Abstract

This project presents a novel technique to render objects under any arbitrary illumination. The illumination conditions in a regular office room are used to capture the reflectance field of an object. It is then rendered under any illumination condition using an image-based relighting algorithm. The results are improved with an optimisation process that enhances the projection of an environment map on the sparse lighting basis defined by the office room. Comparisons with light stage 6 and a free-form light stage method are made. Finally, an easy to use software that gather the three types of relightings is presented.
Acknowledgements

I would like to thank my supervisor, Dr Abhijeet Ghosh, for his time, his quick answers to e-mails and his help. He has always been enthusiastic about this project and he kindly accepted to use his office room for the data captures. He also provided me with the camera for the capture.

I also would like to thank my family for their help and their support during this whole year at Imperial College London. I am very grateful to them for giving me the opportunity to study in London.
# Contents

1 Introduction 2
  1.1 Motivation and goals ........................................ 3
  1.2 Outline ..................................................... 3

2 Research background 4
  2.1 Light probes ............................................... 4
  2.2 High Dynamic Range (HDR) pictures .............................. 6
  2.3 Image-based relighting ....................................... 11
  2.4 The free-form light stage .................................... 15
  2.5 Optimisation of the projection on the lighting basis ....... 18

3 Theoretical background 20
  3.1 Coordinates systems ........................................ 20
  3.2 Solid angle .................................................. 21
  3.3 Matrix theory ............................................... 23
  3.4 Principal Component Analysis ................................. 26
  3.5 Gamma correction ............................................ 28

4 Image-based relighting with light stage 6 30
  4.1 Data used .................................................... 30
  4.2 Image-based relighting step by step .......................... 32
  4.3 Results ....................................................... 40
  4.4 Implementation details ..................................... 44

5 Office room relighting 45
  5.1 Challenges of using an office room as a light stage ........ 45
  5.2 Presentation of the camera .................................. 45
  5.3 Data capture ................................................. 49
  5.4 Data processing ............................................. 53
  5.5 Find the incoming light directions ............................. 56
  5.6 Use masks for each light source ............................... 63
  5.7 Optimisation of the projection on the lighting basis ....... 67

6 Results and comparisons 74
  6.1 The free-form light stage .................................... 74
  6.2 Comparison with the light stage relighting .................... 80

7 Conclusion and future work 84

A Program for image-based relighting 89
  A.1 Presentation ................................................ 89
  A.2 How to use the program ..................................... 90
  A.3 How to compile ? ........................................... 93
  A.4 How to install the program ? ................................ 94
## List of Figures

1.1 Picture of a caustic [26] ................................................. 2
2.1 Grace cathedral light probe [10] ........................................ 5
2.2 Mirror ball and angular map comparison [19] .......................... 5
2.3 Grace cathedral in latitude longitude map [10] .......................... 6
2.4 Picture of Stanford Memorial Church with a low exposure setting [12] . 7
2.5 Picture of Stanford Memorial Church with a high exposure setting [12] . 7
2.6 Radiance values of a high dynamic range picture [12] .................... 8
2.7 Linear mapping of the pixels to display a high dynamic range picture [12] . 9
2.8 Histogram compression of the pixels to display a high dynamic range picture [12] . 9
2.9 Sequence of pictures taken with a 1 f-stop exposure difference between them [12] . 10
2.10 Weighting function for high dynamic range pictures [12] .................. 10
2.11 Reflectance field and non-local reflectance field [11] ..................... 11
2.14 Free-form light stage setup [33] ....................................... 16
2.15 Diagram that explains how to derive the incoming light direction (Free-form light stage) [33] ................................................. 17
2.16 Voronoi diagrams for the free-form light stage [33] ...................... 17
2.17 Approximation of an environment map with spherical harmonics and local lights [43] . 18
2.18 Comparison of the optimised and non optimised results [43] ............... 19
2.19 Comparison of the optimisation results in original and in the PCA space [43] . 19
3.1 Cartesian coordinate system .............................................. 20
3.2 Spherical coordinate system .............................................. 21
3.3 Solid angle ....................................................................... 22
3.4 Jacobian in latitude longitude maps ..................................... 22
3.5 Principal components basis ............................................... 26
3.6 How to display linear intensity values on a screen ....................... 28
3.7 Gamma curve of 2.2 and its corresponding inverse ..................... 29
4.1 Light stage 6 ................................................................... 31
4.2 Reflectance field of the plant 1 .......................................... 31
4.3 Reflectance field of the plant 2 .......................................... 31
4.4 Incoming light direction for the reflectance field ..................... 32
4.5 Incoming light directions for the light stage ........................... 32
4.6 atan and atan2 functions .................................................. 33
4.7 Voronoi diagram for light stage 6 ........................................ 34
4.8 Light stage intensities ....................................................... 35
4.9 Weight of the Voronoi cells with point integration .................... 36
4.10 Weight of the Voronoi cells with Gaussian integration ............... 37
4.11 Color shifting example .................................................... 38
4.12 How to compute the background ........................................ 39
4.13 Mask for the helmet [10] .................................................. 40
Chapter 1

Introduction

From the first animated movie, Toy Story, created by Pixar Animations Studios [41] in 1995 to James Cameron’s Avatar in 2009, computer graphics have evolved at an astonishing pace. The character designs, the animations, the visual effects are only a few examples of the improvements. As both the cinema and the video games industries get bigger, more popular and more lucrative the graphic renderings also get better. Today studios are forced to invest in cutting-edge computers and technologies to produce the best possible renderings for future blockbusters.

In computer graphics, two types of rendering methods can be distinguished. The first one uses rendering algorithms to produce an image of a scene given its description. For instance, the description includes the geometry of the objects and the properties of each material in the scene. The most famous algorithms include the ray tracing [36] and the related photon mapping by the academy award winner Jensen [26]. The latter is particularly interesting as it relies on the physics theory of light in which light can be described as particles called photons. This algorithm models the behaviour of photons in order to produce realistic renderings. It achieves very good results for caustics (figure 1.1) as these corresponds to converging light rays that is to say to a high density of photons.

Figure 1.1: Picture of a Cognac glass. The caustic that appears on the surface is rendered using the photon mapping algorithm. Source [26]

While the first type of rendering method became popular with the rendering equation for com-
puter graphics by Kajiya [27] in 1986, the second type is more recent and uses pictures to produce realistic renderings. Image-based relighting belongs to the second category and is studied in this project.

1.1 Motivation and goals

The goal of this project is to render an object under any illumination condition using an image-based relighting algorithm with pictures taken in a regular office room.

Image-based relighting is a recent technique presented for the first time in 2000 at the international conference on graphics SIGGRAPH. This method uses a set of pictures of an object illuminated from different directions called the reflectance field. Usually the reflectance field is captured with a light stage that is to say with a dome of lights that creates many illumination conditions at a high frequency. A high speed camera can then capture images under these lighting conditions. This type of installation is often used for research or in studio environments to produce realistic renderings of very high quality. For instance, the light stage technology has been used on several movies including Spider-Man™ 2 and Avatar [9].

The main drawbacks of light stages include their price, their complexity to build and automate and their size. Indeed large rooms are required as the dome can have a diameter up to eight meters to capture the reflectance field of large objects such as a human body [40]. As a result light stages are often reserved to studios and research. In this project I use a different approach to capture the reflectance field. The idea is to use the lighting conditions available in a regular office room to do the capture. Every light source from the windows to a computer screen can be used to illuminate an object. This imposes many constraints starting with the room itself. For instance, white walls reflect light creating secondary light sources that have to be taken into account. Besides light coming through a window is uncontrolled as the illumination on a sunny day is brighter than the illumination on a cloudy day. This dependence on the outdoor environment at the moment of the capture is one of the challenges of this project. It will be interesting to see how this novel method renders different types of objects such as diffuse and specular objects.

1.2 Outline

In this thesis, I first explain the research and theoretical backgrounds required to understand the concepts of office room relighting. Then chapter 4 presents how image-based relighting is computed with a light stage. Chapter 5 describes the relighting using a regular office room, from the data capture to the possible optimisations that can improve the final result. Finally comparisons with known methods such as the free-form light stage are presented in chapter 6 in order to evaluate the quality of the results.
Chapter 2

Research background

Image-based relighting is a recent technique and remains an active area of research today. This chapter presents the evolution of the method as well as the main papers that deals with it. These papers were my main resource to understand and implement image-based relighting with a regular office room.

2.1 Light probes

Environment maps and light probes are very important for image-based relighting and are used for many purposes. One of them is to capture a whole sphere of directions in a given environment in order to relight the object in the environment. Debevec [8] used them for the first time in a paper presented at SIGGRAPH 1998 before the light stage technology was even invented.

Light probes are created by taking one or more pictures of a mirror ball. One picture is enough as the mirror ball reflects the entire environment when seen from a given direction. However, a part of the environment is hidden by the reflection of the camera in the mirror ball. Hence two pictures of the mirror ball taken from 90 degrees apart can be assembled to create a light probe. Indeed the area hidden by the camera in the first mirror ball appears in the second picture of the mirror ball. A very famous example is the environment map of the Grace Cathedral (see figure 2.1)

Several parameterizations of environment maps are used in computer graphics. Three of them are presented here. A software such as HDRShop [15] implements panoramic transformations that can change the parameterisation of the environment map.

Mirror ball

A mirror ball simply corresponds to a picture of a mirror ball in a given environment without post processing. Figure 2.1 shows a picture of a mirror ball in the Grace cathedral environment.
Angular map

The mirror ball parameterisation does not use a linear scaling of angles and the pixels closer to the edges of the mirror ball are squashed. More specifically, close to the center of the ball an angle of 90 degrees represents a bigger area than on the edges of the ball [19] that is why the pixels on the edges seem to be squashed. This problem is solved with the angular map parameterisation in which every angle represent the same area of the ball. Figure 2.2 presents the two parameterisations.

Latitude longitude map

The latitude longitude map parameterisation shows the full sphere of directions on a plane similarly to a world map [19]. Figure 2.3 shows the latitude longitude map of the Grace cathedral along with
the directions. The middle corresponds to the back of the cathedral (+z) and the right and left edges to the front (−z). This map can easily be parameterised with spherical coordinates that is why it is the main parameterisation used in this project. For instance, the direct correspondence between a pixel on the latitude longitude map and its direction in spherical coordinates make the calculation of Voronoi diagram easier. Appendix D present the environment maps used in this project.

![Grace Cathedral environment map parameterised with latitude longitude coordinates.](image)

**Figure 2.3:** Grace Cathedral environment map parameterised with latitude longitude coordinates. The directions and the corresponding spherical coordinates are displayed. Source [10]

## 2.2 High Dynamic Range (HDR) pictures

Good light probes that fully capture an environment can only be created with high dynamic range pictures of a mirror ball. This section explains why.

### 2.2.1 Low Dynamic Range and High Dynamic Range pictures

Pictures taken with a camera are limited and cannot display as many details as our eyes can see. For instance, most of the time when a picture of a scene with both bright and dark areas is taken either the dark area is visible and the bright area is saturated or the bright area is visible but the dark area appears black (see figures 2.4 and 2.5). On the camera sensor this corresponds to pixels clamped to 0 (black pixels) or 255 (saturated pixels).

High-dynamic range pictures do not have this problem as they store the radiance\(^1\) of a pixel with floating point numbers [12]. Hence on these pictures the dark parts of an image do not have their value clamped to zero but have very low radiance value. Similarly bright areas are not saturated as their values are not clamped to 255. For instance a high dynamic range picture of the memorial has an average radiance of 120 in the right stained-glass window and 0.002 in the dark area at the left of the picture (figure 2.6). Low dynamic range versions at a middle exposure have an intensity of 0 and 255 for the same areas respectively.

\(^1\)The radiance corresponds to the power emitted by a light source per unit of surface and per unit of solid angle (unit \(\text{W/m}^2\text{sr}\)). [18]
2.2. HIGH DYNAMIC RANGE (HDR) PICTURES

The dynamic range of a picture is defined by the ratio of the brightest pixel of the image over the dimmest pixel of the image. Low dynamic range 8 bits images are limited to a dynamic range of $2^8 = 256$ whereas the dynamic range of HDR pictures can be up to millions for pictures of the sun [18].

2.2.2 Tone Mapping

High dynamic range pictures cannot be displayed with regular viewers as pixels correspond to floating point radiance values and not integers in the range $[0; 255]$. These values can be scaled linearly such as the biggest value corresponds to 255 and the lowest value to 0. However such a mapping of the radiance values is not efficient as it produces dark renderings (see figure 2.7). Therefore tone mapping algorithms are applied to high dynamic range pictures in order to save them as regular 8 bits pictures for display. A famous tone mapping algorithm is called histogram compression. Figure 2.8 shows the result of this algorithm on a picture of Stanford Memorial Church.

2.2.3 How to assemble a High Dynamic Range picture

The algorithm to assemble high dynamic range pictures from a sequence of low dynamic range (LDR) pictures is explained by Debevec et al. [12]. This sequence corresponds to several pictures of the same scene taken with different exposure settings. Typically the aperture is fixed on the camera and the shutter speed varies. In the end the sequence has to cover the full dynamic range of the scene. Qualitatively bright areas of the scene must appear in the picture with the lowest exposure setting whereas dark areas must be visible in the picture with the highest exposure setting. Figure 2.9 shows a sequence of pictures that can be assembled into a high dynamic range picture.
The formula to recover radiance values from the sequence of low dynamic range picture is given by equation 2.1. The pixels in the HDR image correspond to the exponential of this equation.

\[
\forall (i, j) \in [1; \text{height}] \times [1; \text{width}] \quad \ln(E_{i,j}) = \frac{\sum_{k=1}^{P} \omega(Z_{i,j,k})(g(Z_{i,j,k}) - \ln(\Delta t_k))}{\sum_{k=1}^{P} \omega(Z_{i,j,k})} \tag{2.1}
\]

with:

- \( P \) is the number of pictures in the sequence.
- \( E_{i,j} \) is the radiance of pixel \((i, j)\).
- \( Z_{i,j,k} \) is the value pixel \((i, j)\) of picture \(k\) in the sequence of low dynamic range pictures.
- \( \omega \) is a weighting function. This function is plotted on figure 2.10.
- \( g \) is the response curve of the camera.
- \( \Delta t_k \) is the exposure of picture \(k\) in the sequence of low dynamic range pictures.

Let \( Z_{\min} \) and \( Z_{\max} \) be the minimum and the maximum of the value of the pixels in the sequence of low dynamic range images. The weighting function (figure 2.10) equals 0 for \( Z_{\max} \) hence a saturated pixel whose value is clamped to \( Z_{\max} \) is ignored.
2.2. HIGH DYNAMIC RANGE (HDR) PICTURES

![Figure 2.7: High dynamic range picture of Stanford Memorial Church with a linear mapping of the pixels. Source [12]](image1)

![Figure 2.8: High dynamic range picture of Stanford Memorial Church with a histogram compression of the pixels. Source [12]](image2)

2.2.4 File formats

HDR pictures cannot be saved with usual picture formats such as JPEG or ppm. These formats store 8-bits unsigned char values [18]. Hence the dynamic range of the scene is limited to 256 as the pixels are integers between 0 and 255. HDR formats such as Portable FloatMap (.pfm), Greg Ward’s radiance format (.hdr) [45] and OpenEXR (.exr) [24] usually use 16 or 32 bits floating point numbers. With these formats the dynamic range of a picture is bigger as four bytes are used for each color channel instead of one byte. The drawback of HDR imaging is the size of the pictures. For instance, in this project I assembled a HDR image using three pictures of four megabytes each. The HDR result saved as a pfm file takes 300 megabytes. Hence programming with several HDR images can be difficult as a large amount of memory is needed to load pictures. HDR formats often require specific software to display them. I used HDRShop [15] that can display pfm, hdr and 16 bits TIFF formats.
CHAPTER 2. RESEARCH BACKGROUND

Figure 2.9: A sequence of pictures taken with a 1 f-stop exposure difference between them. Source [12]

Figure 2.10: Plot of the weighting function used to assemble high dynamic range images. Source: [12]
2.3 Image-based relighting

2.3.1 Reflectance field of an object

The light stage technology was introduced at SIGGRAPH 2000 by Debevec, Hawkins, Tchou, Duiker, Sarokin and Sagar [11]. This technology was created in order to capture the reflectance field of a human face but can be used for any other object. Given an incoming and an outgoing light direction, the reflectance field corresponds to the amount of light that is reflected by the object in the outgoing direction. The reflectance field is a 8 dimensional function as it also depends on the intersection point between the incoming (respectively outgoing) light ray and any closed surface surrounding the object. The data of a 8 dimensional function requires space to be stored and the calculations with such a function are expensive. As a result a non-local reflectance field is defined in the paper. If the light source producing the incoming light rays is far from the object, then the reflectance field does not depend on the intersection point between the incoming light ray and the object. Hence the reflectance field becomes a 6 dimensional function. Figure 2.11 shows the definitions of the local and non-local reflectance fields.

![Diagram](image.png)

Figure 2.11: (a) Definition of the reflectance field in the general case for all closed surface A surrounding the object (8 dimensions) : \( R(u_i, v_i, \theta_i, \phi_i, u_r, v_r, \theta_r, \phi_r) \). (b) Definition of the non-local reflectance field (6 dimensions) : \( R(\theta_i, \phi_i, u_r, v_r, \theta_r, \phi_r) \). Here the surface A is the human face. Source : [11]

2.3.2 Relighting with a light stage

The first light stage (figure 2.13) is made of a fixed camera and a light source that can take many determined positions around the object. These positions correspond to a uniform sampling of the sphere around the object. Each time the light source takes a position, a picture is taken. At the end of the capture, a set of pictures of the object with different illuminations is obtained. These will be used to create the non-local reflectance functions. During the capture the object has to remain fixed otherwise the reflectance field in incorrect. The article describes two purposes of capturing the reflectance field. The first one is image-based relighting that is to say how to render the object under any other illumination condition. The second one is rendering the human face from another viewpoint.

