École polytechnique de Louvain

Blind De-Rendering

or How to Turn a Wall into a Mirror

Author: **Jean-Louis Voiseux**
Supervisors: **Laurent Jacques, Laurent Demanet**
Reader: **Christophe De Vleeschouwer**
Academic year 2019–2020
Master [120] in Electrical Engineering

Acknowledgements

First of all, I would like to offer my thanks to my master thesis supervisors, Prof. Laurent Jacques and Prof. Laurent Demanet. More than once, the excellent feedback and remarks provided during our meetings have prevented me from losing track of the objective of this work, and their suggestions often proved to be the key to the unblocking of several problems, leading to crucial improvements. Their availability during the unusual circumstances of Spring 2020 also proved to be invaluable.

I would also like to thank my family and friends for their support, and especially to the team at Altheria Solutions for their advice regarding high-performance computing, as well as their repeated signs of encouragement.

"Les miroirs, ces diseurs de vérités, sont haïs ; cela ne les empêche pas d'être utiles."

Victor Hugo, Letter to M. Daelli (publisher of the italian translation of "Les Misérables")

Abstract

This work considers the problem of recovering both the BRDF of a surface and the light distribution incident to that surface from the outbound light distribution. This problem is referred to as blind de-rendering. The proposed geometrical configuration, which consists in a small light source reflected on a flat wall, allows for simplifying assumptions that lead to the reformulation of de-rendering as well-known inverse problems.

One of those inverse problems is the deconvolution problem. Assumptions leading to the formulation of the rendering equation as a convolution are detailed. Numerical experiments demonstrate the effectiveness of deconvolution algorithms in the non-blind case. The limitations of relevant single and multi-channel blind deconvolution algorithms are then discussed.

Rendering can also be expressed as a linear system. An algorithm for blind de-rendering is developed based on this observation, and its performance is assessed. Diversity related to the source and to the BRDF is exploited, and useful matrix manipulations tied to the aforementioned geometrical configurations are detailed.

Contents

Introduction

"How to turn a wall into a mirror". Such a descriptive and ambitious title requires some contextualization. The goal of this master thesis is better understood when considering the properties of a mirror : it is a material that provides perfect specular reflection. For a light source position whose forward direction is perpendicular to the mirror plane, the reflected light distribution will be identical to the incident light distribution.

In other words, if we consider the reflecting surface like a system whose input is an emitted light intensity and whose output is a reflected light intensity , the mirror "system" is equivalent to the identity. The output is equal to the input. Consequently, if one manages to recover the (unknown) light distribution incident to the system, the distribution recovered is equivalent to the one that would be reflected in the mirror case. **"Recovering the incident light distribution from a reflected pattern is akin to turning a wall into a mirror."**

Recovery of the inbound light distribution also allows us to acquire information about hidden objects. Let us imagine that we are standing in front of a wall, behind a small card that projects light on the wall. We cannot see the card, but only the light distribution reflected by the wall. Recovering the light incident to the wall, i.e. the light projected by the card, is equivalent to recover the pattern of the card itself.

Several frameworks allowing us to model light transport across surfaces and global illumination have been developed. Among them, the most successful is without a doubt the rendering equation [1], described by Kajiya in 1986. This equation will be derived in the first chapter of this work, but for the purpose of this introduction, we will write it below :

$$L_r(\boldsymbol{x}, \boldsymbol{\omega_r}) = L_e(\boldsymbol{x}, \boldsymbol{\omega_r}) + \int_{\Omega_{\boldsymbol{x}}} \rho(\boldsymbol{x}, \boldsymbol{\omega_i}, \boldsymbol{\omega_r}) L_r(\boldsymbol{x}', -\boldsymbol{\omega_i}) \cos\theta_i \, d\boldsymbol{\omega_i} \tag{1}$$

This equation basically expresses the fact that the total light intensity radiating from a point $\boldsymbol{x}$ to a direction $\boldsymbol{\omega_r}$ is the sum of the light intensity emitted by that point and the light radiated by that point. The radiated light is the integration over all incoming direction of the incident light, multiplied by a function ρ that computes the proportion of light reflected by the point, for a given outbound and inbound direction.

