

[20] Abhijeet Ghosh, Graham Fyffe, Borom Tunwattanapong, Jay Busch, Xueming Yu, and Paul Debevec. Multiview face capture using polarized spherical gradient illumination. *ACM Transactions on Graphics*, 30(6), 2011. ISSN 07300301. doi: 10.1145/2024156.2024163.

[21] Paulo Gotardo, Jérémy Riviere, Derek Bradley, Abhijeet Ghosh, and Thabo Beeler. Practical dynamic facial appearance modeling and acquisition. *SIGGRAPH Asia 2018 Technical Papers, SIGGRAPH Asia 2018*, 37(6), 2018. ISSN 15577368. doi: 10.1145/3272127.3275073.

[22] Paul Graham, Borom Tunwattanapong, Jay Busch, Xueming Yu, Andrew Jones, Paul Debevec, and Abhijeet Ghosh. Measurement-based synthesis of facial microgeometry. In *ACM SIGGRAPH 2012 Talks*, SIGGRAPH '12, pages 9:1–9:1, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1683-5. doi: 10.1145/2343045.2343057. URL http://doi.acm.org/10.1145/2343045.2343057.

[23] Jianzhu Guo, Xiangyu Zhu, and Zhen Lei. 3ddfa. https://github.com/cleardusk/3DDFA, 2018.

[24] Jianzhu Guo, Xiangyu Zhu, Yang Yang, Fan Yang, Zhen Lei, and Stan Z Li. Towards fast, accurate and stable 3d dense face alignment. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[25] Nejat Guzelsu, John F. Federici, Hee C. Lim, Hans R. Chauhdry, Arthur Ritter, and Tom Findley. Measurement of skin stretch via light reflection. *Journal of Biomedical Optics*, 8(1):80 – 86, 2003. doi: 10.1117/1.1527936. URL https://doi.org/10.1117/1.1527936.

[26] Tim Hawkins, Jonathan Cohen, and Paul Debevec. A photometric approach to digitizing cultural artifacts. In *Proceedings of the 2001 Conference on Virtual Reality, Archeology, and Cultural Heritage*, VAST '01, pages 333–342, New York, NY, USA, 2001. ACM. ISBN 1-58113-447-9. doi: 10.1145/584993.585053. URL http://doi.acm.org/10.1145/584993.585053.

[27] Tim Hawkins, Andreas Wenger, Chris Tchou, Andrew Gardner, Fredrik Göransson, and Paul Debevec. Animatable facial reflectance fields. In *Proceedings of the Fifteenth Eurographics Conference on Rendering Techniques*, EGSR'04, pages 309–319, Aire-la-Ville, Switzerland, Switzerland, 2004. Eurographics Association. ISBN 3-905673-12-6. doi: 10.2312/EGWR/EGSR04/309-319. URL http://dx.doi.org/10.2312/EGWR/EGSR04/309-319.

[28] Henrik Wann Jensen, Stephen R Marschner, Marc Levoy, and Pat Hanrahan. A practical model for subsurface light transport. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 511–518, 2001.

[29] Jorge Jimenez, Timothy Scully, Nuno Barbosa, Craig Donner, Xenxo Alvarez, Teresa Vieira, Paul Matts, Verónica Orvalho, Diego Gutierrez, and Tim Weyrich. A practical appearance model for dynamic facial color. In *ACM SIGGRAPH Asia 2010 Papers*, SIGGRAPH ASIA '10, pages 141:1–141:10, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0439-9. doi: 10.1145/1866158.1866167. URL http://doi.acm.org/10.1145/1866158.1866167.

[30] Vahid Kazemi and Josephine Sullivan. One millisecond face alignment with an ensemble of regression trees. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[31] Chloe Legendre, Wan Chun Ma, Rohit Pandey, Sean Fanello, Christoph Rhemann, Jason Dourgarian, Jay Busch, and Paul Debevec. Learning Illumination from Diverse Portraits. *SIGGRAPH Asia 2020 Technical Communications, SA 2020*, 2020. doi: 10.1145/3410700.3425432.

[32] Wc Ma, Tim Hawkins, and P Peers. Rapid acquisition of specular and diffuse normal maps from polarized spherical gradient illumination. *Proceedings of the 18th Eurographics conference on Rendering Techniques*, pages 183–194, 2007. URL http://dl.acm.org/citation.cfm?id=2383873.