Once the capture is finished, the reflectance function is computed for each pixel of the object. The reflectance function for one specific pixel \((x, y)\) of the object corresponds to a latitude-longitude map in which the value for the direction \((\theta_i, \phi_i)\) is the value of pixel \((x, y)\) in the picture with the incoming light direction \((\theta_i, \phi_i)\). Hence at the end of this process, each pixel of the object has one reflectance function that corresponds to the value of that pixel under the different illumination conditions. Each function has a size of \(N_\theta \times N_\phi\) where \(N_\theta\) and \(N_\phi\) are the number of samples taken for the azimuthal angle \(\theta_i\) and for the polar angle \(\phi_i\). Figure 2.12 shows the reflectance functions of
a human face.

Then the reflectance functions are used to change the illumination on the object. A given environment map corresponding to the new illumination is normalized by multiplying each pixel by its solid angle $sin(\theta)$ and down sampled so that it has the same size as each reflectance function. Then the following sum is computed:

$$\forall\text{ pixel } (x,y)\text{ of the object } L(x,y) = \sum_{(\theta,\phi)} R_{x,y}(\theta,\phi)L_i(\theta,\phi)$$  \hspace{1cm} (2.2) \\

with :

- $L(x,y)$ is the radiance value of pixel $(x,y)$ of the object under the new illumination condition
- $\sum_{(\theta,\phi)}$ is the sum over all $(\theta,\phi)$ in the reflectance function
- $R_{x,y}(\theta,\phi)$ is the value of the reflectance function of pixel $(x,y)$ in the direction $(\theta,\phi)$
- $L_i(\theta,\phi)$ is the value of the normalized environment map in the direction $(\theta,\phi)$.

For each pixel of the final rendering, equation 2.2 is a linear combination of the different illumination conditions. Therefore image based-relighting uses the linearity of light transport.

Figure 2.12: Reflectance functions for a human face. Each element of the matrix is a latitude longitude map that corresponds to a pixel of the face under all the illumination conditions. [11]
2.3. IMAGE-BASED RELIGHTING

2.3.3 Changing the viewpoint

Debevec et al. also describe a technique that uses the light stage to render the human face from another viewpoint [11]. The light stage is here equipped with many cameras in order to capture the reflectance field from different viewpoints. Then for any viewpoint a new reflectance field is extrapolated using a model that describes how light is scattered in the skin. Skin produce both specular and diffuse reflections. Unlike the diffuse component, the specular component depends on the viewpoint. Hence in order to render the human face from a new viewpoint, the specular and diffuse components have to be extrapolated separately.

One way to separate these components is to use polarization. The specular reflection preserves polarization of light whereas the diffuse component after subsurface scattering does not preserve it. Hence if polarizers are used on both the camera and the light source, the specular component can be isolated. However this technique is difficult to put in practice as the direction of the polarizers has to be adjusted for each new position of the light. Moreover twice the number of cameras is required in order to separate and capture both the specular and diffuse components. As a result a color space analysis is preferred. This technique is related to the work of Sato and Ikeuchi [38].
The following vectors are defined:

- \( \vec{d} \) is a 3x1 vector (RGB space) of the diffuse color of the object
- \( \vec{s} \) is a 3x1 vector (RGB space) of the specular color of the object. The specular color corresponds to the color of the light source.
- \( \vec{e} = \vec{d} \times \vec{s} \) is a 3x1 vector that corresponds to the error

The reflectance field of the pixel \((x,y)\) can be written as:

\[
R_{x,y}(\theta, \phi) = \mu_d \vec{d} + \mu_s \vec{s} + \mu_e \vec{e}
\]  

(2.3)

in which \((\mu_d, \mu_s, \mu_e)\) are coefficients that can be calculated by inverting the linear system of equations. The value of the vector \(\vec{s}\) can be easily evaluated as it corresponds to the color of the light source. However the value of \(\vec{d}\) is harder to find and has to be calculated with equation 2.4.

\[
\vec{d} = \text{normalise}(\vec{d}_0 + f(\theta_i, \theta_r)(\vec{d}_0 - \vec{s}))
\]  

(2.4)

with:

- \(\vec{d}_0\) is a 3x1 vector (RGB space) called the uniform diffuse chromaticity
- \(\theta_i\) is the direction of the incoming light
- \(\theta_r\) is the direction of a given camera
- \(f(\theta_i, \theta_r) = \alpha_0 \cos(\theta_i) \cos(\theta_r) + \alpha_1 (1 - \cos(\theta_i)\cos(\theta_i))\)

The values of \(\alpha_0\) and \(\alpha_1\) are evaluated such as they best fit the data. Then the specular component is given by \(S = \max(0, \mu_s) \vec{s}\) and the diffuse component corresponds to a picture of the reflectance field in which the specular component is removed: \(D = R - S\).

The reflectance field from a new viewpoint is created separately for the specular and the diffuse component. Indeed unlike the diffuse component, the specular component depends on both the new viewing direction and the incoming light direction. Considering the specular reflectance function of a given pixel \((x,y)\) of the object, the value of the radiance \(L_{\vec{v}_0}\) located at \((\theta_i, \phi_i)\) in that reflectance function, corresponds to the specular response of the object for the original viewing direction \(\vec{v}_0\) and for the light source direction \(\vec{l}_p\). The specular response of a surface is related to its behavior at the microscopic level. As a result it is evaluated using the Torrance-Sparrow microfacets model [42]. The new specular reflectance function corresponding to the viewing direction \(\vec{v}_n\) is then calculated with equation 2.5.

\[
L_{\vec{v}_n} = L_{\vec{v}_0} \frac{G(\vec{v}_n, \vec{l}_p, \vec{n})F(\vec{v}_n, \vec{l}_p)(\vec{v}_0, \vec{n})}{G(\vec{v}_0, \vec{l}_q, \vec{n})F(\vec{v}_0, \vec{l}_q)(\vec{v}_n, \vec{n})}
\]  

(2.5)

with:

- \(\vec{n}\) the normal at pixel \((x,y)\) of the object
- \(F\) the Fresnel reflectivity
- \(G\) a geometric attenuation term
- \(\vec{l}_q\) defined by: \(\vec{l}_q = 2(\vec{H}.\vec{n})\vec{H} - \vec{n}\) with \(\vec{H}\) the half-way vector between the light source and the new viewing direction.
Each pixel of the new diffuse reflectance function is then calculated using equation 2.4 and the original diffuse reflectance function. As the equation depends on the viewpoint $\theta$, it is shifted twice. First with an angle corresponding to the opposite of the first viewpoint of $-\theta_{r_0}$ and then with an angle corresponding to the new viewpoint $\theta_{r_n}$. Instead of $\vec{d}_0$ the value of the original diffuse reflectance function is used. Once the new reflectance fields are synthesized, the method explained in section 1.1.1 is used in order to generate the illumination of the human face from a new viewpoint. This illumination is then projected on a geometric model of the face under the new viewpoint.

2.3.4 Evolutions of the light stage

The light stage technology has been improved several times over the years. It has known drawbacks such as the time needed for capture. For instance, light stage 1 captures the reflectance field of a human face in one minute [9] whereas the light stage 2 [23] invented in 2000 only requires eight seconds. The second version of the light stage uses an arc of thirty lights that makes the capture faster. Animated facial expression can be also produced with light stage 2. Light stages 3, 4, 5 and 6 have lights sources located on a dome. The object is inside a sphere and can be illuminated from any direction. Light stage 3 introduced high dynamic range color lights that can recreate the illumination of an environment (e.g. the Grace Cathedral). Light stage 5 [48] introduced a high-speed camera that can capture the reflectance field almost in real time. Indeed the light stage changes the illumination conditions so quickly that they are not perceptible to the eye but can be seen with a high-speed camera. Light stage 6 is an alternative version of light stage 5 that uses a dome with a diameter of 8 meters. It can acquire the reflectance field of an animation of the entire body in order to render it in a computer generated scene. Although light stage 4 was presented in 2002 it has never been made. The last generation of light stages uses polarized light sources and cameras to capture both specular and diffuse components separately using a technique presented by Ma et al. [30]. The resulting scans have a higher resolution than previous light stages and let capture the pores of the skin for instance. Besides almost every facial expression can be captured as the time needed for acquisition is very short. The light stage X includes all these enhancements as well as a multi-view acquisition of the face described by Ghosh et al. [21].

2.4 The free-form light stage

Using a light stage gives good results for image-based relighting as it provides a good sampling of incoming light directions. However the light stage technology does not scale easily for big objects such as a car or a tree. Masselus, Dutré and Anrys [33] present the free-form light stage which overcome that difficulty using a portable light source and a fixed camera. Hence unlike the light stage where the light source takes given positions, here the light source can take any position in a whole sphere around the object. The reflectance field captured is again the non-local reflectance field defined by Debevec et al. [11].

Their method is divided in three stages. Firstly the reflectance field of the object is acquired by taking many pictures of it with different positions of the light source. These positions correspond to a non-uniform sampling. Four white diffuse spheres are located around the object and must appear in the pictures (figure 2.14). These spheres are very important for the second step of the method. Usually between four hundred and five hundred pictures are taken. Using few images gives poor results as more samples of the incoming light direction are needed to recreate a correct reflectance field. The spheres have to be correctly placed around the object during the setup otherwise problems may appear. For instance, a part of the object could be hidden by one of the spheres. Likewise, a part of a sphere could appear in the object if it has a high specular component. Besides a sphere could create a shadow on the object that would produce artefacts during the relighting phase.

In the second step, for each picture the white diffuse spheres are used to derive the direction of the incoming illumination. Recovering the direction of the incoming light is possible using the variation of intensity (shadows) on each sphere. As a result, when taking pictures in the first step
care has to be taken that the object does not produce a shadow on one of the spheres. In that case, the picture has to be discarded as it would give incorrect results.

In each image the four spheres are extracted and for each sphere the silhouette direction denoted $\vec{s}_i$ is calculated (figure 2.15). As the radius of each sphere is known, the position of the center $C_i$ can be deduced by minimizing equation 2.6.

$$\|C_i\|^2 - R^2 - (\vec{s}_i \cdot \vec{C}_i)^2 = 0$$  \hspace{1cm} (2.6)

Then for every sphere of a given picture, the normals at each illuminated pixel are calculated. A list of normals is obtained and as the intensity of the light source $I_L$, the intensity of a current point on the sphere $I_p$, the albedo of the diffuse sphere $\rho$, and the normals $\vec{N}_p$ are known the incoming light direction $\vec{L}$ can be derived using a least squares method and Lambert’s Law (equation 2.7).

$$I_p = \rho I_L (\vec{N}_p \cdot \vec{L})$$  \hspace{1cm} (2.7)

The result obtained for $\vec{L}$ best fits the data. If the albedo of the diffuse spheres is unknown, it can be evaluated using a reflectance chart as a reference. Indeed if a picture of the diffuse sphere and the reflectance chart is taken, then the albedo can be derived by comparing the values of the pixels. At the end of this stage each picture of the reflectance field has an incoming light direction associated.

The final step is the relighting of the object. In this stage the incoming light directions obtained earlier are positioned on the mirror ball used for the relighting. Then the mirror ball is divided into cells using an angular Voronoi partition. A Voronoi diagram is a graph made of edges and vertices (the incoming light directions) such as every edge is at equal distance of the two vertices it separates. Each cell in the diagram contains a unique vertex. The angular Voronoi diagram takes into account the angular shape of the mirror ball (figure 2.16). Voronoi diagrams have the property that the pixels in a given cell are the closest pixels to a given light direction [29]. A low sampling
of incoming light directions in an area would produce large cells in the Voronoi diagram.

Once the Voronoi diagram is obtained, each cell is integrated in order to get the radiance value of the cell. The solid angle for each pixel has to be taken into account during the integration as a latitude-longitude parameterisation is used. Each cell now corresponds to an original picture (given by the incoming light direction that is to say the vertex) and has a weight. Finally the object is relit using a linear combination of the pictures taken in step one and the weights.

Figure 2.15: The parameters used to derived the normal at an illuminated point on the sphere are described on this figure. Source: [33]

Figure 2.16: **Left**: An environment map with the Voronoi cells. **Right**: The corresponding radiance values after integration. Source: [33]
2.5 Optimisation of the projection on the lighting basis

One important problem of image-based relighting is the number of illumination conditions needed in order to relight an object. Tunwattanapong, Ghosh and Debevec [43] address this problem using the decomposition of an environment map on the spherical harmonics and local lights basis. Spherical harmonics capture low frequencies of the environment map whereas local lights are suited for high frequencies. Such a decomposition gives a good approximation of the environment map with only a few light sources.

More specifically, the environment map is first decomposed into \( n \) spherical harmonics that captures the low frequencies of the map. Then the result is subtracted from the original environment map giving the residual map. Hence the residual environmental map only contains high frequencies (see figure 2.17). It is sampled using \( m \) uniformly distributed point lights. The \( n + m \) components correspond to a lighting basis that approximates the environment map. This approximation can be improved by solving an optimisation problem. A weight is associated to each of the \( n + m \) components and the goal of the optimisation is to find the combination of weights that gives the best approximation of the environment map. Mathematically, this corresponds to the minimisation of equation 2.8. This optimisation problem can be solved with an algorithm that minimises a nonlinear function.

\[
\min_x f(x), f(x) = \|Ax - y\|_2
\] (2.8)

with:

- \( y \) is the column vector of the environment map
- \( x \) is the column vector of weights. The \( i^{th} \) is the weight given to the \( i^{th} \) light of the lighting basis.
- \( A \) is the matrix that projects the environment map onto the spherical harmonics and local lights basis.

Tunwattanapong et al. [43] find that solving the optimisation problem in the principal component space with principal component analysis (PCA) leads to a faster computation and better results. More specifically, the eigenvectors of the covariance matrix of \( A \) are calculated and correspond to a matrix \( P \). Then the vectors of the environment map and of the projection on the weighted lights basis are projected into the principal component space. The optimisation problem becomes the minimisation of equation 2.9.
2.5. OPTIMISATION OF THE PROJECTION ON THE LIGHTING BASIS

\[
\min_x f(x), \quad f(x) = \|A^T Ax - A^T y\|_2 
\] (2.9)

That decomposition of the environment map decreases the number of pictures needed when the reflectance field is captured. A light stage with high-dynamic range lights such as light stage 5 can be used to create an illumination corresponding to a specific spherical harmonic of the environment map. Then a picture of the object under that specific illumination gives directly the value of the corresponding coefficient of the spherical harmonic. Only \(n + m\) coefficients are needed, therefore only \(n + m\) pictures are taken during the capture, \(n + m\) being very small compared to the number of pictures needed with the original light stage. For instance the best results are obtained with \(n = 9\) and \(m = 11\) that is to say 20 light sources instead of 400 illumination conditions required for the free-form light stage. Figure 2.18 presents comparison between the optimised and non optimised results. Figure 2.19 shows a comparison between the optimisation in original and in PCA space.

Figure 2.18: (a) Relighting without optimisation (b) Relighting with optimisation in the original space (c) Relighting with optimisation in PCA space (d) Ground truth for a relighting with 156 illumination conditions. Source: [43]

Figure 2.19: (a) Relighting with optimisation in the original space. (b) Relighting with optimisation in PCA space. (c) Ground truth for a relighting with 156 illumination conditions. Source: [43]

Finally editing the illumination conditions is possible using the spherical harmonics and local lights. Each coefficient in the decomposition can be adjusted to change the influence of the corresponding light. This editing is not computationally expensive as few data is stored (\(n + m\) coefficients). Besides for high frequency illumination, Gaussian light sources can be used instead of point light sources in order to create renderings closer to the truth.
Chapter 3

Theoretical background

This chapter explains the mathematical background that is needed to understand the algorithms implemented in this project.

3.1 Coordinates systems

The light stage illuminates an object from several directions that are sometimes given in Cartesian coordinates and sometimes in spherical coordinates. Besides spherical coordinates are used to parameterise environment maps in a latitude longitude format. As a result transformations are required to change between these coordinate systems. This section presents the Cartesian and spherical coordinate systems along with the corresponding transformations. My main reference is the book Physically based rendering : from theory to implementation by Pharr and Humphreys [36].

3.1.1 Cartesian coordinates

In a three dimensional space, the Cartesian coordinate system is the most simple one. It uses three distances, one for each dimension, to identify a point. In this system a point is denoted $M(z, x, y)$.

Figure 3.1: Cartesian coordinate system
3.2. SOLID ANGLE

Latitude longitude environment maps use \((z, x, y)\) as a direct basis that is to say the axis that goes up is the y axis. As a result this is the basis chosen in this project an in every algorithm related to image-based relighting.

3.1.2 Spherical coordinates

The spherical coordinate system is very important in graphics as many functions are defined over a sphere of direction. Hence they are easily represented with this coordinate system. A point \(M\) is represented by two angles and its distance to the origin \(O\). \(M\) is denoted \(M(\rho, \theta, \phi)\) in the spherical coordinate system.

\[
\begin{align*}
\rho &= \sqrt{x^2 + y^2 + z^2} \\
\theta &= \arccos\left(\frac{y}{\rho}\right) \\
\phi &= \arctan\left(\frac{x}{z}\right)
\end{align*}
\]

\[
\begin{align*}
z &= \rho \sin(\theta) \cos(\phi) \\
x &= \rho \sin(\theta) \sin(\phi) \\
y &= \rho \cos(\theta)
\end{align*}
\]  

Figure 3.2: Spherical coordinate system

Equation 3.1 correspond to the transformation from Cartesian to spherical and equation 3.2 is the converse transformation.

3.2 Solid angle

Solid angles are two dimensional angles in a three dimensional space. They are a very important unit of measure in computer graphics as three dimensional objects are used. For instance, solid angles measure how widespread a light source is in a light probe.

An angle in a two dimensional plane is a one dimensional unit often measured in degrees or radians. Given the circle on figure 3.3 with a radius of \(\rho\), the angle is defined by the ratio \(\alpha = \frac{A}{\rho}\) with \(A\) the length of the arc whose angle is being measured [46].