Let us focus on the integral term. Our goal, stated in the first two paragraphs of this introduction, is the recover $L_r(\boldsymbol{x}', -\boldsymbol{\omega_i})$, the incident light reflected by a point $\boldsymbol{x}'$ on the wall. However, if we aim to recover that quantity starting with only the total reflected light $L_r(\boldsymbol{x}, \boldsymbol{\omega_r})$, we also need to recover ρ, which we call the BRDF. The recovery of those two quantities from the total reflected light is the blind de-rendering problem, and **our first goal is to identify an algorithm that allows for the simultaneous recovery of both**.

Problems similar to the one we are going to study are not rare in the literature. If we focus solely on inverse light transport, the field of non-light of sight imaging, which is about acquiring information about hidden scenes, has produced interesting topics. "Computational mirrors"[2], "Light field imaging from shadows"[3], or "Turning corners into cameras"[4] are projects the rely on light transport to infer information about hidden objects.

Studying the rendering equation in the framework of signal processing is not a new idea either. An entire project has been dedicated to inverse rendering as a deconvolution [5], and our own formulation of the rendering equation as a convolution was heavily inspired by an existing framework for frequency analysis of light transport[6].

Our task is simplified by the constrained geometrical configuration for which we need to solve this problem : a small light source faces a single flat wall. This configuration will allow us to simplify the problem, and to express rendering as either a convolution (Chapter 3) or a linear system (Chapter 4). Those two chapters should be seen as independent from each other, as both explore a different way of solving the inverse problem.

Recovering the aforementioned quantities from a set of observations is called an inverse problem (rendering itself being the forward problem). Inverse problems are often ill-posed, which means that their solution either does not exists, is not unique, or is unstable. Throughout this work, we will need to do several assumptions on the incident light distribution and the BRDF that improve our ability to solve the inverse problem. **Formulating the hypothesis that ensure the recovery the recovery of the incident light distribution and the BRDF for the chosen de-rendering algorithm is the second goal of this thesis**.

Notation conventions

Throughout this work, our notations will stick to the conventions described in Table 1.

Notation	Meaning
x, X	Non-bold symbol : scalar
$\boldsymbol{x}$	Lowercase bold symbol : vector
$\boldsymbol{X}$	Uppercase bold symbol : matrix
$\boldsymbol{X}^T$	Matrix transpose
Ω_x	Describes a set of elements

Table 1: Additional notations for discretization

Chapter 1

The Rendering Equation

In this chapter, we introduce the notion of BRDF, derive the rendering equation, formulate a solution and showcase the path-tracing algorithm. Having a good grasp on rendering, the forward problem, is essential to solve the inverse problem.

1.1 Radiometric quantities

We start by formulating our spherical coordinate system, in order to introduce the concepts of solid angle, radiance and irradiance.

1.1.1 Spherical coordinates

A spherical coordinate system is a coordinate system that can be used to describe the position of a point in a three-dimensional space, using three numbers. While numerous conventions [7] are used throughout the literature, we chose to use the standard ISO convention, described as follows :

- r, the radial distance, describes the distance of a point from the origin. We have $r \geq 0$
- θ, the polar angle (from the positive z axis), describes the elevation of the point. We have $\theta \in [0, \pi]$
- ϕ, the azimuthal angle (from the positive x axis) of the point in the xy plane. We have $\phi \in [0, 2\pi[$

The coordinate system in represented on Figure 1.1. Using the expressions below, one relate our spherical coordinate system (r, θ, ϕ) to the cartesian coordinates (x, y, z) :