[33] Tom Malzbender, Dan Gelb, and Hans Wolters. Polynomial texture maps. *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 2001*, pages 519–528, 2001. doi: 10.1145/383259.383320.

[34] Abhimitra Meka, Christian Häne, Rohit Pandey, Michael Zollhöfer, Sean Fanello, Graham Fyffe, Adarsh Kowdle, Xueming Yu, Jay Busch, Jason Dourgarian, Peter Denny, Sofien Bouaziz, Peter Lincoln, Matt Whalen, Geoff Harvey, Jonathan Taylor, Shahram Izadi, Andrea Tagliasacchi, Paul Debevec, Christian Theobalt, Julien Valentin, and Christoph Rhemann. Deep reflectance fields: High-quality facial reflectance field inference from color gradient illumination. *ACM Trans. Graph.*, 38(4), July 2019. ISSN 0730-0301. doi: 10.1145/3306346.3323027. URL https://doi.org/10.1145/3306346.3323027.

[35] Mark Mudge, Jean-Pierre Voutaz, Carla Schroer, and Marlin Lum. Reflection transformation imaging and virtual representations of coins from the hospice of the grand st. bernard. In *VAST*, volume 6, pages 29–40, 2005.

[36] Mark Mudge, Tom Malzbender, Carla Schroer, and Marlin Lum. New reflection transformation imaging methods for rock art and multiple-viewpoint display. In *Ioannides, M.; Arnold, D.; Niccolucci, F. & Mania, K., eds., The 7th International Symposium on Virtual Reality, Archaeology and Cultural Heritage*, volume 6, pages 195–202. Vast, 2006.

[37] Koki Nagano, Graham Fyffe, Oleg Alexander, Jernej Barbiç, Hao Li, Abhijeet Ghosh, and Paul Debevec. Skin microstructure deformation with displacement map convolution. *ACM Trans. Graph.*, 34(4):109:1–109:10, July 2015. ISSN 0730-0301. doi: 10.1145/2766894. URL http://doi.acm.org/10.1145/2766894.

[38] Thomas Nestmeyer, Jean-François Lalonde, Iain Matthews, and Andreas M Lehrmann. Learning physics-guided face relighting under directional light. In *Conference on Computer Vision and Pattern Recognition*, pages 5123–5132. IEEE/CVF, June 2020.

[39] Augusto L.P. Nunes, Anderson Maciel, Gary W. Meyer, Nigel W. John, Gladimir V.G. Baranoski, and Marcelo Walter. Appearance modelling of living human tissues. *Computer Graphics Forum*, 38(6):43–65, 2019. doi: 10.1111/cgf.13604. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.13604.

[40] Pieter Peers, Naoki Tamura, Wojciech Matusik, and Paul Debevec. Post-production facial performance relighting using reflectance transfer. *ACM Transactions on Graphics*, 26(99):52, 2007. ISSN 07300301. doi: 10.1145/1239451.1239503.

[41] Jérémy Riviere, Paulo Gotardo, Derek Bradley, Abhijeet Ghosh, and Thabo Beeler. Single-shot high-quality facial geometry and skin appearance capture. *ACM Trans. Graph.*, 39(4), July 2020. ISSN 0730-0301. doi: 10.1145/3386569.3392464. URL https://doi-org.tudelft.idm.oclc.org/10.1145/3386569.3392464.

[42] Brian Schulkin, Hee C. Lim, Nejat Guzelsu, Glen Jannuzzi, and John F. Federici. Polarized light reflection from strained sinusoidal surfaces. *Appl. Opt.*, 42(25):5198–5208, Sep 2003. doi: 10.1364/AO.42.005198. URL http://ao.osa.org/abstract.cfm?URI=ao-42-25-5198.

[43] Soumyadip Sengupta, Angjoo Kanazawa, Carlos D Castillo, and David W Jacobs. Sfsnet: Learning shape, reflectance and illuminance of facesin the wild'. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6296–6305, 2018.

[44] A. Shenitzer, B.A. Rosenfeld, and H. Grant. *A History of Non-Euclidean Geometry: Evolution of the Concept of a Geometric Space*. Studies in the History of Mathematics and Physical Sciences. Springer New York, 1988. ISBN 9780387964584.