The solid angle is defined similarly to the angle. It corresponds to a two dimensional measure in steradians (unit \(sr\)) in a three dimensional space. Given the disk on figure 3.3, the solid angle is defined by equation 3.3 [47]. More specifically, any object can be projected on a sphere of radius \(\rho\) and therefore has a solid angle defined by equation 3.3.

\[
\Omega = \frac{A}{\rho^2}
\]  

(3.3)
CHAPTER 3. THEORETICAL BACKGROUND

Figure 3.3: (a) Definition of an angle in a plane $\alpha = \frac{A}{\rho}$. (b) Definition of a solid angle in a three dimensional space $\Omega = \frac{A}{\rho^2}$.

Given a latitude longitude map, the solid angle of each pixel has to be taken into account in the calculations. Indeed in the latitude longitude format a small patch in the middle of the environment map and a patch at the top (or the bottom) have the same area. However, on a sphere an area at the top of the sphere is smaller than an area at the middle of the sphere (Figure 3.4). In order to compensate this, each pixel of the environment map has to be multiplied by the Jacobian $|\sin(\theta)|$ of the transformation from spherical to cartesian coordinates [20].

Figure 3.4: Left : Two patches defined on a sphere. Right : The same patches defined on a latitude longitude map. The size of the pink patch is bigger with this parameterisation. Source : [20]
3.3 Matrix theory

Matrices are very important in this project and in computer graphics in general. Indeed a picture on a computer correspond to matrix of pixels. Besides image-based relighting uses the projection of an environment map on a given lighting basis. This can be mathematically described with matrices. Finally the matrix theory is needed to understand principal component analysis that appears in the optimisation process for image-based relighting used by Tunwattanapong, Ghosh andDebevec [43]. My main reference for the matrix theory is the book Matrices Theory and Applications by Serre [39].

In this section \( \mathbb{K} \) denotes either the field \( \mathbb{R} \) or \( \mathbb{C} \).

3.3.1 Basic concepts

Let \( n, p \) and \( q \) three positive integers in \( \mathbb{N} \).
\( \mathfrak{M}_{n,p}(\mathbb{K}) \) denotes the set of matrices with \( n \) rows and \( p \) columns and \( \mathfrak{M}_n(\mathbb{K}) \) the set of square matrices. The identity matrix of \( \mathfrak{M}_n(\mathbb{K}) \) is denoted \( I_n \) and verifies equation 3.4 and is defined by equation 3.5.

\[
\forall A \in \mathfrak{M}_{n,p}(\mathbb{K}) \quad AI_p = I_n A = A. \tag{3.4}
\]

\[
I_n = \begin{pmatrix} 1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 1 \end{pmatrix} \tag{3.5}
\]

Let \( A = (a_{i,j})_{1 \leq i \leq n}^{1 \leq j \leq p} \in \mathfrak{M}_{n,p}(\mathbb{K}) \) and \( B = (a_{i,j})_{1 \leq i \leq p}^{1 \leq j \leq q} \in \mathfrak{M}_{p,q}(\mathbb{K}) \).

Matrix transpose

The transpose of matrix \( A \) is a \( p \times n \) matrix denoted \( A^\top = (a_{i,j}')_{1 \leq i \leq p}^{1 \leq j \leq n} \). Its elements are given in equation 3.6.

\[
\forall (i,j) \in [1;p] \times [1;n] \quad a_{i,j}' = a_{j,i} \tag{3.6}
\]

If a matrix is equal to its own transpose \( (A^\top = A) \), then it is called a symmetric matrix.

Matrix multiplication

The product of a \( n \times p \) matrix and a \( n \times q \) matrix is a \( n \times q \) matrix. The coefficients of \( C = AB = (c_{i,j})_{1 \leq i \leq n}^{1 \leq j \leq q} \in \mathfrak{M}_{n,q}(\mathbb{K}) \) can be calculated with equation 3.7.

\[
\forall (i,j) \in [1;n] \times [1;q] \quad c_{i,j} = \sum_{k=1}^{p} a_{i,k} b_{k,j} \tag{3.7}
\]

The multiplication depends on the size of the matrices. Square matrices of the same size can always be multiplied together but the multiplication is not commutative (the product \( AB \) may be different from the product \( BA \)). Besides, if \( q = 1 \), then \( B \in \mathfrak{M}_{p,1}(\mathbb{K}) \) is a column vector of size \( p \) and the product \( AB \) is a matrix-vector multiplication. The result is a column vector of size \( n \).

Invertible matrices

Definition 1. A square matrix \( A \in \mathfrak{M}_n(\mathbb{K}) \) is invertible if there exist \( B \in \mathfrak{M}_n(\mathbb{K}) \) such that \( AB = BA = I_n \). If \( B \) exists, it is denoted \( A^{-1} \). The set of invertible matrices is designated by \( GL_n(\mathbb{K}) \).
**Orthogonal matrix**

Orthogonal matrices are one of the key elements of principal component analysis.

**Definition 1.** A real-valued square matrix \( A \in \mathbb{M}_n(\mathbb{R}) \) is orthogonal if it verifies \( A^\top A = AA^\top = I_n \). The set of orthogonal matrices is designated by \( O_n \).

From the definition, every orthogonal matrix is invertible and its inverse is its transposed matrix. Besides, the rows (or the columns) of an orthogonal matrix form an orthonormal basis of \( \mathbb{R}^n \).

### 3.3.2 Change of basis

#### Canonical basis

\( \mathbb{K}^n \) is a vector space over \( \mathbb{K} \) of dimension \( n \). The basis described in equation 3.8 is called the canonical basis of \( \mathbb{K}^n \).

\[
e_1 = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \quad e_2 = \begin{pmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{pmatrix}, \quad \ldots, \quad e_n = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}
\]

**Linear mappings and matrices**

Let \( E \) and \( F \) two vector spaces over \( \mathbb{K} \) of dimension \( p \) and \( n \) respectively. Let \( \beta_E = (e_1, \ldots, e_p) \) a basis of \( E \) and \( \beta_F = (f_1, \ldots, f_n) \) a basis of \( F \).

A matrix \( A = (a_{i,j})_{1 \leq i \leq p, 1 \leq j \leq n} \in \mathbb{M}_{n,p}(\mathbb{K}) \) can be described as the matrix of a linear mapping \( u : E \to F \) such as equation 3.9 stands.

\[
\forall j \in [1;p] \quad u(e_j) = \sum_{k=1}^{n} a_{k,j} f_k
\]

The linear mapping \( u \) is well defined as its action on a basis of \( E \) describes its action on any vector of \( E \). Indeed any vector of \( E \) is a linear combination of the vectors that compose any basis of \( E \).

In the case of square matrices, a matrix \( A = (a_{i,j})_{1 \leq i,j \leq p} \in \mathbb{M}_{p}(\mathbb{K}) \) can be described as the matrix of an endomorphism \( u : E \to E \). In this case, \( u \) is defined by equation 3.10.

\[
\forall j \in [1;p] \quad u(e_j) = \sum_{k=1}^{p} a_{k,j} e_k
\]

#### Equivalence and similarity

**Definition 1.** Two matrices \( A \in \mathbb{M}_{n,p}(\mathbb{K}) \) and \( B \in \mathbb{M}_{n,p}(\mathbb{K}) \) are equivalent if there are two matrices \( P \in GL_p(\mathbb{K}) \) and \( Q \in GL_n(\mathbb{K}) \) such as \( B = Q^{-1} A P \).

In this case the matrices \( A \) and \( B \) represent the same linear mapping but different bases of the vector spaces \( E \) and \( F \) are used.

Let \( \beta_E \) and \( \beta'_E \) be the bases of vector space \( E \) and \( \beta_F \) and \( \beta'_F \) be the bases of vector space \( F \). Then \( P \) corresponds to the matrix of change of basis \( \beta_E \) to basis \( \beta'_E \) and \( Q \) corresponds to the matrix of change of basis \( \beta_F \) to basis \( \beta'_F \).
3.3. MATRIX THEORY

Definition 1. Two square matrices $A,B \in \mathcal{M}_n(\mathbb{K})$ are similar if there is an invertible matrix $P \in GL_n(\mathbb{K})$ such as $B = P^{-1}AP$.

In this case the matrices $A$ and $B$ represent the same endomorphism but different bases of the vector space $E$ denoted $\beta$ and $\beta'$. $P$ corresponds to the change of basis matrix from $\beta$ to the basis $\beta'$.

Formula of change of basis for a vector

Let $X \in \mathbb{K}^n$ a column vector of size $n$ and $P \in GL_n(\mathbb{K})$. Then $P$ corresponds to a matrix of change of basis of the canonical basis of $\mathbb{K}^n$ to a basis $\beta$ of $\mathbb{K}^n$. The coordinates of the vector $X$ in basis $\beta$ are the coefficients of the vector $X' = P^{-1}X$.

3.3.3 Eigenvectors and eigenvalues

Eigenvalues and eigenvectors are a very important concept as they simplify the study of matrices to bases in which they are triangle or diagonal.

Definition 1. Let $A \in \mathcal{M}_n(\mathbb{K})$ a square matrix. A scalar $\lambda \in \mathbb{K}$ is an eigenvalue of $A$ if there exist a non zero vector $X \in \mathbb{K}^n$ such as $AX = \lambda X$. In this case $X$ is called the eigenvector of $A$ associated to the eigenvalue $\lambda$.

The set of the eigenvalues of matrix $A$ is called the spectrum of $A$. The eigenvalues of matrix $A$ are the roots of the characteristic polynomial $\text{determinant}(A - XI_n)$. It is sometimes defined as $\text{determinant}(XI_n - A)$ but this only changes the polynomial by a factor of $(-1)^n$.

Diagonalisation

If the characteristic polynomial has $n$ distinct roots then $A$ can be diagonalized and the eigenvectors are a basis of $\mathbb{K}^n$. In this case, let $(\lambda_1, ..., \lambda_n) \in \mathbb{K}^n$ be the $n$ distinct eigenvalues of matrix $A$ and $P \in GL_n(\mathbb{K})$ the matrix of the corresponding eigenvectors (each column of this matrix is an eigenvector of $A$).

In the basis of its eigenvectors, $A$ is a diagonal matrix (equation 3.11).

$$P^{-1}AP = \begin{pmatrix} \lambda_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_n \end{pmatrix} \quad (3.11)$$

The process of calculating the eigenvalues and the eigenvectors of a given matrix is called diagonalization.

Theorem 1 is called the spectral theorem and provides a criteria under which matrices are diagonalizable. It can be applied for covariance matrices that are the key element of principal component analysis.

Theorem 1. A symmetric real-valued matrix is orthogonally diagonalizable. In other words, for all symmetric and real-valued matrix $A \in \mathcal{M}_n(\mathbb{K})$, there exist $O \in O_n$ such as $O^T AO$ is diagonal.

This theorem describes the fact that the endomorphism associated to a real-valued symmetric matrix is diagonalizable in an orthonormal basis.
3.4 Principal Component Analysis

Principal component analysis (PCA) is used in the optimisation process by Tunwattanapong et al. [43] that improves the renderings obtained with image-based relighting. Hence it is necessary to understand how this analysis works in order to understand how the optimisation algorithm is implemented in the project. My main reference for this section is Duncan Gillies’ lectures at Imperial College London [22].

3.4.1 Goal

Given a data set of variables, principal component analysis finds the basis in which the variance of the data set along each dimension is maximized. This is an orthonormal basis called the principal component basis. The projection of the data set into this basis gives the directions in which the data set varies the most. However some dimensions account less than others for the total variance of the data set. Hence these dimensions can be removed without a big loss of information. As a consequence calculations performed in the principal component basis are faster than in the original basis as less coordinates are necessary to represent each data point. This is the main goal of principal component analysis. Figure 3.5 shows an example of a data set with its principal component basis.

![Figure 3.5: Principal component analysis finds the basis (u,v) in which the variance of the data set along each dimension is maximised. Source [22]](image)

3.4.2 Principal component analysis step by step

Let \( N \in \mathbb{N} \) be the number of points in the data set and \( n \in \mathbb{N} \) the number of coordinates for each point. Usually \( n \) is greater than \( N \).

Principal component analysis uses the covariance matrix of the data set in order to find the directions where the data varies the most. First the data set is put into a matrix \( D = (d_{i,j})_{1 \leq i \leq N}^{1 \leq j \leq N} \) (equation 3.12) with \( d_{i,j} \) representing the \( j^{th} \) coordinate of the \( i^{th} \) data point.

\[
D = \begin{pmatrix}
  d_{1,1} & \cdots & d_{1,n} \\
  \vdots & \ddots & \vdots \\
  d_{N,1} & \cdots & d_{N,n}
\end{pmatrix}
\]  

(3.12)
3.4. PRINCIPAL COMPONENT ANALYSIS

The mean $\mu$ of the data set is also calculated. It is a row vector which corresponds to the average of the rows in the matrix. Then the mean is subtracted from each row of the matrix giving a mean centered matrix $U$ that describes the data set. From this matrix, the covariance matrix of the data set is computed with equation 3.13. It is a $n \times n$ matrix.

$$\Sigma = \frac{U^TU}{N-1}$$ (3.13)

The covariance matrix is a real-valued and symmetric matrix. Hence it is orthogonally diagonalisable with the spectral theorem. Its eigenvectors are called the principal components of the data set and corresponds to the sought basis (basis $(u,v)$ in figure 3.5). These eigenvectors have to be ordered by eigenvalues in a decreasing order. Indeed the eigenvector whose eigenvalue is the highest corresponds to the direction in which the data set varies the most. The eigenvector whose eigenvalue is the lowest corresponds to the direction in which the data set varies the least. Hence the number of dimensions that represent the data can be reduced by removing the less significant principal components that is to say by removing the eigenvectors with the lowest eigenvalues. Information is lost in this process that is why the correct number of dimensions has to be removed. If $(\lambda_1, ..., \lambda_n)$ denotes the eigenvalues of the covariance matrix in a decreasing order ($\lambda_1 > ... > \lambda_n$), then the percentage of variance that an eigenvector accounts for is given by equation 3.14.

$$\forall i \in [1; n] \ \alpha_i = \frac{100\lambda_i}{\sum_{i=1}^{n} \lambda_i}$$ (3.14)

Therefore a possible way to determine how many principal components can be removed is to compute every $\alpha_i$ and find $k \in \mathbb{N}$ such as equation 3.15 stands. $\tau$ is a defined threshold. A typical value for $\tau$ would be 99%.

$$\sum_{i=1}^{k} \alpha_i > \tau$$ (3.15)

Any data point $p_{\text{original}}$ in the original basis can be projected into the principal components basis with equation 3.16

$$p_{\text{pca}} = (p_{\text{original}} - \mu)\Phi$$ (3.16)

with:

- $p_{\text{original}}$ is a row vector of the coordinates of a point in the original basis.
- $p_{\text{pca}}$ is a row vector of the coordinates of a point in the principal component basis.
- $\mu$ is the mean of the data set (row vector).
- $\Phi$ is the matrix of change of basis from the original basis to the principal component basis.

The columns of $\Phi$ are the eigenvectors of the covariance matrix.

3.4.3 Example on images

Principal component analysis is often used in image processing. A data set of $N$ images of size $500 \times 500$ can be represented by $N$ row vectors, each vector having $n = 500 \times 500 = 250000$ coordinates. Each row vector corresponds to an image in which the rows are concatenated together. Each coordinate of the vector corresponds to a specific pixel. Hence the data set of $N$ images can be represented by a matrix of size $N \times n$. Equation 3.17 gives the form of that matrix, with $p_{i,j}$ being the $j^{th}$ pixel ($j \in [1; n]$) of image $i \in [1; N]$.
Thus principal component analysis can remove many coordinates to represent the data in a simpler way. This leads to faster computations with only a controlled loss of information. As images have many pixels, principal component analysis is a very interesting method for computer graphics.

3.5 Gamma correction

The gamma correction is a correction applied on computer screens to display the correct value of the pixels of an image. It is very important in this project as the results have to be correctly displayed in order to be evaluated. My reference for the following explanation is the work of Poynton [37].

Intensity also called irradiance is a quantity that corresponds to the amount of light received per surface unit. Its unit is $W/m^2$. Cameras capture the intensity in a linear manner that is to say on the sensor of a camera a pixel with a value of 255 is twice as bright in terms of intensity as a pixel with a value of 128. However when that picture is displayed on a computer screen, a pixel with a value of 128 on the camera has a lower value on the screen leading to a dark image. This is due to the fact that screens do not display the pixels values with a linear response. The actual value of a pixel displayed by a screen is given by equation 3.18 and is called the gamma curve. The curve is plotted on figure 3.7.

$$p = 255 \times \left( \frac{I_{255}}{I} \right)^\gamma$$  \hspace{1cm} (3.18)

with :

- $p$ is the value of the pixel displayed by the screen. $p$ can take values between 0 and 255.
- $I$ is the intensity of the pixel captured by the camera. Its values are between 0 and 255. $\frac{I}{255}$ corresponds to the intensity of the pixel.

The typical values for gamma are in the range $[1.0; 3.0]$ [18]. For instance, if a pixel has a value of $I = 128$ on the camera, the value displayed by the screen with a gamma of 2.2 will be $p = 255 \times \left( \frac{128}{255} \right)^{2.2} = 56$. Hence the displayed result has only 44% of its real intensity value. The idea behind the gamma correction is to retrieve the original linear intensity values by applying an inverse gamma curve (equation 3.19) to the values of the camera. As a result the values displayed by the computer screen are linear and the picture has the correct appearance. Diagram 3.6 describes the process.

$$p = 255 \times \left( \frac{I}{255} \right)^{\frac{1}{\gamma}}$$  \hspace{1cm} (3.19)

In computer vision and graphics, images are processed with the linear intensity values of the camera that is to say without gamma correction. Hence care has to be taken that the calculations...
3.5. GAMMA CORRECTION

Figure 3.7: Gamma curve of 2.2 and its corresponding inverse

are not performed on gamma corrected images. High dynamic range pictures (section 2.2) retrieve the linear intensity values of a scene. Hence no gamma correction has to be applied on HDR pictures as their response curve is already linear.
Chapter 4

Image-based relighting with light stage 6

This project deals with the capture of the reflectance field of an object in a regular office room instead of a light stage in order to achieve the relighting of the object under any illumination condition. Not using a light stage for the capture is a challenging task as an office room does not create known and controlled illumination conditions. However several steps of image-based relighting do not depend on the way the reflectance field is captured. As a result the understanding and the correct C++ implementation of the light stage relighting is an important first stage in this project as a part of the code can be adapted for the office room relighting. This section presents how I implemented the light stage relighting and compare my results to the results obtained by Paul Debevec[10].