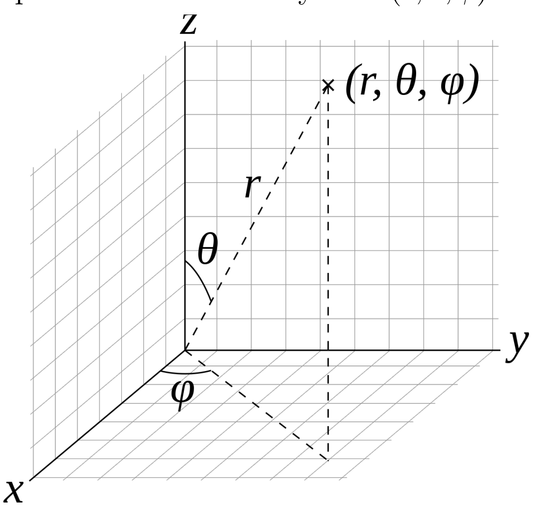

Figure 1.1: Spherical coordinates [8]

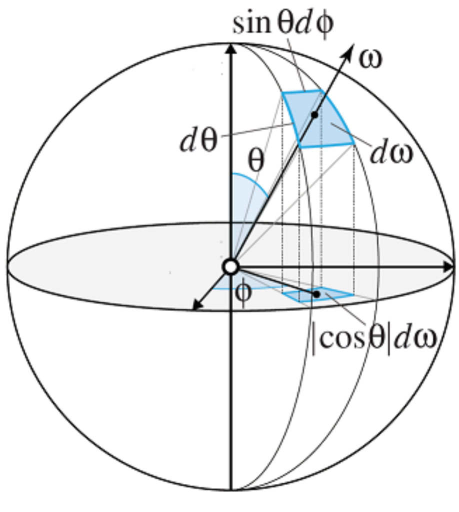

Figure 1.2: Solid angle [9]

$$r = \sqrt{x^2 + y^2 + z^2} \qquad \theta = \arccos\frac{z}{r} \qquad \phi = \arctan\frac{y}{x}$$
$$x = r\sin\theta\cos\phi \qquad y = r\sin\theta\sin\phi \qquad z = r\cos\theta$$

1.1.2 Solid angles

The solid angle [10] $d\omega$ of an infinitesimal surface element dS, as seen from a given point, is defined as the area of dS divided by the square of the distance to that point. In the following applications, we consider dS to be an infinitesimal element of the surface of a sphere of radius r. Since this element is infinitely tiny, its sides can be considered flat, and its area is the product of the length of its sides.

Considering the spherical coordinate system previously defined, the side with constant ϕ is a circular arc of length $rd\theta$. The side with constant θ is also a circular arc, whose radius depends on the elevation angle θ. Its length is thus equal to $r\sin\theta d\phi$. We finally obtain the expression of the solid angle corresponding to our small surface element (see Figure 1.2) :

$$d\omega = \frac{dS}{r^2} := \frac{r^2\sin\theta d\theta d\phi}{r^2} = \sin\theta d\theta d\phi. \tag{1.1}$$

1.1.3 Radiance and Irradiance

Given an infinitesimal area A, the radiance [11][12] is the radiant flux (the radiating power) emitted per unit projected area (in the emitting direction) per unit solid angle. It can be written as :

$$L(\boldsymbol{x}, \omega) = \frac{\partial^2\boldsymbol{\Phi}}{\partial A_\perp \partial\omega}. \tag{1.2}$$

- $\boldsymbol{x}$ is the position of the emitting point.
- $\boldsymbol{\omega}$ is the direction in which the light is emitted. Its associated solid angle $d\omega$ corresponds to the surface seen by the detector on the unit hemisphere surrounding the emitting surface.
- Φ is the radiant flux.
- $A_\perp$ is the projection of the emitting area towards the direction of emission.

Under the assumption that the emitting surface is Lambertian [13], the radiant power observed from the surface is directly proportional to the cosine of angle θ between the surface normal and the direction of the ray cast by the observer to the surface. As such, we can rewrite the radiance as a function of A, the emitting area :

$$L(\boldsymbol{x}, \omega) = \frac{\partial^2\boldsymbol{\Phi}}{\partial(A\cos\theta)\,\partial\omega}. \tag{1.3}$$

The irradiance is the radiant flux received per unit area :

$$E(\boldsymbol{x}, \omega) := \frac{\partial\boldsymbol{\Phi}}{\partial A}. \tag{1.4}$$