[45] Jianbo Shi et al. Good features to track. In *1994 Proceedings of IEEE conference on computer vision and pattern recognition*, pages 593–600. IEEE, 1994.

[46] Zhixin Shu, Sunil Hadap, Eli Shechtman, Kalyan Sunkavalli, Sylvain Paris, and Dimitris Samaras. Portrait lighting transfer using a mass transport approach. *ACM Trans. Graph.*, 37(1), October 2017. ISSN 0730-0301. doi: 10.1145/3095816.

[47] Aliaksandr Siarohin, Stéphane Lathuilière, Sergey Tulyakov, Elisa Ricci, and Nicu Sebe. First order motion model for image animation. In *Conference on Neural Information Processing Systems (NeurIPS)*, December 2019.

[48] Olga Sorkine-Hornung and Michael Rabinovich. Least-squares rigid motion using svd. https://igl.ethz.ch/projects/ARAP/svd_rot.pdf, 1 2017.

[49] Tiancheng Sun, Jonathan T. Barron, Yun-Ta Tsai, Zexiang Xu, Xueming Yu, Graham Fyffe, Christoph Rhemann, Jay Busch, Paul Debevec, and Ravi Ramamoorthi. Single image portrait relighting. *ACM Trans. Graph.*, 38(4), July 2019. ISSN 0730-0301. doi: 10.1145/3306346.3323008. URL https://doi.org/10.1145/3306346.3323008.

[50] Tiancheng Sun, Zexiang Xu, Xiuming Zhang, Sean Fanello, Christoph Rhemann, Paul Debevec, Yun Ta Tsai, Jonathan T. Barron, and Ravi Ramamoorthi. Light Stage Super-Resolution: Continuous High-Frequency Relighting. *ACM Transactions on Graphics*, 39(6), 2020. ISSN 15577368. doi: 10.1145/3414685.3417821.

[51] Sarah Tariq, Andrew Gardner, Ignacio Llamas, Andrew Jones, Paul Debevec, and Greg Turk. Efficient estimation of spatially varying subsurface scattering parameters for relighting. pages 1–9, 01 2006.

[52] Paul Viola and Michael J Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004.

[53] Ting-Chun Wang, Arun Mallya, and Ming-Yu Liu. One-shot free-view neural talking-head synthesis for video conferencing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021.

[54] Zhibo Wang, Xin Yu, Ming Lu, Quan Wang, Chen Qian, and Feng Xu. Single image portrait relighting via explicit multiple reflectance channel modeling. *ACM Transactions on Graphics*, 39 (6), 2020. ISSN 15577368. doi: 10.1145/3414685.3417824.

[55] Tim Weyrich, Wojciech Matusik, Hanspeter Pfister, Bernd Bickel, Craig Donner, Chien Tu, Janet McAndless, Jinho Lee, Addy Ngan, Henrik Wann Jensen, and Markus Gross. Analysis of human faces using a measurement-based skin reflectance model. *ACM Trans. on Graphics (Proc. SIGGRAPH 2006)*, 25(3):1013–1024, 2006. doi: http://doi.acm.org/10.1145/1179352.1141987.

[56] Zexiang Xu, Kalyan Sunkavalli, Sunil Hadap, and Ravi Ramamoorthi. Deep image-based relighting from optimal sparse samples. *ACM Transactions on Graphics*, 37(4):1–13, 2018. ISSN 15577368. doi: 10.1145/3197517.3201313.

[57] Xiuming Zhang, Sean Fanello, Yun Ta Tsai, Tiancheng Sun, Tianfan Xue, Rohit Pandey, Sergio Orts-Escolano, Philip Davidson, Christoph Rhemann, Paul Debevec, Jonathan T. Barron, Ravi Ramamoorthi, and William T. Freeman. Neural Light Transport for Relighting and View Synthesis. *ACM Transactions on Graphics*, 40(1), 2021. ISSN 15577368. doi: 10.1145/3446328.

[58] Hao Zhou, Sunil Hadap, Kalyan Sunkavalli, and David Jacobs. Deep single-image portrait relighting. *Proceedings of the IEEE International Conference on Computer Vision*, 2019-Octob: 7193–7201, 2019. ISSN 15505499. doi: 10.1109/ICCV.2019.00729.