4.1 Data used

Paul Debevec provides reflectance fields of objects captured with light stage 6 on his website [10] in the Light Stage Data Gallery section. This data can be used to compute the light stage relighting of a given object. Light stage 6 (see figure 4.1) is a dome of lights with two-thirds of the light sources located on the dome and one third located on the ground to simulate the light coming from the floor [40]. For each object, the reflectance field corresponds to 253 pictures of the object illuminated from different directions (see figures 4.2 and 4.3). Reflectance fields of six objects are currently available: a fighting knight, a standing knight, a kneeling knight, a helmet viewed from the front or the side and a plant.

As Paul Debevec explains on his website, the pictures provided have a gamma correction (see section 3.5) of 2.2 applied. This correction has to be removed before computing the result to retrieve the original linear intensity values given by the camera. Skipping this step would produce incorrect final results. Moreover, care has to be taken that the value of the pixels without the gamma correction is calculated with float values between 0.0 and 1.0 for each pixel of the image. Therefore if a picture is loaded as an 8 bits unsigned char picture and its pixels are denoted $p_{R/G/B}$ depending on the color channel, the equation to remove the gamma is given by the equation 4.1.

$$p_{R/G/B \ without \ gamma} = \left( \frac{p_{R/G/B}}{255} \right)^{2.2}$$  \hspace{1cm} (4.1)

Besides two text files that contains information about the light stage and the light sources used during the capture are provided. The first text file contains the light intensities for each channel (red, green and blue) of the 253 light sources of light stage 6. This information is very important to calculate the final relit result (see section 4.2.3). The second text file contains the light directions given in Cartesian coordinates of the 253 light sources of the light stage. These directions are useful to know how the sphere of the incoming light directions is sampled during the data capture. However the website does not specify if the directions are given from the light source towards the object or from the object towards the light source. This information is important to avoid wrong calculations in the next steps. Fortunately it can be derived from the pictures of the reflectance field.
4.1. DATA USED

Figure 4.1: A picture of the interior of light stage 6. Source [17]

Figure 4.2: One image of the reflectance field of the plant: the light comes from the top. Source [10]

Figure 4.3: Another image of the reflectance field of the plant: the light comes from the bottom left. Source [10]

Figure 4.4 shows picture 52 of the reflectance field of the helmet and displays the coordinate system used for the relighting of the object. The file that contains the light directions for the light stage gives for the 52nd illumination condition a negative x and y and a positive z. As the light comes from the top right of the helmet in the picture, it can be derived that the light directions given in Paul Debevec’s file are from the light source towards the object. In this project, the object is taken as a reference hence the opposite direction is used for the calculations.
CHAPTER 4. IMAGE-BASED RELIGHTING WITH LIGHT STAGE 6

Figure 4.4: Picture 52 of the reflectance field of the helmet: the light comes from the top-right. The corresponding direction has a negative x and y in Paul Debevec’s file. Source [10]

4.2 Image-based relighting step by step

The complete relighting of an object can be done in several steps. This section explains in details how these steps are computed in this project.

4.2.1 Spherical coordinates

The incoming light directions given in the text file are in Cartesian coordinates. However the environment maps I used are latitude longitude maps and are parameterized with spherical coordinates. Hence the first step consists in reading the text file and convert each incoming light direction to spherical coordinates. Then these coordinates are scaled to the size of the environment map and plotted on it (figure 4.5). For instance if the environment map has a width of 1024 pixels and a height of 512 pixels, then the spherical coordinate $\phi$ is scaled from the range $[0; 2\pi[$ to the range $[0; 1024]$ and $\theta$ is scaled from $[0; \pi]$ to the range $[0; 511]$.

Figure 4.5: Plot of the light stage directions on the Grace Cathedral environment map
4.2. IMAGE-BASED RELIGHTING STEP BY STEP

Implementation details

The conversion from Cartesian to spherical coordinates as presented in section 3.1.2 requires the arctangent function. The C++ math.h library has two functions to compute the arctangent, denoted atan and atan2. Their signatures are:

- double atan(double x); returns a value in the range $\left[-\frac{\pi}{2}; \frac{\pi}{2}\right]$  
- double atan2(double y, double x); returns a value in the range $[-\pi; \pi]$. It computes $\text{atan}(\frac{y}{x})$ for two real values $(x, y)$ with $y$ non-zero.

The mathematical function that corresponds to the arctangent is the first one: atan. atan2 computes the value of a quotient which is useful for the Cartesian to spherical coordinates conversion. According to the Linux manual [6], for a given point $M(x, y)$ atan2 is designed to compute the correct value of the angle $\Phi$ of the vector $\overrightarrow{OM}$ (see figure 4.6) depending on the sign of each coordinate $x$ and $y$. The value of the angle being in the range $[-\pi; \pi]$. Therefore I used atan2 in my implementation.

When the first function calculates $\text{atan}(\frac{y}{x})$ the result is in the range $\left[-\frac{\pi}{2}; \frac{\pi}{2}\right]$. Hence it does not correspond to the value of the angle for a negative x. Besides the atan function returns an error in a few cases. For instance $\text{atan}(\frac{1}{0^+})$ is not defined whereas $\text{atan2}(\frac{1}{0^+})$ returns $\frac{\pi}{2}$ which is the limit of the arctangent function.

![Figure 4.6: In this case atan2(x, y) returns 3\pi/4 for \(\Phi\) whereas atan(\(\frac{y}{x}\)) returns -\pi/4.](image-url)
4.2.2 Voronoi diagram

The incoming light directions of the light stage now correspond to points plotted on the environment map. Similarly to the method explained by Masselus et al. [33] for the Free-form Light Stage, the next step is to calculate the Voronoi diagram obtained with these points. As explained in section ??, a Voronoi diagram is composed of cells that contains the pixels that are the closest to a considered direction. Hence it associates an area of the environment map to an incoming light direction.

![Figure 4.7: Plot of the light stage directions and the corresponding Voronoi diagram on the Grace Cathedral environment map](image)

The C++ library OpenCV [25] is used to compute the Voronoi diagram and to paint it on the environment map (figure 4.7). In the Voronoi diagram, each cell has a center that corresponds to an incoming light direction in the light stage. This direction has itself a corresponding image in the reflectance field, which is a picture of the object illuminated from that direction. Hence each cell in the diagram corresponds to a picture in the reflectance field of the object.

Light stage 6 has 253 lights located on a dome. Two thirds of the lights are located in the upper hemisphere and one third are on the ground in order to simulate lighting coming from the floor [40] (figure 4.1). Hence the Voronoi diagram has bigger cells at the bottom as the light stage has less light sources in that area. Besides the cells in the diagram are also bigger in the middle of the latitude longitude map as it represents the back of the environment. The number of samples in this region of the sphere is smaller because the light coming from that area does not influence the relighting significantly. Hence the reflectance field does not contain many pictures of the object illuminated from the back.

4.2.3 Integration of the Voronoi cells

After the calculation of the Voronoi diagram each cell is integrated in order to get a weight for the corresponding image. These weights are then used to compute the final relit result. As described in equation 4.2 each weight has three components, one for each color channel of the image (red, green and blue).

For each cell i \( \omega_i = (\omega_{i,\text{red}}, \omega_{i,\text{green}}, \omega_{i,\text{blue}}) \) (4.2)

Two integration methods are used in this project depending on how the cells of the Voronoi diagram are interpreted. These methods take into account the intensities of each light source in the light stage. These are plotted on figure 4.8. The intensities of the lights located at the back of the sphere are lower than those located in the upper hemisphere. This is due to the fact that during
the relighting process only the front of the object is visible. Hence the lighting coming from the back of the object does influence the final result a lot.

Figure 4.8: Plot of the light stage intensities. The whiter a cell is, the more intense the corresponding light source is.

Point integration

The first integration method is the one used by Masselus et al. [33]. I call it the point integration. For each color channel, the weight of a cell is given by the sum of its pixels, taking into account the solid angle of the pixel and the intensity of the light source. Hence the weight of the cell represents the total amount of energy in that cell for each color channel. The formulas for the weights considering point light sources are given in equations 4.3, 4.4 and 4.5 and are plotted on figure 4.9.

\[
\text{For each cell } i \quad \omega_{i,\text{red}} = \sum_{\text{pixel } p \in C_i} p_{\text{red}} \sin(\theta_p) I_i
\]
\[
\text{For each cell } i \quad \omega_{i,\text{green}} = \sum_{\text{pixel } p \in C_i} p_{\text{green}} \sin(\theta_p) I_i
\]
\[
\text{For each cell } i \quad \omega_{i,\text{blue}} = \sum_{\text{pixel } p \in C_i} p_{\text{blue}} \sin(\theta_p) I_i
\]

with:

- $p_{\text{red/blue/green}}$ is the value of the current pixel for one of its three color channels
- $I_i$ is the intensity of the light source corresponding to cell $C_i$
- $\sin(\theta_p)$ is the solid angle for pixel $p$
- $C_i$ is the set that contains the pixels of cell $i$
Figure 4.9: The weights of each cell for a point integration for the Grace Cathedral environment map.

**Gaussian integration**

The intensity of a light source is usually decreasing with the distance to its center. As the center of a Voronoi cell represents an incoming light direction, the pixels that are closer to the center of the cell have to be taken into account more importantly than the pixels that are far from the center during the integration. The weight of the cell is therefore modulated by a two dimensional Gaussian function. The formulas for the resulting weights are given in equations 4.6, 4.7 and 4.8.

For each cell $i$

$$\omega_{i,\text{red}} = \sum_{\text{pixel } p \in C_i} p_{\text{red}} \sin(\theta_p) I_i G(\mu_i, \sigma, p_x, p_y)$$  (4.6)

For each cell $i$

$$\omega_{i,\text{green}} = \sum_{\text{pixel } p \in C_i} p_{\text{green}} \sin(\theta_p) I_i G(\mu_i, \sigma, p_x, p_y)$$  (4.7)

For each cell $i$

$$\omega_{i,\text{blue}} = \sum_{\text{pixel } p \in C_i} p_{\text{blue}} \sin(\theta_p) I_i G(\mu_i, \sigma, p_x, p_y)$$  (4.8)

where

- $p = (p_x, p_y)$ are the pixels coordinates in the latitude longitude map.
- $p_{\text{red/blue/green}}$ is the value of the current pixel for one of its three color channels
- $I_i$ is the intensity of the light source corresponding to cell $C_i$
- $\sin(\theta_p)$ is the solid angle for pixel $p$
- $G(\mu, \sigma, x, y) = \frac{1}{2\pi\sigma_x\sigma_y} e^{-\frac{(x-\mu_x)^2}{2\sigma_x^2} - \frac{(y-\mu_y)^2}{2\sigma_y^2}}$ is a two dimensional gaussian with mean $\mu = (\mu_x, \mu_y)$ and standard deviation $\sigma = (\sigma_x, \sigma_y)$.
- $C_i$ is the set that contains the pixels of cell $i$.

For each cell $i$, the mean of the gaussian denoted $\mu_i = (x_{\text{center } i}, y_{\text{center } i})$ is the center of the Voronoi cell. The standard deviation of the gaussian is chosen empirically depending both on the size of the light sources and the size of the cells in the Voronoi diagram. A wide gaussian gives brighter results as many pixels around the center of each Voronoi cells are taken into account. In contrast a narrow gaussian gives darker result as the integration of the cells gives low weights. The
influence of the variance of the Gaussian on the final relit result is described in section 4.3.1. The weights for Gaussian light sources are shown on figure 4.10.

The final weights correspond to a piecewise constant approximation of the environment map. The higher the number of lighting conditions, the higher the number of weights and the better the environment maps will be approximated by the weights. However the drawback of a very high sampling is the high number of pictures needed for the reflectance field and the amount of memory required for the computation of the relighting.

For instance in the light stage relighting of the plant, 253 images of size 1312x864 are loaded. Each pixel has three color channel that are converted to 32-bits (4 bytes) floating point numbers for the calculations. Hence each image takes $4 \times 3 \times 1312 \times 864 = 13602816$ bytes that is to say 13.6 megabytes in memory. As a result the whole light stage relighting requires $13.6 \times 253 = 3444$ Gigabytes of memory. Most of today’s computers have between 4 and 8 Gigabytes of memory that is why a higher number of pictures in the reflectance field or pictures with a higher definition would require much more memory than what current computers have.

### 4.2.4 Normalisation step and final result

The final result of the relighting is a linear combination of the pictures of the reflectance field as explained in the original paper on image-based relighting [11].

$$X = \sum_{i=1}^{N} \omega_i X_i$$

with :

- $N$ is the number of images in the reflectance field (253 for light stage 6).
- $(X_i)_{i \in [1; N]}$ are the pictures of the reflectance field.
- $(\omega_i)_{i \in [1; N]}$ are the weights for each picture (or each cell).

In the linear combination 4.9, $X_i$ and $\omega_i$ are vectors with three components, one for each color channel. Therefore a multiplication component by component is used in the calculation.
The integration of the cell is computed with high dynamic range environment maps. Hence the sum of the resulting weights can be higher than one, depending on the environment map. For display purposes, the sum of the weights has to be lower or equal to one otherwise the final result is too bright or even completely white. Indeed an image displayed on a screen is made of unsigned char values between 0 and 255 for each color channel. If the sum of the weights is greater than one, then pixels can have values above 255 in the final result. This would result in a very bright image. Therefore a normalisation step is used to display the relit result.

Two points have to be respected in this step. First the sum of the weights for each color channel has to be less or equal to one. Secondly each color channel has to be normalized by the same amount otherwise color shiftings in the final result could appear (figure 4.11).

Figure 4.11: A wrong normalisation leads to a shift in the colors

In order to respect these two constraints I use the following normalisation step. For each color channel the sum of the weights is computed. Three sums are obtained: one for the red channel, one for the green channel and one for the blue channel. Then the maximum of these three sums is used as a normalizing constant for the weights. This step is described by the equations 4.10, 4.11 and 4.12 with \( \omega_i = (\omega_{i,\text{red}}, \omega_{i,\text{green}}, \omega_{i,\text{blue}}) \) that denotes the normalised weights for cell i.

For each cell i \( \omega_{i,\text{red}} = \frac{\omega_{i,\text{red}}}{\alpha} \) \hspace{1cm} (4.10)

For each cell i \( \omega_{i,\text{green}} = \frac{\omega_{i,\text{green}}}{\alpha} \) \hspace{1cm} (4.11)

For each cell i \( \omega_{i,\text{blue}} = \frac{\omega_{i,\text{blue}}}{\alpha} \) \hspace{1cm} (4.12)

with :

\[ \alpha = \max(\sum_{i=1}^{N} \omega_{i,\text{red}}, \sum_{i=1}^{N} \omega_{i,\text{green}}, \sum_{i=1}^{N} \omega_{i,\text{blue}}) \] \hspace{1cm} (4.13)

In order to display the final result, each image of the reflectance field is split into its three color channels and each of these is multiplied by its corresponding weight. Then the sum of the resulting
4.2. IMAGE-BASED RELIGHTING STEP BY STEP

The color channels of the reflectance field is computed. This step is described in equations 4.14, 4.15, 4.16.

\[
X_{\text{red}} = \sum_{i=1}^{N} \omega_{i,\text{red}} X_{i,\text{red}} \tag{4.14}
\]

\[
X_{\text{green}} = \sum_{i=1}^{N} \omega_{i,\text{green}} X_{i,\text{green}} \tag{4.15}
\]

\[
X_{\text{blue}} = \sum_{i=1}^{N} \omega_{i,\text{blue}} X_{i,\text{blue}} \tag{4.16}
\]

Finally the three color channels \((X_{\text{red}}, X_{\text{green}}, X_{\text{blue}})\) are merged to get the final relit result. For a correct display on computer screens, a gamma correction of 2.2 is applied.

4.2.5 Change the background

After the computation of the final result, I implemented a function to change the background of the object. The goal is to make as if the object was in the new environment. This step is important as it is the only way the quality of the relit result can be evaluated. For this purpose I used a mask for each object. A mask is a binary image that is to say an image in which the pixels can only take two values : black (0) for the object and white (255) for the background (figure 4.13). Paul Debevec provides such masks for the objects [10].

The function works as follows. First the mask is read in order to know which pixels are in the background. Each of these pixels has a direction from the camera point of view. Then, assuming an orthographic camera, this direction is calculated (figure 4.12) and converted into spherical coordinates. For the calculations the size of the image is scaled between \([-0.5; 0.5] \times [-0.5; 0.5]\). For instance the direction of the top left pixel of the image is \((-1, -0.5, 0.5)\) in the \((z, x, y)\) basis. Finally the direction in spherical coordinates corresponds to a pixel in the latitude longitude environment map which is the pixel of the background. This process is similar to a ray tracing algorithm.

![Figure 4.12: The red direction for pixel \((x, y)\) of the image is used to find the corresponding pixel of the background.](image)

---

The content is a detailed explanation of image-based relighting, focusing on the computation of color channels of the reflectance field and the process of changing the background after the relit result is computed. The equations for color channel computation are provided, followed by a description of the background change function, including the use of masks and the conversion of pixel directions into spherical coordinates.
4.3 Results

4.3.1 Point light sources and Gaussian light sources

The results I obtained for the relighting of the helmet and the plant are shown in figures 4.14, 4.15, 4.16, 4.17 and 4.18. Each picture of a relit object takes approximately 20 seconds to compute on a computer with 8 gigabytes of memory and a 1.7 GHz Intel Core i7 processor with 2 physical cores (4 logical cores with the Hyper-Threading technology). My results can be compared to Paul Debevec’s results as he provides the helmet and the plant relit in the Grace cathedral environment map (figures 4.19 and 4.20). They both look very similar. However this comparison is only possible for the Grace cathedral as he does not provides results of the relighting in other environments.