The infinitesimal irradiance dE can thus be defined as the incident radiance seen by a solid angle $d\omega$:

$$dE(\boldsymbol{x}, \omega) = L(\boldsymbol{x}, \boldsymbol{\omega})\cos\theta\, d\omega. \tag{1.5}$$

1.2 BRDF and the Reflection Equation

Each material reflects a portion of the incoming light. A way to quantify this portion is to use the concept of reflectance. The reflectance of a surface can be expressed as follows [14] :

$$\rho = \frac{L_r}{E_i}, \tag{1.6}$$

where L_r and E_i are the reflected radiance and the incident irradiance, respectively. It is worth noting that this definition of reflectance only applies to isotropic cases, as the radiances have no directional dependency. However, the appearance of real materials depends on the orientation of both the light source and the viewer : as such, we wish to generalize the reflectance concept by taking in account the incident and observation directions.

To do so, we start by defining two solid angles, $d\omega_r$ and $d\omega_i$, respectively corresponding to $\boldsymbol{\omega}_r$ and $\boldsymbol{\omega}_i$, the directions of reflection and of incidence. We define the BRDF (Bidirectional Reflectance Distribution Function) as as the ratio between the power in a particular outgoing direction $\boldsymbol{\omega}_r$ (infinitesimal radiance) and the power in a particular incoming direction $\boldsymbol{\omega}_i$ (infinitesimal irradiance) :

$$\rho(\boldsymbol{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_r) := \frac{dL_r(\boldsymbol{x}, \boldsymbol{\omega}_r)}{dE(\boldsymbol{x}, \boldsymbol{\omega}_i)}, \tag{1.7}$$

dE is the irradiance linked to the incident radiance L_i. Using (1.5), we can rewrite the BRDF as follows (see Figure 1.3) :

$$\rho(\boldsymbol{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_r) = \frac{dL_r(\boldsymbol{x}, \boldsymbol{\omega}_r)}{L_i(\boldsymbol{x}, \boldsymbol{\omega}_i) \cos\theta_i \, d\omega_i}. \tag{1.8}$$

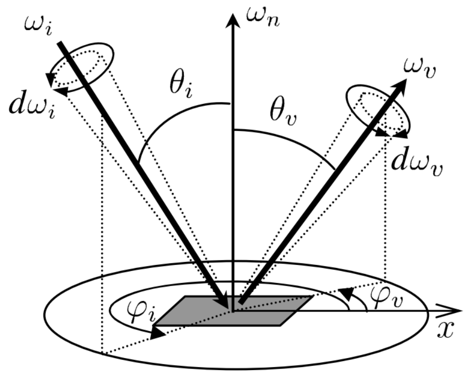

Figure 1.3: BRDF representation [14]

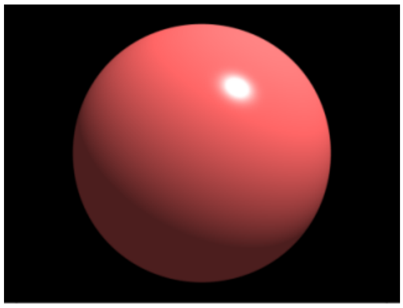

Figure 1.4: Phong Model [15]

BRDF models are linear, which will allow us to express the reflected light in one direction as the sum of the contribution of all incident directions (see (1.14)). In practice, many reflectance models are simplified subsets of the BRDF (such as the diffuse case, described below). In order to accurately represent physical surfaces, a BRDF should respect the following properties [16] :

- Non-negativity : $\forall \boldsymbol{\omega}_i, \boldsymbol{\omega}_r, \quad \rho(\boldsymbol{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_r) \geq 0$
- Reciprocity : The light path is reversible : $\boldsymbol{\omega}_i$ and $\boldsymbol{\omega}_r$ can be inverted without impacting the fraction of reflected light.
- Energy conservation : The energy reflected should not exceed the total incident energy :

$$\forall \boldsymbol{\omega}_i, \quad \int_\Omega \rho(\boldsymbol{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_r) \cos\theta_r d\boldsymbol{\omega}_r \leq 1. \tag{1.9}$$

BRDF models can be either isotropic or anisotropic [17]. An isotropic BRDF is invariant to the rotation around the normal of the considered surface. As such, we can rewrite it as follows :

$$\rho(\boldsymbol{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_r) = \rho(\boldsymbol{x}, (\theta_i, \phi_i), (\theta_o, \phi_o)) = \rho(\boldsymbol{x}, \theta_i, \theta_o, \phi_o - \phi_i). \tag{1.10}$$

In several cases, this description is insufficient. Hair, and brushed metal are two surface types that are not well modelled under the assumption of variance to rotation around the surface normal.