I also compared the results obtained with point integration and Gaussian integration. The more narrow the two dimensional Gaussian is, the more the contrast that is to say the difference between the bright and dark areas of the result increases. The wider the Gaussian is, the more the result of the Gaussian integration tends to the result of the point integration. The difference in contrast between point and Gaussian integration can be seen on figure 4.21. Gaussian integration give results that are closer to reality as only the energy that is very close to the incoming light directions is considered. More results of the objects relit in different environment maps are available in appendix B.

4.3.2 Videos of the results

Similarly to the presentation of the results that Paul Debevec does on his website, I made videos of the relighting of the objects in their environment. While the environment rotates, the relighting of the object changes.

This is done by generating 360 pictures of the relit object with 360 different rotations of the environment map (one picture per degree of the environment map). The Voronoi diagram does not change between each picture but the environment map rotates (figure 4.22). Each time a new picture is generated the azimuthal angle ($\phi$ angle in spherical coordinates) is increased by one degree. Then I used Apple iMovie software [2] to assemble the 360 pictures and create a video of the object.
4.3. RESULTS

Figure 4.14: Relighting of the helmet in the Grace cathedral using a point integration. Original rotation of the environment map.

Figure 4.15: Relighting of the helmet in the Grace cathedral using a point integration with a 90 degrees rotation.

Figure 4.16: Relighting of the helmet in the Grace cathedral using a point integration with a 180 degrees rotation.

Figure 4.17: Relighting of the helmet in the Grace cathedral using a point integration with a 270 degrees rotation.

rotating in the environment. Each picture lasts 0.1 second, hence the video has only 10 frames per seconds that is why the animation is not smooth (a video needs to have at least 24 frames per seconds in order to have a smooth animation). This is a limitation of iMovie as 0.1 second is the lowest time frame I can use to display a picture.
Figure 4.18: Two renderings of the plant in the Grace Cathedral with a 180 degrees rotation

Figure 4.19: Relighting of the helmet in the Grace Cathedral by Paul Debevec. [10]

Figure 4.20: Relighting of the plant in the Grace Cathedral by Paul Debevec. Source [10]
4.3. RESULTS

Figure 4.21: **Left:** Relighting of the plant in the Grace cathedral with a point integration. **Right:** Relighting of the plant in the Grace cathedral with a Gaussian integration. The Gaussian integration has more contrast and the orange area is brighter.

Figure 4.22: The Grace cathedral environment map with a $-\frac{\pi}{2}$ rotation. The Voronoi diagram does not change.
4.4 Implementation details

The C++ OpenCV library [35] uses matrices to store pictures. The function imread that loads the pictures has two parameters. The first one is the path of the image and the second parameter describes the type of image that is loaded. The pictures of the reflectance field are 8-bits PNG files, hence the flag CV_LOAD_IMAGE_COLOR is used to load images with values between 0 and 255. For all the calculations including the gamma correction and the computation of the linear combination for the final result, the values of the matrices have to be floating points numbers between 0.0 and 1.0. Hence it is necessary to use the convertTo function to convert a matrix from OpenCV 8 bits unsigned chars CV_8C3 to OpenCV 32 bits floats CV_32FC3 and then divide the matrix by 255.

After all the computations the result has to be saved to a file using OpenCV imwrite function. This function saves images that have integers between 0 and 255, therefore the matrix has to by multiplied by 255, converted to 8 bits unsigned chars and then saved.

Lastly, one important thing to know about OpenCV library is that the images are stored in Blue, Green and Red format instead of the usual Red, Green and Blue format. This is the source of many mistakes made with OpenCV.
Chapter 5

Office room relighting

This chapter describes how to achieve the relighting of an object with the data captured in a regular office room. This includes the data capture itself, the pre-processing of pictures and how to generate a correct approximation of an environment map given the lighting basis of the office room. Finally the results are enhanced using an optimisation process that improves the projection on this lighting basis.

5.1 Challenges of using an office room as a light stage

The capture of the reflectance field of an object is the key element of image-based relighting. Its capture is made easier with a light stage as many parameters are controlled and known. For instance, on a light stage the light sources are all identical. Their solid angle, their intensities and their directions are known parameters. However in an office room every light source can be used from a wide window to a computer screen in order to create as many lighting conditions as possible. As a result the properties of each light source change between the lighting conditions. These have to be derived in order to achieve a correct relighting.

Apart from the light sources themselves, other problems appear when a light stage is not used to capture the reflectance field. First the number of available lighting conditions in an office room is around ten whereas light stage 6 uses 253 lighting conditions. Besides the light directions are uniformly distributed on a sphere for the light stage whereas in an office room this is unlikely to happen. These two facts lead to areas of environment maps not being sampled and produce inaccurate approximations.

Secondly, in a light stage environment the entire room is designed such as indirect light does not modify the reflectance field. However in an office room the walls diffuse light in the entire room and act as secondary light sources. Therefore indirect light has to be taken into account in the relighting process. Finally two lighting conditions might overlap in an office room creating a non orthogonal basis for projection. For example this is the case of indirect light as it contributes in several lighting conditions.

5.2 Presentation of the camera

I captured the reflectance fields of two objects, a bird and an egg, with a Canon EOS 650D camera. Some specific features of the camera were used during the capture. These are described in this section along with the basics concepts of photography needed to understand how these features work.
5.2.1 Settings

When a picture is taken with a camera, light comes through the lens towards the sensor. The sensor is only exposed to light for a fixed amount of time that determine the exposition of the picture. The more the sensor is exposed to light the brighter the picture is. A good looking picture need to be correctly exposed, not to much otherwise the picture is too bright and sufficiently so it is not too dark. Hence taking a nice picture is not an easy task and requires a knowledge of the camera settings. These are described in this section. Their understanding is especially relevant for a graphics related project in which the relative intensity values of the pictures of the reflectance field are important. My main reference for this section is the well written Photography Life website [32].

Shutter speed

The shutter speed corresponds to the speed at which the shutter of the camera closes when a picture is taken. The slower the shutter is, the more the sensor can receive light and the brighter the picture is. However the slower the shutter speed is the more the pictures are sensitive to motion blur and camera shakes which might result in blurry pictures. Hence for the longest exposition settings a tripod is needed to keep the camera steady. The Canon EOS 650D can modify the shutter speed by a third of a f-stop increment. Let $x$ be a real value. A change of $x$ f-stops corresponds to a multiplication of each pixel of the picture by a factor of $2^x$ [12, 18]. Care has to be taken with the values of the shutter speed on the camera. Indeed $\frac{1}{8}$ of a second and $\frac{1}{15}$ of a second correspond to a 1 f-stop difference that is to say to a factor of 2 but $\frac{1}{15}$ is not equal to $\frac{1}{8}$.

Aperture

The aperture corresponds to the size of the diaphragm of the camera that is to say to the size of the hole through which the light comes into the camera. The wider the aperture is, the more the sensor receives light and the brighter the picture is. Besides the aperture gives a control on the depth of field which can be useful for artistic effects. The depth of field corresponds to the distance after which the objects in the background are blurry and not entirely visible. With a high depth of field the background is visible at a high distance. With a low depth of field the background becomes blurry rapidly. A wide aperture decreases the depth of field as shown on figure 5.1. The notation for apertures is $f/x.y$ where $x.y$ is a decimal number. The bigger the decimal number is, the smaller the aperture is. For instance, $f/11$ corresponds to a size of the diaphragm smaller than $f/5.6$ (figures 5.2 and 5.3).

Figure 5.1: Left : Picture of a mirror ball with an aperture of $f/6.6$, a shutter speed of 1/6 and an ISO of 100. The depth of field is low as the window in the background is blurry. Right : Picture of a mirror ball with an aperture of $f/25$, a shutter speed of 1.6 seconds and an ISO of 100. The shutter speed is voluntary slow in order to have a similar exposure in both pictures. The depth of field is high as the window in the background is visible.
5.2. PRESENTATION OF THE CAMERA

ISO

The ISO parameter corresponds to the sensitivity of the sensor of the camera. The bigger the ISO, the more the sensor is sensitive to light and the brighter the resulting picture is. High values of the ISO are usually used for pictures taken in dark environments without flash. However the main drawback is the noise. The higher the ISO, the more noise is visible on the picture. Figure 5.4 compares a high and a low ISO setting.

For pictures taken in low lighting conditions such as dark rooms, three possibilities can be considered to have a good looking picture. First the flash can be used but the light it creates can produce white reflections and artificially change the colors of the scene. Secondly, a high ISO can be used along with a fast shutter speed. This would produce a bright image and motion blur can easily be avoided as a fast shutter speed is used. However noise is visible at high ISO settings. Finally a low ISO setting along with a long exposure time can be used. However this requires a tripod to keep the camera steady during the capture. Indeed the slow shutter speed makes the picture sensitive to motion blur.

5.2.2 Automatic Exposure Bracketing (AEB)

The Automatic Exposure Bracketing is a very useful feature to assemble the HDR images needed for the reflectance field. It allows the user to take three pictures in a row with three different exposures without manually changing the settings of the camera [3]. Hence the risk of moving the camera in between the pictures is minimised as the user does not have to change any parameter. The AEB works as follows. First the middle exposure is chosen and then a relative distance to this exposure is specified. On the Canon EOS 650D, this distance can be up to $+2$ f-stops with a third of a f-stop increment. The exposure change of AEB is produced by a change in the shutter speed setting. The possible combinations of exposures using AEB are given in table 5.1.

Professional cameras does not have the limitation of three pictures with three different exposures for the Automatic Exposure Bracketing. This is useful when scenes have a very high dynamic range. For instance the interior of a room has a million times less light than an outdoor environment on a very sunny day [18]. To fully capture the dynamic range of such a scene the sequence of low dynamic range pictures has to cover many more exposure settings. Hence the number of images in the sequence is higher. However in this project three pictures are enough to cover the dynamic range of the office room.
Figure 5.4: **Top:** Picture taken with an ISO of 100, a 4 seconds shutter speed and an aperture of \( f/6.3 \). A slow shutter speed is required to have a similar exposure on both pictures. **Bottom:** Picture taken with an ISO of 12800, a \( \frac{1}{20} \) seconds shutter speed and an aperture of \( f/6.3 \). The ISO noise is visible.

<table>
<thead>
<tr>
<th>Lowest exposure</th>
<th>Middle exposure</th>
<th>Highest exposure</th>
</tr>
</thead>
<tbody>
<tr>
<td>(-\frac{1}{3})</td>
<td>0</td>
<td>+(\frac{1}{3})</td>
</tr>
<tr>
<td>(-\frac{2}{3})</td>
<td>0</td>
<td>+(\frac{2}{3})</td>
</tr>
<tr>
<td>-1</td>
<td>0</td>
<td>+1</td>
</tr>
<tr>
<td>-(\frac{4}{3})</td>
<td>0</td>
<td>+(\frac{4}{3})</td>
</tr>
<tr>
<td>-(\frac{5}{3})</td>
<td>0</td>
<td>+(\frac{5}{3})</td>
</tr>
<tr>
<td>-2</td>
<td>0</td>
<td>+2</td>
</tr>
</tbody>
</table>

Table 5.1: Possible combinations of exposures settings with AEB mode
5.2.3 Camera modes

The Canon EOS 650D such as most of today’s cameras has five main modes called Intelligent Auto, Program (P), Shutter Priority (Tv), Aperture Priority (Av) and Manual (M). Although the Intelligent Auto is the easiest mode to use in order to produce a good looking picture, it is not the mode adapted to a graphics related project. In this mode the user cannot change the parameters described previously as the camera does it automatically. As a consequence when the light in the office room varies to capture the reflectance field, the Intelligent Auto mode changes the parameters in between the pictures in order to adapt them to the changing light. As a result the relative brightness of a picture compared to another one would not be correct creating a wrong reflectance field.

**Program**

In the program mode, the shutter speed and the aperture are chosen automatically by the camera and the ISO can be set manually. I used this mode for the first data capture and it was a mistake as the shutter speed setting changed between the lighting conditions (the aperture remained constant during the capture). Fortunately this can be corrected with scaling factors for the computation of the relighting (see section 5.4.1).

**Shutter Priority**

In this mode the shutter speed and the ISO can be chosen manually and the camera sets the aperture automatically. The aperture setting is chosen by the camera in a way that the picture is correctly exposed with the current shutter speed setting. This mode is often used to take pictures of moving objects that require a very fast shutter speed [32].

**Aperture Priority**

The aperture priority mode let the user to manually chose the aperture and the ISO. The camera sets the shutter speed automatically to have a correct exposure on the final picture. This mode is often used to have a control over the depth of field [32].

**Manual**

In the manual mode every parameter can be manually set by the user. It is the most difficult mode to use as incorrect settings quickly produce underexposed or overexposed pictures. I used the manual mode for all the data captures except the first one as it is the only way to make sure that the settings do not change between two lighting conditions.

After this overview of the camera I believe that the manual mode combined with the automatic exposure bracketing is at the same time the easiest way and the most accurate way to capture high dynamic range reflectance fields.

5.3 Data capture

This section explains how the capture of the data was done and the important points to remember when capturing data for a graphics related project.

5.3.1 Principle of the capture

In a regular office room the maximum number of lighting conditions that can be used to capture a reflectance field would be ten. The office I used has seven lighting conditions, which is very small to sample a sphere of directions. Hence a smartphone flashlight was used in order to create two more lighting conditions from the sides of the object. This leads to a lighting basis for the relighting made of nine different light sources shown on figure 5.8 and listed here:
• Right window full opened
• Right window half opened (the bottom half of the window is the light source)
• Left window full opened
• Left window half opened (the bottom half of the window is the light source)
• Windows closed and no other light in the office room
• Two front house light sources
• Two back house light sources
• smartphone torchlight on the left
• smartphone torchlight on the right

The capture of the reflectance fields works as follows. An object is sequentially illuminated by the light sources listed above. When one of these conditions is used all the other light sources have to be switched off. For each lighting condition three pictures of the object corresponding to three different exposures are taken with the automatic exposure bracketing mode. These will be used to assemble HDR pictures for the relighting.

As the windows are used as light sources, the light varies depending on the outdoor environment. If a cloud hides the sun between the picture of an object with two different exposures then the light inside the room changes and the pictures have to be discarded as the final HDR picture would not be correct. Hence the pictures with the windows have to be taken quickly.

5.3.2 Installation

A specific setup was used during the data capture in order to get a correct reflectance field. The camera and the object were located on two different tripods in the aim of making sure that they do not move during the capture. Besides, the pictures were taken with a remote to avoid any movements of the camera. A laptop with Canon EOS Utility software [4] installed on it can also be used. This software let the user to remotely control the camera from the laptop and unlike the remote, every parameter on the camera can be adjusted including the shutter speed, aperture and ISO. Hence the software can be used to change the settings of the camera without touching it. Any unwanted movement is therefore avoided. However care has to be taken that the screen of the laptop does not act as a new light source in the office room. Hence when using EOS Utility, the brightness of the screen has to be turned to its minimum and the screen should not be orientated towards the object. I also wore dark clothes during the data capture to avoid any indirect light reflections. Indeed white clothes reflects light and act as a new light source even if it may appear unnoticeable. This problem is also mentionned by Debevec and al. [11] for the reflectance field capture of a human face.

I captured the reflectance fields of two objects an egg (figure 5.5) and a bird (figure 5.6). The egg has a very diffuse texture whereas the bird has specular highlights on it. These objects were chosen to see how the office room method renders diffuse and specular reflections. Besides the shape of the egg is interesting as it is convex and similar to a sphere. Hence it can be used to see if the final relighting is correct. For instance, an environment with a bright light source on the left would relight the left of the egg as its normals are oriented in the opposite direction. Finally, I took pictures of a mirror ball (figure 5.7) in order to capture the state of the office room for each lighting condition.

As the tripods could not be centered due to the furniture in the office room, the objects and the camera were slightly located on the right of the room. Hence the right window corresponds to the light source that illuminates the object the most. As a result this lighting condition was used as a reference to calibrate the exposure setting. For each object the calibration is done such as the
5.3. DATA CAPTURE

brightest areas of the object does not saturate at the darkest exposure and the darkest areas of the object are visible at the highest exposure.

The pictures of the bird and the egg were taken with exposures -2 f-stops, 0 f-stops and +2 f-stops whereas for the mirror ball the exposure settings were -5 f-stops, -3 f-stops and -1 f-stops in the program mode. This difference is due to the fact that the mirror ball is highly reflective. The camera saturates for the mirror ball and a lower exposure setting is required. However in every case the relative exposure change between the pictures remains 2 f-stops.

![Figure 5.5: The egg is a diffuse object](image)

![Figure 5.6: The bird has both a diffuse component and some specular highlights.](image)

5.3.3 Assemble HDR images

The data has to be post processed after the capture. For each lighting condition, the pictures of the objects and the mirror ball are assembled into HDR pictures. I used Canon’s software Digital Photo Professionnal [4] for this purpose. Unfortunately the software cannot save pictures in 32 bits floating point numbers such as the pfm format as it only uses JPEG and 16 bits TIF format. I chose the latter for the project. However I think that 16 bits for each color channel are enough to store the HDR pictures. Indeed the dynamic range of the objects and the mirror ball in the office room is not large enough to require 32 bits.
5.3.4 Create the lighting basis

Each picture of the mirror ball captures the state of the office room for a given lighting condition. The set of mirror balls constitute the lighting basis of the relighting. As a result the mirror ball is cropped in every HDR picture. Then I used the panoramic transformation tool in HDRShop software [15] to transform the mirror ball into a latitude longitude map. Figure 5.8 show these environment maps. They will be needed to find the incoming light direction for each lighting condition.

![Figure 5.8: The 9 lighting conditions displayed in latitude longitude maps. (a) Right window full opened, (b) right window half opened, (c) Left window full opened, (d) Left window half opened, (e) Windows closed and no light sources in the room, (f) Two front house light sources, (g) Two back house light sources, (h) smartphone torchlight on the left, (i) smartphone torchlight on the right](image)
5.4 Data processing

Several image processing computations have to be carried out on the pictures of the reflectance field and the lighting basis before the computation of the relighting. This section describes these steps.

5.4.1 Global scaling factors

For the data capture of the bird and the egg I took the first lighting condition (right window full opened) as a reference. The camera was set up such as the objects had a correct exposure with this lighting condition. However during the data capture I made a mistake as I used the Program mode on the camera. As a result, the shutter speed setting changed for each lighting condition. For instance the pictures of the egg in the dark room were taken with a shutter speed of \( \frac{1}{15} \) seconds whereas in the first lighting condition the shutter speed was \( \frac{1}{15} \) seconds. This means that the picture of the object in the dark room looks as bright as the picture of the object illuminated by the right window full opened. This is obviously not the case in reality meaning that the reflectance field is not correct.

Fortunately this can be easily corrected as the aperture did not change during the capture. The real reflectance field of each object can be retrieved by multiplying each picture by a scaling factor. As explained in [12, 18] the scaling factor for a picture is \( 2^{\text{distance}} \) with distance being the exposure difference in f-stops between the picture of reference (right window full opened) and the current picture. The scaling factors for each object are calculated and presented in tables 5.2, 5.3 and 5.4.

5.4.2 Colour scaling factor

The light sources used in the different lighting conditions have two colors. The illumination created by the smartphone torchlight and the windows is white whereas the house lights are orange (figure 5.8). As a consequence for the house lights the objects diffuse an orange color which modifies their overall appearance and the reflectance field. More specifically this has an impact on the final relighting as the reflectance field was not captured with lights of similar color. The final result would look more orange, especially the specular reflections on the bird. This might not be noticeable in the Grace cathedral as it has many orange light sources but in an environment map that does not have orange colors such as the Uffizi environment map it is noticeable. Hence I used a RGB color scaling factor to solve this problem. It is calculated such as the pictures of the objects illuminated

<table>
<thead>
<tr>
<th>Egg lighting condition</th>
<th>Shutter speed (seconds)</th>
<th>Distance to reference (f-stops)</th>
<th>Scaling factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 (Reference)</td>
<td>( \frac{1}{15} )</td>
<td>0</td>
<td>1.0</td>
</tr>
<tr>
<td>2</td>
<td>( \frac{1}{15} )</td>
<td>0</td>
<td>1.0</td>
</tr>
<tr>
<td>3</td>
<td>( \frac{1}{15} )</td>
<td>(-\frac{1}{3})</td>
<td>(2^{-\frac{1}{3}})</td>
</tr>
<tr>
<td>4</td>
<td>( \frac{1}{5} )</td>
<td>(-\frac{5}{3})</td>
<td>(2^{-\frac{5}{3}})</td>
</tr>
<tr>
<td>5</td>
<td>1.3</td>
<td>(-\frac{13}{3})</td>
<td>(2^{-\frac{13}{3}})</td>
</tr>
<tr>
<td>6</td>
<td>0.5</td>
<td>(-3)</td>
<td>(2^{-3})</td>
</tr>
<tr>
<td>7</td>
<td>0.5</td>
<td>(-3)</td>
<td>(2^{-3})</td>
</tr>
<tr>
<td>8</td>
<td>1.6</td>
<td>(-\frac{14}{3})</td>
<td>(2^{-\frac{14}{3}})</td>
</tr>
<tr>
<td>9</td>
<td>1.6</td>
<td>(-\frac{14}{3})</td>
<td>(2^{-\frac{14}{3}})</td>
</tr>
</tbody>
</table>

Table 5.2: Scaling factors for the pictures of the egg
### Chapter 5. Office Room Relighting

#### Table 5.3: Scaling factors for the pictures of the bird

<table>
<thead>
<tr>
<th>Bird lighting condition</th>
<th>Shutter speed (seconds)</th>
<th>Distance to reference (f-stops)</th>
<th>Scaling factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 (Reference)</td>
<td>1/8</td>
<td>0</td>
<td>1.0</td>
</tr>
<tr>
<td>2</td>
<td>1/8</td>
<td>0</td>
<td>1.0</td>
</tr>
<tr>
<td>3</td>
<td>1/5</td>
<td>-2/3</td>
<td>2^-2/3</td>
</tr>
<tr>
<td>4</td>
<td>0.4</td>
<td>-5/3</td>
<td>2^-5/3</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>-4</td>
<td>2^-4</td>
</tr>
<tr>
<td>6</td>
<td>0.5</td>
<td>-2</td>
<td>2^-2</td>
</tr>
<tr>
<td>7</td>
<td>0.5</td>
<td>-2</td>
<td>2^-2</td>
</tr>
<tr>
<td>8</td>
<td>1.3</td>
<td>-10/3</td>
<td>2^-10/3</td>
</tr>
<tr>
<td>9</td>
<td>1.3</td>
<td>-10/3</td>
<td>2^-10/3</td>
</tr>
</tbody>
</table>

#### Table 5.4: Scaling factors for the pictures of the mirror ball

<table>
<thead>
<tr>
<th>Mirror ball lighting condition</th>
<th>Shutter speed (seconds)</th>
<th>Distance to reference (f-stops)</th>
<th>Scaling factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 (Reference)</td>
<td>1/30</td>
<td>0</td>
<td>1.0</td>
</tr>
<tr>
<td>2</td>
<td>1/50</td>
<td>+2/3</td>
<td>2^2/3</td>
</tr>
<tr>
<td>3</td>
<td>1/25</td>
<td>-1/3</td>
<td>2^-1/3</td>
</tr>
<tr>
<td>4</td>
<td>1/13</td>
<td>-1/3</td>
<td>2^-1/3</td>
</tr>
<tr>
<td>5</td>
<td>0.3</td>
<td>-10/3</td>
<td>2^-10/3</td>
</tr>
<tr>
<td>6</td>
<td>1/15</td>
<td>-1</td>
<td>1/2</td>
</tr>
<tr>
<td>7</td>
<td>1/13</td>
<td>+4/3</td>
<td>2^4/3</td>
</tr>
<tr>
<td>8</td>
<td>1/4</td>
<td>-3</td>
<td>2^-3</td>
</tr>
<tr>
<td>9</td>
<td>0.3</td>
<td>-10/3</td>
<td>2^-10/3</td>
</tr>
</tbody>
</table>
by the house lights have the same overall color than the pictures under other lighting conditions.

I used the wall behind the object to obtain this new scaling factor. It is supposed to be painted with a white diffuse paint meaning that the color of the light source is reflected by the wall. Hence I extracted a small patch of the wall in the picture taken with the house lights and the window (figures 5.9 and 5.10). I took the average RGB color of each patch denoted \((R_{\text{window}}, G_{\text{window}}, B_{\text{window}})\) for the window and \((R_{\text{house lights}}, G_{\text{house lights}}, B_{\text{house lights}})\) for the house lights. Then the color scaling factors are given by equations 5.1, 5.2 and 5.3.

\[
\alpha_{\text{red}} = \frac{R_{\text{window}}}{R_{\text{house lights}}} = 0.58696 \\
\alpha_{\text{green}} = \frac{G_{\text{window}}}{G_{\text{house lights}}} = 0.6471 \\
\alpha_{\text{blue}} = \frac{B_{\text{window}}}{B_{\text{house lights}}} = 0.780822
\]

In order to apply these factors, the pictures of the objects illuminated by the house lights are split into their three color channels. Each channel is multiplied by the corresponding color scaling factor. Then the channels are merged to get the objects illuminated with white house lights. Figure 5.11 show a picture of the reflectance field of the egg with and without the color scaling factor.

![Figure 5.9: Patch of the wall illuminated by the right window full opened.](image)

![Figure 5.10: Patch of the wall illuminated by the house lights](image)

### 5.4.3 Remove overlaps

An overlap in the lighting basis is defined by a superposition of two light sources in two different lighting conditions. For instance, two pictures of an object, one taken with the window full opened and the other with the window half opened have an overlap. Indeed the energy of the light coming from a half of the window is accounted twice. This has two consequences. Firstly the pictures of the reflectance field do not have the same importance, as some of them have contain the same information several times. Secondly, the lighting basis is not orthogonal. As a result some areas of the environment map will be accounted twice when projected onto the lighting basis. This will create an incorrect relit result.

In the lighting basis used the illumination conditions corresponding to the windows full opened and half opened overlap. Hence two overlaps are observed with this lighting basis. In order to remove these, the environment map of the half window is subtracted to the full window. The
resulting environment map corresponds to a non overlapping lighting condition. It has only the top half window and replaces the full window lighting condition. The same operation is done on the pictures of the reflectance field to make them match the new lighting basis.

5.4.4 Remove indirect light

Indirect lighting is not considered in the reflectance field captured with a light stage as the whole environment is controlled. In an office room the white walls diffuse light and therefore contribute to the overall illumination of the room. This effect is called global illumination and is related to the design of the room. Hence it is a constant that appears in every picture of the reflectance field creating overlaps and a non orthogonal lighting basis. Indirect light contribution can be removed with the pictures taken with all the light sources switched off (dark room). Indeed the small amount of light that appears in these pictures corresponds to the indirect lighting. Therefore the picture of the dark room is subtracted to each latitude longitude map of the office room. Besides the same subtraction is done for the pictures of the reflectance field: the picture of an object illuminated in the dark room is subtracted to every other picture of the reflectance field. For these calculations to work, the global scaling factors have to be applied first as the pictures need to have the same reference for the exposure.

5.5 Find the incoming light directions

Unlike the light stage the incoming light directions are not known in advance and have to be derived. For this purpose I used the latitude longitude maps of the lighting basis. Indeed finding the position of a light source on these pictures is equivalent to find the incoming light direction in spherical coordinates. Several methods can be used to find the position of light sources and are explained in this section.

5.5.1 Manual identification of light sources

The simplest way to find the position of light sources is to display each lighting condition as a latitude longitude map, and use the mouse to click on the light sources. The OpenCV library includes functions to detect the events of the mouse. A callback function can be attached to a window that displays a picture. Then it is called each time the user moves the mouse or clicks on one of
5.5. FIND THE INCOMING LIGHT DIRECTIONS

its buttons. With these events, the location of the light sources can be known from the user’s inputs.

Two types of interactions are allowed. For small light sources the user can use the left click of
the mouse to add a point light source to the lighting basis. For a large area light source, the user can
draw a rectangle by clicking and holding the right button of the mouse. The rectangle corresponds
to the bounding box of the area light source. Then the center of the area light source is used as the
direction of the incoming light for the area light source. Hence for each setup of the office room the
user can select the light source and add the corresponding direction to the lighting basis. Then a
Voronoi diagram is generated. Figure 5.12 shows the diagram for a manual selection of light sources.

Figure 5.12: Voronoi diagram generated for a manual selection of light sources.

The main advantage of a manual identification of light sources in the room is that they can be
located accurately. However this is only true for light sources that correspond to direct light such
as the windows, a computer screen or house lights. Indirect light that was not removed is hard to
quantify and to select manually. The user could draw rectangles on bright areas (e.g walls) but this
is not an accurate method.

5.5.2 Automatic identification of light sources using importance sampling

The manual selection of light sources is a problem as the task cannot be automated as it require
inputs from the user. The first idea I had to automate the process was to detect the areas of high
energy in an environment map with a similar algorithm to LightGen plugin for HDRShop [5]. This
type of algorithm is known as importance sampling in computer graphics and is often used to render
computer generated objects in real environments \(^1\).

A common importance sampling algorithm is called inverse cumulative distribution function and
works as follows. First the intensity of each pixel of the environment map is computed and normal-
ized such they sum to one (equation 5.4). The result forms the intensity probability distribution of
the environment map. Then the associated cumulative distribution function is calculated (equation
5.5).

\[
\forall (i,j) \in [1; \text{height}] \times [1; \text{width}] \quad P_I(i,j) = \frac{I_{i,j}}{\sum_{1 \leq i \leq \text{height}} \sum_{1 \leq j \leq \text{width}} I_{i,j} \sin(\theta_{i,j})} \quad (5.4)
\]

\(^1\)This technique is called image-based lighting [7]
CHAPTER 5. OFFICE ROOM RELIGHTING

with:

- \( I_{i,j} = \frac{R_{i,j} + G_{i,j} + B_{i,j}}{3} \) is the intensity of pixel \((i, j)\) that is to say the average of its color channels.
- \( \sin(\theta_{i,j}) \) solid angle of pixel \((i, j)\) of the environment map.
- width is the width of the environment map
- height is the height of the environment map

\[
\forall (x, y) \in [1; \text{height}] \times [1; \text{width}] \quad C_f(x, y) = \sum_{1 \leq i \leq y \atop 1 \leq j \leq x} P_f(i, j) \tag{5.5}
\]

Figure 5.13: Left : A one dimensional probability density function. Right : The corresponding cumulative distribution function. A uniform sampling on the y axis is generated. The non uniform sampling according to the function appears on the x axis.

The second step of the algorithm is to generate \( N \) samples in the range \([0; 1]\) uniformly distributed. Then the inverse image of each sample by the cumulative distribution function is computed. This generates a non uniform sampling according to the cumulative distribution on the \((x, y)\) axis that corresponds to the areas of high energy. Figure 5.14 shows the result of such a sampling in the one dimensional case.

The higher the number of samples the more accurate the approximation becomes. Typically between \(2^{10} = 1024\) and \(2^{14} = 16384\) samples are required for a good approximation. However such a high number of samples cannot be used to create a Voronoi diagram. Hence the idea is to create clusters of samples to approximate a wide area of high energy by only one direction that will be used to compute the Voronoi diagram. A k-means algorithm can find these clusters. Figure 5.14 shows the result of the clustering for 45 light sources in the Grace cathedral.

I applied this algorithm to find the incoming light direction in each lighting condition of the office room. Figure 5.16 and 5.17 show two light selections using the algorithm. For a small light source, the indirect light created by the light source is negligible. Hence the chosen incoming light direction is close to reality (figure 5.16). For the two house lights (figure 5.17) the indirect light produced by the light sources themselves cannot be removed by the subtraction of the dark room. Hence the algorithm does not select the lights properly due to the reflection of light on the walls in the office room. Figure 5.18 shows the Voronoi diagram generated with this algorithm. As several clusters can be computed for one lighting condition, one picture of the reflectance field can be associated to more than one Voronoi cell.
5.5. FIND THE INCOMING LIGHT DIRECTIONS

Figure 5.14: Sampling of the Grace cathedral environment map with the importance sampling algorithm using 8192 samples.

Figure 5.15: Selection of 45 light sources in the Grace Cathedral environment map using importance sampling combined with a k-means algorithm. The selection of light sources is convincing.

Figure 5.16: Light source selection using the importance sampling with 32792 samples combined with a k-means algorithm. The position of the light source is close to reality.
CHAPTER 5. OFFICE ROOM RELIGHTING

Figure 5.17: Light source selection using the importance sampling with 32792 samples combined with a k-means algorithm. The position of the light sources is not correct.

Figure 5.18: Voronoi diagram generated for the office room (combination of the importance sampling with 32768 samples and the k-means algorithm).

5.5.3 Automatic identification of light sources using energy

A third way to choose an incoming light direction for a given state of the office room is to use the centroid of the energy of the office room. The energy of a pixel, $I_p$, also known as the intensity, is defined by the average of its three channel colors. The centroid of an image is computed as follows. First the values of the intensity of the image are normalized to make them sum to one. The normalized energy for a pixel $p$ is given by equation 5.6.

$$ I_p = \frac{I_p}{\sum_{\text{pixel } p \in \text{image}} I_p} \quad (5.6) $$

Then the values of the normalized intensity are summed from left to right and top to bottom starting from the first pixel of the image. When the the sum reaches 0.5 or slightly above, the current pixel is chosen as centroid of the energy and corresponds to the incoming light direction. Figure 5.19 is the Voronoi diagram generated with the centroid of the energy.

5.5.4 Results

Several Voronoi diagrams can be generated depending on the method that selects the incoming light direction. Unlike the light stage, in the office room some lighting conditions can have more than one
5.5. FIND THE INCOMING LIGHT DIRECTIONS

Figure 5.19: Voronoi diagram generated with the centroid of the energy for each lighting condition.

Voronoi cell associated to it. This can happen for both the manual selection of light sources and the importance sampling algorithm but not for the centroid of the energy as it is unique. Indeed the lighting conditions corresponding to the house lights have two light sources instead of one. As a result two light sources can be selected generating two Voronoi cells instead of one. This changes the way the Voronoi cells are integrated. For a lighting condition, I compute the sum of the energy in every Voronoi cell associated to that lighting condition. Equation 5.7 describes the integration of Voronoi cells using the sum of the energy and equation 5.8 corresponds to the Gaussian integration.

For each lighting condition \( L \) \( \omega_{i,channel} = \sum_{\text{Cells } C \subseteq L} \sum_{\text{pixel } p \in C} p_{\text{channel}} \sin(\theta_p) \) (5.7)

For each lighting condition \( L \) \( \omega_{i,channel} = \sum_{\text{Cells } C \subseteq L} \sum_{\text{pixel } p \in C} p_{\text{channel}} \sin(\theta_p) G(\mu_C, \sigma, x, y) \) (5.8)

with :

- \( p = (p_x, p_y) \) are the pixels coordinates in the environment map.
- \( p_{\text{channel}} \) is the value of the current pixel for one of its three color channels
- \( \sin(\theta_p) \) is the solid angle for pixel \( p \)
- \( G(\mu_C, \sigma, x, y) = \frac{1}{2\pi\sigma_x\sigma_y} e^{-\frac{(x-\mu_x)^2}{2\sigma_x^2} - \frac{(y-\mu_y)^2}{2\sigma_y^2}} \) is a two dimensional Gaussian with mean \( \mu_C = (\mu_x, \mu_y) \) and standard deviation \( \sigma = (\sigma_x, \sigma_y) \).
- \( C \) is the set that contains the pixels of the Voronoi cell \( C \).