Practical examples of BRDFs include [18]:

- The mirror case : the incoming light is reflected across the normal. Intuitively, the condition for a reflected ray of light to be seen by the observer in a 3D space is that θ, the elevation angle, must be the same as the source, and that their ϕ is opposed. We obtain the following expression :

$$\rho(\boldsymbol{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_r) = \frac{\delta(\cos\theta_i - \cos\theta_r)}{\cos\theta_r}\delta(\phi_i - \phi_r + \pi). \tag{1.11}$$

- The diffuse case : the incoming light is reflected uniformly in all directions. As such, there is no direction dependency in the expression of the BRDF, which is a constant, often written as a quotient between the albedo α and π :

$$\rho(\boldsymbol{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_r) = \frac{\alpha}{\pi}. \tag{1.12}$$

- Empirical or physical reflection models, such as the Phong model or the Cook-Torrance model (see Figure 1.4).

Rearranging the terms of equation , we obtain the expression of the reflected radiance due to the local reflection associated with $\boldsymbol{\omega}_i$:

$$dL_r(\boldsymbol{x}, \boldsymbol{\omega}_r) = \rho(\boldsymbol{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_r) L_i(\boldsymbol{x}, \boldsymbol{\omega}_i) \cos\theta_i \, d\boldsymbol{\omega}_i. \tag{1.13}$$

The total light reflected from a surface locally in $\boldsymbol{x}$ and then reflected in the direction $\boldsymbol{\omega}_r$ is then given by the following expression, which consists in integrating the contributions of all incident radiances over the hemisphere surrounding the surface of interest :

$$L_r(\boldsymbol{x}, \boldsymbol{\omega}_r) = \int_{\Omega_{\boldsymbol{x}}} \rho(\boldsymbol{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_r) L_i(\boldsymbol{x}, \boldsymbol{\omega}_i) \cos\theta_i \, d\boldsymbol{\omega}_i. \tag{1.14}$$

1.3 Formulating the Rendering Equation

The equation (1.14) only takes in account direct incident lights, and thus only allows us to compute the shading of surfaces directly exposed to light sources. As such, some modifications are required in order to solve the global illumination problem. Most importantly, indirect reflections should be taken in account by our model.

First, we write a slightly modified version of (1.14), taking in account the possibility of the surface being a light source :

$$L_r(\boldsymbol{x}, \boldsymbol{\omega}_r) = L_e(\boldsymbol{x}, \boldsymbol{\omega}_r) + \int_{\Omega_{\boldsymbol{x}}} \rho(\boldsymbol{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_r) L_i(\boldsymbol{x}, \boldsymbol{\omega}_i) \cos\theta_i \, d\boldsymbol{\omega}_i, \tag{1.15}$$

Then, in order to account for indirect reflections, we make the assumption that L_i, the radiance incident to our surface of interest, is actually the radiance reflected by another point $\boldsymbol{x}'$ in the direction $-\boldsymbol{\omega}_i$. Thus, we obtain [19]:

$$L_r(\boldsymbol{x}, \boldsymbol{\omega}_r) = L_e(\boldsymbol{x}, \boldsymbol{\omega}_r) + \int_{\Omega_{\boldsymbol{x}}} \rho(\boldsymbol{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_r) L_r(\boldsymbol{x}', -\boldsymbol{\omega}_i) \cos\theta_i \, d\boldsymbol{\omega}_i. \tag{1.16}$$

We now have a model for the global illumination problem. However, the (1.16) contains L_r terms at both sides of the expression. In the following section, we describe the method employed to solve such an equation.