Figures 5.12, 5.18, 5.19 show the Voronoi diagrams obtained with the three methods described earlier. Most of the cells are big compared to the light stage as few lighting conditions are used. Large cells are a problem as the energy that is far from the lighting direction is taken into account and associated to this lighting direction during the integration. The renderings obtained with these diagram are presented on figures 5.20 and 5.21. The results are very diffuse and the specular reflections are not visible on the bird. I think the results obtained with the centroid of the energy are the best results as they look more colourful than the other ones.

\(^2\)The energy of the house lights compared to the right window full opened (reference) is already normalised with the colour scaling factor.
Figure 5.20: Egg relit in the Grace cathedral. **Left:** Manual selection of light sources. **Middle:** Importance sampling combined with k-means. **Right:** Centroid of energy

Figure 5.21: Bird relit in the Grace cathedral. **Top:** Manual selection of light sources. **Middle:** Importance sampling combined with k-means. **Bottom:** Centroid of energy
5.6 Use masks for each light source

5.6.1 Method

The Voronoi diagram is not adapted to the office room relighting as it creates big cells for a few lighting conditions. Hence I used another method that describes the light sources of the office room with masks. Each mask has two values 0 (black) for the light source and 1 (white) otherwise. Then the formulas to integrate each mask are given in equation 5.9.

\[
\omega_{i,\text{channel}} = \sum_{\text{pixels } p \in M} p_{\text{channel}} \sin(\theta_p)
\]  

(5.9)

It is closer to the truth as for a given lighting condition the size of the mask depends on the size of the light source. Hence the solid angle of the light source is taken into account with the mask. Besides by changing the size of a mask it is possible to change the importance of the corresponding light compared to another one. For instance in the reflectance field the lighting conditions associated to the windows tend to dominate as the they create a wide and bright illumination. Therefore if the masks associated to the windows are smaller than the masks associated to the smartphone LED illumination then the importance of the windows in the relighting can be relatively lowered compared to the LED.

The masks can be drawn with a software such as Paint but care has to be taken that they do not overlap. If an overlap is created, the lighting basis is not orthogonal and this area will account the energy of the environment map twice creating a wrong relighting. In particular the mask corresponding to the dark room has to be accounted differently. Indeed it is very large as the illumination of the dark room comes from a wide area around the windows. As a result it creates an overlap with the masks of the windows. To avoid such a problem I computed a residual mask which corresponds to the mask of the dark room in which all the other masks are removed.

As the masks are binary images, the residual mask can be computed using a logic formula. The algorithm works as follows. First the masks of all the light sources (dark room excluded) are gathered on a single image. Then I wrote the truth table 5.10 in order to derive the logic formula 5.5. The table is easy to understand. For instance the second line says if there is an overlap between a light source and the mask of the dark room then the residual mask is white meaning that the energy of that area will not be accounted for the dark room (it is already accounted for the light source). Figure 5.22 is a diagram that summarises the computation of the residual mask.

\[
R = (\overline{A} \land \overline{B}) \lor (A \land \overline{B}) \lor (A \land B)
\]  

(5.10)

with :

- \(A\) the binary image corresponding to all the masks gathered.
- \(\overline{A}\) the binary image corresponding to the negation of \(A\).
- \(B\) is the binary image corresponding to the mask of the dark room.

<table>
<thead>
<tr>
<th>All masks (dark room excluded)</th>
<th>mask of the dark room</th>
<th>residual mask</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 (black)</td>
<td>0 (black)</td>
<td>1 (white)</td>
</tr>
<tr>
<td>1 (black)</td>
<td>0 (black)</td>
<td>1 (white)</td>
</tr>
<tr>
<td>0 (black)</td>
<td>1 (white)</td>
<td>0 (black)</td>
</tr>
<tr>
<td>1 (black)</td>
<td>1 (white)</td>
<td>1 (white)</td>
</tr>
<tr>
<td>1 (white)</td>
<td>1 (white)</td>
<td>1 (white)</td>
</tr>
</tbody>
</table>

Table 5.5: Truth table to derive the logic formula that calculates the residual mask.
• \( \overline{B} \) the binary image corresponding to the negation of \( B \).

For every lighting condition (except the dark room):
create a mask on the light source

For the dark room: create a mask for the indirect illumination

1

1'

2

2'

Masks for all the light sources
(direct illumination)

Finally residual mask (indirect light)
Indirect illumination mask in which the masks of other light sources are removed

Figure 5.22: Diagram for the computation of the residual mask. The masks are painted in red for clarity but should be black. The final residual mask is associated to the pictures taken in the dark room.

Besides small masks have to be avoided as their corresponding weight change quickly for small rotations of the environment map (especially in the case of high frequency lighting). Indeed small masks have fewer pixels. Consequently even if the value of few pixels change with the rotation, it might produce big changes in the final weight as HDR environment maps are used.
5.6.2 Results

The last steps of the relighting are unchanged compared to the light stage relighting. The weights have to be normalized in the same manner and the relit result is computed with a linear combination of the pictures of the reflectance field. In order to generate the background of the objects, I created a mask for the bird and the egg with Photoshop CS2 [1]. These masks are presented on figures 5.23 and 5.24. The final result is saved in a 16 bits PNG file to preserve the dynamic range of the images. Besides no gamma is applied on the final result as HDR pictures are used for the reflectance field.

Figure 5.23: Mask of the bird.  
Figure 5.24: Mask of the egg.

Figures 5.25 and 5.26 show the results of the relighting of the bird and the egg without optimisation. They are closer to the results obtained with the centroid of the energy. However the specular highlights on the bird are still not visible. Even if the final results are not necessarily better than the previous ones, they are closer to the truth as the areas of the environment map that do not appear in the lighting basis are not taken into account.
CHAPTER 5. OFFICE ROOM RELIGHTING

Figure 5.25: **Top**: Bird relit in the Grace cathedral without optimisation. **Bottom**: Bird relit in the Pisa courtyard without optimisation.

Figure 5.26: **Left**: Egg relit in the Grace cathedral without optimisation. **Right**: Egg relit in the Pisa courtyard without optimisation.
5.7 Optimisation of the projection on the lighting basis

The weights used for the relighting correspond to a piecewise constant basis that approximates the environment map. Qualitatively, the optimisation process of Tunwattanapong, Ghosh and Debevec [43] gives more or less importance to each light source in order to improve the approximation of the environment map and hence ameliorate the final result. Mathematically, each weight is multiplied by a coefficient so that the new weights lead to a better approximation of the environment map. The optimisation corresponds to the minimisation of a given function in the original space or the principal components space of the projection of the environment map on the lighting basis. I used the C++ optimisation library Dlib [28] for this purpose.

5.7.1 Optimisation in original space

Principle

The optimisation process can be computed in the original space of the projection of the environment map on the lighting basis. According to Tunwattanapong et al. [43] it corresponds to the minimisation of function 5.11. Equation 5.12 is the same function but expanded and adapted to the case of masks and Voronoi cells.

\[ \min_x f(x), f(x) = \|Ax - y\|_2 \] (5.11)

with:

- \( y \) is the column vector of the environment map
- \( x \) is the column vector of weights
- \( A \) is the matrix that projects the environment map onto the lighting basis.

The optimisation is done in intensity space taking into account the solid angle.

\[
\min_{(\omega_1, \ldots, \omega_9)} f(\omega_1, \ldots, \omega_9) = \min_{(\omega_1, \ldots, \omega_9)} \sqrt{\sum_{i=1}^{\text{height}} \sum_{j=1}^{\text{width}} (\alpha_{k_{i,j}} \omega_{k_{i,j}} - I_{i,j} \sin(\theta_{i,j}))^2} \] (5.12)

with:

- \( \alpha_n \) is the intensity of the weights of cell \( n \) that is to say the average of the color that correspond to cell \( n \) (after integration).
- \( \omega_n \) is the variable of cell \( n \). The goal is to optimise the value of these nine variables (one for each lighting condition).
- \( I_{i,j} \) is the intensity of pixel \((i, j)\) of the environment map.
- \( k_{i,j} \) is the number of the cell in which pixel \((i, j)\) is located.
- \( \sin(\theta_{i,j}) \) is the solid angle of pixel \((i, j)\)
- \( \text{width} \) is the width of the environment map
- \( \text{height} \) is the height of the environment map

Equation 5.12, can be computed with the following algorithm written in pseudo code.
I used a constrained optimisation to minimise this function. Indeed the new weights have to be positive as negative pixels in the final result would not make sense. Regarding the upper bound for the constraints I first ran the optimisation process with a high value for the upper bound (e.g 10000). Then I reduced the upper bound to 10 as the optimal variables have the same order of magnitude (around $10^{-5}$) for the environment maps considered.
5.7. OPTIMISATION OF THE PROJECTION ON THE LIGHTING BASIS

Implementation details

I used Dlib function find_min_box_constrained to compute the optimisation. One of the parameters of this function is the address (or function pointer) of the function to be optimised. Let \( f_{opt} \) denote this function. \( f_{opt} \) depends on the lighting conditions and consequently on the masks. Hence it needs to share variables with the instance of the object that computes the office room relighting. For instance the weight of each mask after integration and the name of the environment map used are such variables. However no parameters can be sent to \( f_{opt} \) through Dlib function find_min_box_constrained. The only way I found to solve this problem was to use global variables for these parameters. I tried to limit their number as the usage of global variables is not recommended in C++ programming.

5.7.2 Optimisation in principal component space

The same optimisation can also be computed in the principal component analysis space. It requires to compute the principal component space of the matrix that projects the environment map onto the lighting basis of the office room (denoted \( A \) in equation 5.11). It can be computed with the following code.

```cpp
//Computes the projection matrix
//weights is an array that contains the weights of the lighting basis.
//weights[i] contains three values (R,G,B) that is the weight of mask i.

void computeProjectionMatrix()
{
  float weightsIntensity[9];
  for each lighting condition 0<k<9
    weightsIntensity[k] is the average of the three values of weights[k];
    Load the masks that defines the light source (For the dark room load the residual mask);
    Concatenate every row of the matrix such as the image is represented by one row vector;
    Transpose the result into a column vector;
    Multiply the vector by weightIntensity[k];
    This column vector constitutes column k of the matrix A;
}
```

At the end of this procedure, each column of matrix \( A \) corresponds to a vector that describes one lighting condition. Hence \( A \) is the matrix that projects an environment map onto the lighting basis of the office room. Then I used OpenCV to compute the principal component space of the covariance matrix of \( A \). OpenCV has a PCA object that has a method called project. It projects the vector given as a parameter into the PCA space by multiplying it by the transpose of the matrix of change of basis from the original basis to the principal components basis.

The following pseudo code explains how the function that will be optimised in PCA space is computed. This function can be minimised using find_min_box_constrained of the Dlib library.
5.7.3 Results and discussion

Comparisons between the optimised results in original space and in PCA space are shown on figures 5.27 and 5.28. The best results are obtained for the optimisation in the original space. Indeed the egg has a nice diffuse colour and the specular highlights are visible on the bird. The PCA space optimisation is still better than the result without optimisation but not as good the original space optimisation. Besides for small rotations of the environment map (e.g 1 degree), the results of the PCA space optimisation change drastically (figure 5.29). This does not happen for the original space optimisation. One explanation would be the few number of lighting conditions. Indeed the approximation of the environment map done by Tunwattanapong et al. [43] uses twice the number of lighting conditions I have. Besides they use spherical harmonics and local light to both approximate low and high frequencies of the environment map. Without these, the PCA approximation does not give good results.

Influence of the size of the masks

Non overlapping masks are the best way to define an orthogonal basis when few lighting conditions are used. The size of each mask depends on the solid angle of its associated light source. As the windows have the biggest solid angle they have an important contribution in the final relighting. If their masks are too big, they can dominate and lead to a wrong relighting. As a result the masks of the windows have to be smaller than the masks of other lighting conditions. Figure 5.30 shows an example of such a lighting basis. However this type of basis is adapted to environment maps with low frequency lighting such as Pisa courtyard but not for environment maps in which lights vary rapidly. Indeed when a high frequency environment map is rotated, the weight of small masks vary rapidly as they contain few pixels and the relighting vary too quickly. This does not happen for environment maps that contain low frequency lighting as the all pixels have similar brightness and colours. Therefore I use two sets of masks, one for environment maps with high frequency lighting and one for environment maps with low frequency lighting. These are displayed on figures 5.30 and

```cpp
float functionToOptimisePCA(column_vector variables)
{
    Load the HDR environment map.;
    for each pixel of the environment map.
    {
        Compute the intensity of the pixel multiplied by its solid angle;
        Save the result in a matrix denoted IntensityEM;
    }
    Concatenate every row of IntensityEM matrix such as the image is represented by one row vector;
    Transpose the result into a column vector y;
    Project y into the PCA space and save the result in a column vector yPCA;
    Compute the matrix vector multiplication A*variables and projects the results into the PCA space;
    Save the result in a column vector xPCA;
    return Norm2(xPCA-yPCA); //Euclidean norm
}
```
5.7. OPTIMISATION OF THE PROJECTION ON THE LIGHTING BASIS

Figure 5.27: Pictures of the bird relit in the Grace cathedral. **Top:** no optimisation process. **Middle:** optimisation in original space. **Bottom:** optimisation in PCA space. The optimisation gives better results for the specular highlights.

Figure 5.28: Pictures of the bird relit in the Grace cathedral. **Left:** no optimisation process, **Middle:** optimisation in original space, **Right:** optimisation in PCA space
Figure 5.29: Relighting of the bird in the Grace cathedral with a 2 degrees rotation and with the PCA optimisation. The diffuse colour of the bird has entirely changed compared to figure 5.27. In the original space the relighting remains the one shown in figure 5.27 for such a small rotation.

5.31.

Figure 5.30: Lighting basis adapted for environment maps with low frequency lighting. The masks corresponding to the windows are smaller. The residual mask is not displayed.

Figure 5.31: Lighting basis adapted for environment maps with high frequency lighting. The masks corresponding to the windows are bigger. The residual mask is not displayed.

As a summary small lobes for the windows are more adapted to environment maps with low frequency lighting to avoid their domination in the final result. However bigger masks have to be used for environment maps with high frequency lighting in order to get better results and stabilise the value of final weights when using rotations. Figures 5.32 and 5.33 present comparisons of the results obtained with the two types of lighting bases. More results are presented in appendix C.
5.7. OPTIMISATION OF THE PROJECTION ON THE LIGHTING BASIS

Figure 5.32: Comparison of the relighting of the egg in an environment map with high frequency lighting (Grace cathedral with optimisation). **Left:** basis adapted to low frequencies. **Right:** basis adapted to high frequencies.

Figure 5.33: Comparison of the relighting of the egg in an environment map with low frequency lighting (Pisa courtyard with optimisation). **Left:** basis adapted to low frequencies. **Right:** basis adapted to high frequencies. The windows dominate in the picture on the right due to the size of the masks used for these lighting conditions.
Chapter 6

Results and comparisons

The results obtained with the office room relighting can be hardly evaluated without a comparison with a known technique. This chapter presents two comparisons one with the free-form light stage [33] and another with the light stage relighting. In the first method I captured the reflectance field of the egg under a high number of illumination conditions and produced the relighting. The second method uses the objects of the light stage relighting with the office room method.

6.1 The free-form light stage

6.1.1 Data capture

In the free-form light stage technique I captured the reflectance field of the egg with the smartphone torchlight as a hand-held light source. The lighting basis contains 142 illumination conditions taken in a whole sphere of direction. The pictures were only taken with a low dynamic range due to the high number of lighting conditions. Indeed assembling HDR pictures requires time as three pictures are taken for each lighting condition instead of one. Besides I used the manual mode of the camera during the capture to avoid a change in the settings between each lighting condition. Consequently no scaling factors are needed to compute the final relighting.

Unfortunately the egg could not be illuminated from certain directions as we did the capture in an office room. Indeed the furniture could not be moved and blocked many directions. Besides the mirror ball was located on the table therefore the object could not be illuminated from the bottom as the light source could not be seen in the mirror ball. Figure 6.1 shows one of the picture taken during the data capture.

After the capture I cropped the mirror ball and the egg in the 142 pictures. Moreover each mirror ball is transformed in a latitude longitude map (figure 6.2) that corresponds to a lighting condition for the free-form light stage.
6.1. THE FREE-FORM LIGHT STAGE

Figure 6.1: One picture of the reflectance field of the Egg with the free-form light stage method. The mirror ball is used to derive the incoming light direction.

Figure 6.2: Latitude longitude map of a lighting condition for the free-form light stage method


6.1.2 Generate the Results

As seen on figure 6.2, the torchlight creates a widespread illuminated area on the mirror ball due to the reflection of light on the shiny ball. Hence choosing the centroid of the energy as an incoming light direction is not adapted as the widespread area shifts the center of the energy. The incoming light direction is chosen with a manual selection of light sources in order to have a more accurate result. However this require time as the user has to manually select a light source in 142 different lighting conditions.

Unlike the office room relighting I have enough lighting conditions to generate a correct sampling of the sphere of directions. Figure 6.3 shows the corresponding Voronoi diagram. It has big cells at the bottom as the object could not be illuminated from these directions. This results in a poor result in this area. The cells are integrated in the same manner as the light stage relighting. A Gaussian modulation can also be taken into account in the integration. Figure 6.4 shows the weights of each cell in the Voronoi diagram.

![Voronoi diagram for the free-form light stage](image)

Figure 6.3: Voronoi diagram for the free-form light stage

![Weights for the Voronoi diagram using a point integration](image)

Figure 6.4: Weights for the Voronoi diagram using a point integration

Finally, a gamma correction has to be applied on the final result as I used low dynamic range images. Without it the relighting cannot be compared to the high dynamic range pictures of the office room relighting.
6.1.3 Results and comparisons

This section presents the results of the relighting with the free-form light stage method, considered as the ground truth, and compare them to the office room relighting.