1.4 Solving the Rendering Equation

Eq. (1.16) is a Fredholm equation of the second kind, whose canonical form is [20]:

$$y(x) = f(x) + \lambda \int_{\Omega} K(x, \omega) y(\omega) d\omega. \tag{1.17}$$

These equations can be solved by constructing the Neumann Series of the kernel K. To do so, we rewrite (1.16) using the linear operator theory. We define the transport operator T (which effectively represents the action of the kernel of (1.16) on L_r) as follows :

$$TL_r := \int_{\Omega_{\boldsymbol{x}}} \rho(\boldsymbol{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_r) L_r(\boldsymbol{x}', -\boldsymbol{\omega}_i) \cos\theta_i \, d\boldsymbol{\omega}_i. \tag{1.18}$$

We can thus rewrite the (1.16) using the operator formalism :

$$L_r = L_e + TL_r, \tag{1.19}$$

By applying (1.19) inductively, we obtain :

$$L_r = L_e + TL_e + T^2 L_e + L^3 L_e + ... \tag{1.20}$$

This infinite series converges if $|T| < 1$, which is the case here, using the law of conservation of energy. As such, the (1.20) converges to the exact solution of the rendering equation. Additionally, if $|T| < 1$, we have :

$$L_r = (1 - T)^{-1} L_e. \tag{1.21}$$

This result has an intuitive physical interpretation :

- The L_e term corresponds to the emission from the light source contained in the surface.
- The TL_e term corresponds to direct illumination of the surface.
- The $T^2 L_e$ term corresponds to the indirect illumination due to one "bounce" of the light rays
- ...and so on.

Practically, there are two commonly used ways [21] to numerically solve the rendering equation : Finite-Element methods and Monte Carlo methods. An illustration of the first class of methods is the Radiosity computation, while modern path tracers implement the second class of methods. One of the main drawbacks of Radiosity lies in its inability to easily render non-diffuse surfaces, while Monte Carlo methods allow for the computation of very good approximate solutions to the rendering equation.

In preparation for this document, and to get a intuition for the practical resolution of the rendering equation, a path-tracer (inspired by [22]) was implemented in MATLAB. The resulting render is displayed on Figure 1.5. The walls have a diffuse BRDF, the left sphere has a mirror BRDF, and the right sphere has a glass-like dielectric BRDF.

The working principle of the path tracer is straightforward : for each pixel, cast numerous random rays and let them bounce several times, ideally until they reach an emissive body, then sum the contribution of all the rays in order to render the pixel. The underlying mathematics are described here [23].

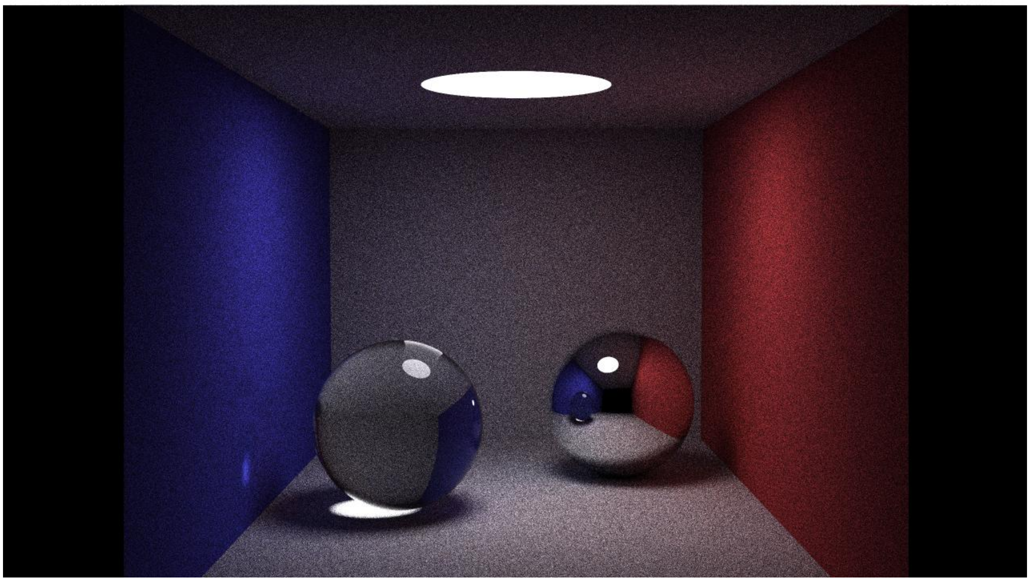

Figure 1.5: Render obtained using our own path tracer. Since the focus was on understanding its underlying principles, there were no optimisations, hence the mediocre quality of the render.

Chapter 2

Rendering Reflections on a Wall

In this Section, we formulate the rendering equation in a very specific configuration : a small light source (with a given light distribution : think of a playing card or a mobile phone) faces a flat wall, and an observer placed in front of the same wall obtains the reflection of the source on the wall.

2.1 Wall Reflections : Simplifying the Rendering Equation

Restricting the rendering process to the aforementioned configuration allows us to make several assumptions regarding the BRDF of the wall and the reflected object. These assumptions will then allow us to :

- Implement a rendering algorithm way simpler than path-tracing, but that fully solves the direct rendering problem in our case and provides an input for the inverse problem that we aim to solve.
- Show that rendering is equivalent to well-known operations (convolution in section 3.1, matrix product in section 4.1). That equivalence will ease the construction of a resolution method for the inverse problem.

2.1.1 Notations and Geometrical Configuration

Our geometrical configuration of interest is as follows :

> **Hypothesis 1 - Geometrical configuration** : A light source is placed at a known position in front of (parallel to) a wall with a given BRDF and a known normal vector. The reflection of the source on the wall is seen by an observer modelled by a point, whose position is also known.

The notations used in this Section are given in the table 2.1. We state the following conventions :

- The wall is defined as the plane located in $(x, 0, z)$.
- Consequently, its normal is the vector $\boldsymbol{n_w} = (x, -1, z)$
- The observer is positioned in $(0, y, 0)$.

In order to adapt the rendering equation defined in the previous chapter, we rewrite (1.16) below and insert the quantities defined above :

$$L_r(\boldsymbol{x}, \boldsymbol{\omega_r}) = L_e(\boldsymbol{x}, \boldsymbol{\omega_r}) + \int_{\Omega_{\boldsymbol{x}}} \rho(\boldsymbol{x}, \boldsymbol{\omega_i}, \boldsymbol{\omega_r}) L_r(\boldsymbol{x}', -\boldsymbol{\omega_i}) \cos\theta_i \, d\boldsymbol{\omega_i}.$$

Notation	Meaning
$\boldsymbol{o}$	Position of the observer. $\boldsymbol{o} \in \mathbb{R}^3$
$\boldsymbol{w}$	Position of a point on the wall. $\boldsymbol{w} \in \mathbb{R}^3$
$\boldsymbol{n_w}$	Unit vector normal to the wall. $\boldsymbol{n_w} \in \mathbb{R}^3$
$\boldsymbol{s}$	Position of a point of the light source. $\boldsymbol{s} \in \mathbb{R}^3$
$\boldsymbol{r_o}$	Vector pointing from the point $\boldsymbol{x}$ to the position $\boldsymbol{o}$ of the observer. $\boldsymbol{r_o} \in \mathbb{R}^3$
$\boldsymbol{r_s}$	Vector pointing from the point $\boldsymbol{x}$ to the point $\boldsymbol{s}$ of the light source. $\boldsymbol{r_s} \in \mathbb{R}^3$
$\boldsymbol{\omega_o}$	Unit vector corresponding to the direction of $\boldsymbol{r_o}$. $\boldsymbol{\omega_o} \in \mathbb{R}^3$
$\boldsymbol{\omega_s}$	Unit vector corresponding to the direction of $\boldsymbol{r_s}$. $\boldsymbol{\omega_s} \in \mathbb{R}^3$
θ_s	Elevation angle between $\boldsymbol{n_w}$ and $\boldsymbol{\omega_s}$. $\theta_s \in [0, \frac{\pi}{2}]$
θ_o	Elevation angle between $\boldsymbol{n_w}$ and $\boldsymbol{\omega_o}$. $\theta_o \in [0, \frac{\pi}{2}]$