Point and Gaussian integration for the free-form light stage

Figure 6.5 is a comparison between the Gaussian integration and the point integration for the free-form light stage. The point integration is more colourful than the Gaussian one. However the light coming from the top of the egg is more visible with the Gaussian integration.

Figure 6.5: (a) Relighting in the Grace cathedral with the free-form light stage method and a point integration. (b) Relighting in the Grace cathedral with the free-form light stage method and a Gaussian integration (variance 300 in both dimensions)

Comparison with the office room relighting

For a correct comparison the results presented here only use the point integration for the free-form light stage. Indeed, the weights computed with the masks for the office room relighting are not modulated by a Gaussian. All the results for the office room relighting are optimised in the original space as it gives the best results.

Figures 6.6, 6.7, 6.8 and 6.9 present three comparisons between the office room relighting and the free-form light stage. For a diffuse object and an environment map with low frequency lighting such as the Uffizi gallery and the Pisa courtyard, the office room relighting achieves good results. The diffuse colour of the egg is similar in both cases. For an environment map with high frequency lighting such as the Grace cathedral, the diffuse colour is also correct. However the office room method does not render directional illumination such as the orange area on the left of the egg. It is slightly visible in the relighting of the egg with the free-form light stage.
Figure 6.6: Relighting in the Grace cathedral. **Left:** Office room relighting. **Right:** Ground truth.

Figure 6.7: Relighting in the Uffizi Gallery. **Left:** Office room relighting. **Right:** Ground truth.
6.1. THE FREE-FORM LIGHT STAGE

Figure 6.8: Relighting in the Pisa courtyard. **Left**: Office room relighting. **Right**: Ground truth.

Figure 6.9: Relighting in the Eucalyptus Grove. **Left**: Office room relighting. **Right**: Ground truth.
6.2 Comparison with the light stage relighting

6.2.1 Principle

The results of the light stage relighting can be hardly compared with the office room method as I cannot capture the reflectance field of the helmet and the plant in the office room. However the reflectance field can be generated using the environment maps that forms the lighting basis of the office room. Indeed as a first step I can relit in the office room the helmet and the plant with the light stage relighting method. At the end of this step I get 9 images for each object. They correspond to their simulated office room reflectance field. Then as a second step I can use this new reflectance field in order to simulate the result of the office room method on the plant and the helmet. Figure 6.10 is a diagram that summarizes this process.

The reflectance field pictures of the helmet and the plant are low dynamic range pictures. However as each environment map is a high-dynamic range picture, the results of their relighting in the office room can be saved as HDR pictures. This step is required as the office room method uses a HDR reflectance field.

6.2.2 Results

Figures 6.11 and 6.12 compare the relighting of the helmet for the light stage relighting and the simulated office room relighting. As expected the diffuse colour is close to the truth but the specular reflections are not correctly rendered. Again this is due to the few lighting conditions used for the office room relighting.

Figures 6.13 and 6.14 compare the relighting of the plant for the same methods. The bidirectional reflectance distribution function (BRDF) of the plant has a high diffuse component and a low specular component. Hence it is closer to the objects used for the office room relighting that is why it is a better object to make comparisons. The colour of the final result is close to the light stage relighting. Again the directional illumination (orange area on the left of the Grace cathedral) is not correctly rendered due to the poor sampling of the office room relighting.
Figure 6.10: Diagram that presents how the objects of the light stage relighting can be relit with the office room method.
CHAPTER 6. RESULTS AND COMPARISONS

Figure 6.11: Relighting of the helmet in the Grace cathedral. **Left:** Light stage relighting. **Right:** Simulated office room relighting with original space optimisation.

Figure 6.12: Relighting of the helmet in the Pisa courtyard. **Left:** Light stage relighting. **Right:** Simulated office room relighting with original space optimisation.

Figure 6.13: Relighting of the plant in the Grace cathedral. **Left:** Light stage relighting. **Right:** Simulated office room relighting with original space optimisation.
Figure 6.14: Relighting of the plant in the Pisa courtyard. **Left:** Light stage relighting. **Right:** Simulated office room relighting with original space optimisation.
Chapter 7

Conclusion and future work

Today, the production of blockbusters takes time and is very expensive. In particular the cost of visual effects is very high and only few studios in the world can afford the technology to make them. Weta Digital (The Hobbit)[14], Double Negative (Harry Potter)[34] and Industrial Light & Magic (Star Wars)[31] are examples of such studios. This project proves that image-based relighting can be achieved at a cheap price with the illumination conditions available in a regular office room. This technique only requires two tripods, a camera and a computer. Therefore I believe that it can be used by small visual effects studios to relight objects at a very low cost. Obviously, the results are not as good as the light stage relighting but this was expected because of the few lighting conditions available.

The comparisons with the free-form light stage technique showed that the office room relighting gives good results for diffuse objects. Specular reflections and directional illumination are not correctly rendered. Indeed the specular component of an object depends on both the point of view and the incoming light direction and it cannot be fully captured with only a few lighting conditions. However this project demonstrates that if the projection of an environment map on a sparse lighting basis is optimised, then the rendering of the specular component can be improved.

Concerning the relighting process, the office room method does not use the usual tessellation of environment maps due to the few illumination conditions available. Indeed Voronoi diagrams are not adapted to sparse lighting bases as they tend to cover wide areas of environment maps leading to incorrect approximations. Instead I showed that masks that depend on the solid angle of each light source give better results with a sparse lighting basis.

In the original paper that presents the light stage [11], Debevec, Hawkins, Tchou, Duiker, Sarokin and Sagar also present a method to render a human face from another viewpoint using the data captured with the light stage. As a future work, it would be interesting to see if the office room method can be extended to render the egg or the bird from another viewpoint. On one hand it should be easier as subsurface scattering is negligible for these objects. Hence no model of the skin is required and the specular and diffuse component can be rendered more easily. On the other hand the rendering can be harder as the office room method has many constraints that a light stage does not have. For instance, the illumination that comes through the windows is time dependent as it relies on an uncontrolled outdoor environment. As a result, it would be difficult to take pictures of an object from several viewpoint without a change in the illumination. Again comparisons with both the free-form light stage and the simulated office room relighting on the light stage objects would be possible to evaluate the results.
Bibliography


Appendices
Appendix A

Program for image-based relighting

The main goal of the project is to implement a novel method that uses a regular office room to relight objects under any illumination condition. However, the light stage relighting and the free-form method were useful to either understand how image-based relighting works or to compare my results with a known technique. Therefore I implemented an easy to use graphical user interface that gather the three relighting methods used in this project. The results presented in this report can be easily reproduced with this program. This chapter describes how to use the software and how to compile it. A section also describes the documentation for developers who want to implement a new type of relighting.

A.1 Presentation

The software is written in C++ with the libraries OpenCV 2.4.2 [25], Qt 5.0.1 [13] and Dlib 18.9 [28]. OpenCV is a computer vision library that is available for several programming languages such as C++ and Python. I used it for everything that is related to images processing including the calculation of linear combinations of images, the computation of the Voronoi diagram and the plot of diagrams. I also used it to display images of the office room in order to manually select the light sources. This could have been done with the Qt library.

Qt is a cross-platform library to create graphical user interfaces (GUI). This means that the code written for Linux also compiles on MacOS or Windows as shown on figures A.1 and A.2. The operating system displays a graphical interface that is adapted to its environment. As I work on both MacOS and Linux this is a very interesting feature and is the main reason I chose Qt.
Dlib is a C++ optimisation library that is easy to use and install. Indeed it is only necessary to give the path of the library in order to compile a program that uses it. The library can compute many optimisation processes. I used it to find the minimum of non linear functions.

A.2 How to use the program

The software I designed in this project can produce three different types of relighting: the light stage relighting, the free-form light stage relighting and the office room relighting. This section describes the settings of each relighting and how a user can produce renderings easily.

A.2.1 Light stage relighting

Figure A.3 presents the tab that corresponds to the interface for the light stage relighting. The user can select a few parameters. The first one is the object that will be relit. Two objects are available: the helmet and the plant. The second parameter is the environment map in which the object will be relit. Five environment maps are available: the Grace cathedral, the Uffizi Gallery, the Pisa courtyard, the Eucalyptys Grove and St. Peter's Basilica. The third parameter is the type of integration of the Voronoi cells. As explained in section 4.2.3 the pixels in a given cell can be modulated by a two dimensional Gaussian. This parameter allows the user to choose the Gaussian modulation or not. The last parameter corresponds to the number of rotations of the environment maps. If N offsets are chosen then the object will be relit under these N rotations of the environment map. The angle between two rotations is \( \frac{2\pi}{N} \). The number of rotations can be a number up to 360 that corresponds to a rotation of one degree for each frame generated.

When the user presses the "Start" button, the computation begins and a new window appears (figure A.4). It displays the progress of the relighting. Each time a step is finished it is written in the text area on the right of the window. When the whole computation is over the final result is displayed on the left of the window. A bar on the top right of the window shows the progress of the calculation.

![Figure A.3: Graphical user interface for the light stage relighting.](image-url)
A.2. HOW TO USE THE PROGRAM

Figure A.4: Window that displays the progress of the relighting.

A.2.2 Free-form light stage relighting

The graphical interface for the free-form light stage relighting is presented on figure A.5. The parameters environment map, integration method and number of offsets are explained in the previous section (light stage relighting). The number of lighting conditions is the number of images in the reflectance field of the object. In this project we took 142 pictures of the egg illuminated by a hand-held LED light source. The parameter exposure modifies the exposure of the final result to avoid dark or bright renderings. Its value is in f-stops. A +2.25 f-stops usually gives good results. This does not change the exposure of the background.

The free-form light stage relighting only uses the manual selection of light sources. However it is a long process as I have 142 lighting conditions. Besides if an error is made in the middle of the manual selection, the user has to redo the 142 selections. Therefore I implemented a feature that saves the manual selection and the corresponding Voronoi diagram in two text files. In order to do that the user can click on manual selection and then tick the checkbox "Save Voronoi diagram". Then he has to manually select the 142 light sources (see section A.2.3 for an explanation on the manual selection). Once it is done, the selection is saved. Later if he wants to do a new relighting with the free form light stage he can click on the button "Load from file" to reuse the Voronoi diagram he generated previously.

A.2.3 Office room relighting

The office room relighting tab has many parameters. Some of them including the object, the environment map, the integration method and the number of offsets are explained in the light stage relighting section (A.2.1). Others such as the exposure change are described in the free-form relighting section (A.2.2). The number of lighting conditions corresponds to the number of pictures in the reflectance field of the objects. I used 9 lighting conditions in this project. The user can choose a method to select the incoming light direction in each lighting condition. InverseCDF, Median Energy and Masks are three automatic methods that does not require inputs from the user unlike the manual selection.

InverseCDF corresponds to the importance sampling algorithm combined with the k-means algorithm described in section 5.5.2. In this case the number of samples in the importance sampling algorithm can be chosen below the light identification box. Median energy corresponds to the al-
algorithm that uses the centroid of the energy as the incoming light direction (section 5.5.3). Masks is the method that uses masks to define each light source instead of the Voronoi diagram (section 5.6).

When the user chooses the manual selection, the program displays each lighting condition. Then he has two possibilities. The first one is to use the left click of the mouse to draw a point on the picture (figure A.7). This point will be used as the light direction for the current lighting condition. The second possibility is to use the right click of the mouse to draw a rectangle on a light source (figure A.8). This is particularly useful for an area light source (e.g a window). The center of the rectangle is then used as the incoming light direction in the Voronoi diagram. It is also possible to draw several points and rectangle for one lighting condition. For instance the lighting condition that corresponds the house lights has two incoming light directions (the two spots of lights.) In order to confirm the manual selection of a lighting condition any key can be pressed.

If the weights are computed with masks, two sets of masks can be chosen. "High frequency lighting" is a set of masks that is more adapted to environment maps that have light sources that vary quickly (e.g the Grace cathedral) whereas "Low frequency lighting" corresponds to masks that are more adapted to environment maps whose light sources vary slowly (e.g the Uffizi gallery). This is explained in section 5.7.3

Finally the optimisation can be enabled either in the original space or in the principal component space (see section 5.7 for more details). It only works with the selection of light sources with masks. The optimisation gives better results but the computation is longer. For instance in the original space it can take up to 15 minutes to calculate the final result depending on the environment map, the set of masks used and the number of offsets. The PCA space optimisation is quicker and requires less than a minute to compute but its results are not as good as the original space optimisation.
A.3. **HOW TO COMPILE ?**

It took me time to understand how to compile with Qt 5.0.1, OpenCV 2.4.2 and Dlib 18.9. To avoid a loss of time in the future, this section explains how to do it. I wrote the code on Qt Creator an easy to use Integrated Development Environment (IDE) that comes with Qt library installed. In order to compile with other libraries than Qt a few lines have to be added in the .pro file of the project. On the latest version of MacOSX (Mavericks 10.9) the line "QMAKE_MACOSX_DEPLOYMENT_TARGET 10.9" is compulsory. Without it the program cannot compile. The following code is a template of the .pro file.
A.4 How to install the program?

Once compiled the program needs few folders and files to work. Four folders called environment_maps, lighting_conditions, images and Results have to be created in the working directory. The folder environment_maps contains the environment maps used to relit the objects in both pfm and ppm format (for display). Each environment map must have a size of $1024 \times 512$ and a specific name. Table A.1 gives the file name of each environment map.

The folder lighting_conditions has two sub-folders named free_form and office_room. These contain the latitude longitude maps of each lighting condition for the free-form light stage and the office room relighting. Besides the office_room sub folder has itself two sub directories called high_freq and low_freq that contain the two sets of masks adapted to the environment maps that have high or low frequency lighting.

The folder images has three sub-folders named free_form, office_room and light_stage each of these contains the pictures of the reflectance field of the objects and the corresponding masks. For the light stage the pictures can be downloaded on Paul Debevec website [10] and have to be
Table A.1: Name of the files of the environment maps

<table>
<thead>
<tr>
<th>Environment map</th>
<th>File name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grace cathedral</td>
<td>grace_latlong</td>
</tr>
<tr>
<td>Pisa courtyard</td>
<td>pisa_courtyard1024</td>
</tr>
<tr>
<td>St Peter’s Basilica</td>
<td>stpeters_probe</td>
</tr>
<tr>
<td>Eucalyptus grove</td>
<td>eucalyptus_grove</td>
</tr>
<tr>
<td>Uffizi gallery</td>
<td>uffizi</td>
</tr>
</tbody>
</table>

unzipped in the corresponding directory. The masks must have the names Helmet_mask.png and Plant_mask.png.

Finally a directory named Results with three sub folders named free_form, office_room and light_stage have to be created. These are empty but will contain the results of the renderings.

A.5 Documentation

Figure B.6 shows the documentation that I wrote for the program using Doxygen [44]. It is provided in the archive with the source code of the project (index.html file in the documentation directory). It describes every function, class, methods and attributes that is used in this project. It is particularly useful for developers who want to write their own relighting algorithm. Indeed it can be easily done by creating a new class that inherits the Relighting class. Then the methods relighting and rayTraceBackground have to be implemented using the documentation of the LightStageRelighting, FreeFormRelighting and OfficeRoomRelighting classes.

Figure A.9: A picture of the documentation in the HTML format.
Appendix B

Light stage results

This appendix presents several light stage renderings of the plant and the helmet.

Figure B.1: Relighting of the helmet in the eucalyptus grove using a point integration. **Left:** Original orientation of the environment map. **Left:** 180 degrees rotation of the environment map.

Figure B.2: Relighting of the plant in the eucalyptus grove using a point integration. **Left:** Original orientation of the environment map. **Left:** 180 degrees rotation of the environment map.
Figure B.3: Relighting of the helmet in St Peters Basilica using a point integration. **Left:** Original orientation of the environment map. **Left:** 270 degrees rotation of the environment map.

Figure B.4: Relighting of the plant in the eucalyptus grove using a point integration. **Left:** Original orientation of the environment map. **Left:** 180 degrees rotation of the environment map.
Figure B.5: Relighting of the helmet in the Uffizi gallery using a point integration. **Left:** Original orientation of the environment map. **Left:** 270 degrees rotation of the environment map.

Figure B.6: Relighting of the plant in the Uffizi gallery using a point integration. **Left:** Original orientation of the environment map. **Left:** 270 degrees rotation of the environment map.
Appendix C

Office room results

Renderings of the bird and the egg with the office room method are presented in this appendix. Every result is computed using an optimisation in original space as it gives the best results. The specular highlights are visible in every picture.

Figure C.1: Relighting of the bird in the Eucalyptus Grove

Figure C.2: Relighting of the bird in the Pisa courtyard
Figure C.3: Relighting of the bird in the Uffizi gallery.

Figure C.4: Relighting of the bird in St Peters Basilica.

Figure C.5: Relighting in St Peters Basilica. Left: Office room relighting. Right: Ground truth.
Appendix D

Environment maps

The environment maps used in this project are presented in this appendix. The oldest maps have a low resolution and are available on Paul Debevec’s website [10]. The newest and high resolution environment maps are available on the University of Southern California website [16]. The higher the resolution the more information about light intensities and colors the environment can store. Besides high resolution light probes avoid aliasing artefacts when the background is ray traced but the computation is noticeably slower. Some environment maps such as the Grace cathedral (figure D.1) have high frequency lighting meaning that the intensity and the colors of the light sources vary rapidly in the image. Others such as the Uffizi gallery (figure D.2) have a much lower frequency lighting. This property of environment maps is important for image-based relighting and especially for the office room relighting as few lights sources are used.

Figure D.1: Grace cathedral latitude longitude map. Source [10]

Figure D.2: Uffizi gallery latitude longitude map. Source [10]
Figure D.3: Pisa courtyard latitude longitude map. Source [16]

Figure D.4: Eucalyptus Grove latitude longitude map. Source [10]

Figure D.5: St. Peter’s Basilica latitude longitude map. Source [10]