Table 2.1: Notations

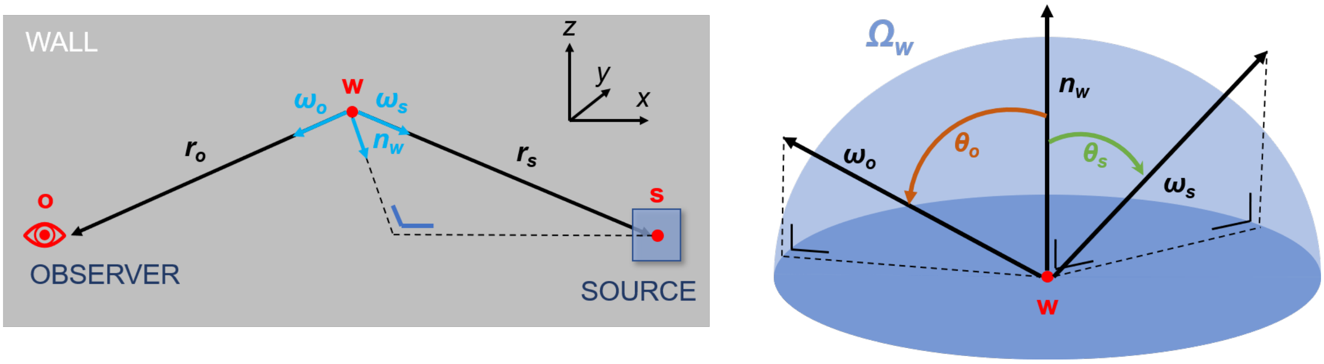

Figure 2.1: Geometrical configuration

$$L_r(\boldsymbol{w}, \boldsymbol{\omega_o}) = L_e(\boldsymbol{w}, \boldsymbol{\omega_o}) + \int_{\Omega_w} \rho(\boldsymbol{w}, \boldsymbol{\omega_s}, \boldsymbol{\omega_o}) L_r(\boldsymbol{s}, -\boldsymbol{\omega_s}) \cos\theta_s \, d\boldsymbol{\omega_s}. \tag{2.1}$$

To give the reader a better intuition for our geometrical configuration, we provide a descriptive diagram on Figure 2.1. That specific geometrical configuration allows us to make assumptions on light transport, on the wall and on its BRDF.

This geometrical configuration has also a direct impact on the expression of $L_r(\boldsymbol{x}, \omega)$ and $L_e(\boldsymbol{x}, \omega)$, respectively the total intensity of light emitted/reflected by a point $\boldsymbol{x}$ in the direction $\boldsymbol{\omega}$.

Since we know the position of the observer and the normal vector of the wall, there is a bijection between the set containing all the possible $\boldsymbol{\omega_o}$ and all the points of the wall. In other words, given the position of the observer, the normal of the wall and a direction $\boldsymbol{\omega_o}$, it is possible to find the corresponding wall point $\boldsymbol{w}$. By definition of $\boldsymbol{\omega_o}$, we have :

$$\boldsymbol{\omega_o} = \frac{\boldsymbol{o} - \boldsymbol{w}}{||\boldsymbol{o} - \boldsymbol{x}||_2} \quad \longleftrightarrow \quad \boldsymbol{w} = \boldsymbol{o} - \boldsymbol{\omega_o} ||\boldsymbol{r_o}||_2, \tag{2.2}$$

Finally, we use trigonometry, as well as the definition of dot product, to compute $||\boldsymbol{r_o}||_2$. Following our conventions, we assume that the observer is positioned at $(0, y_o, 0)$.

$$\boldsymbol{\omega_o} \cdot \boldsymbol{n_w} = ||\boldsymbol{\omega_o}|| \; ||\boldsymbol{n_w}|| \; \cos\theta_o \quad \longleftrightarrow \quad \cos\theta_o = \boldsymbol{\omega_o} \cdot \boldsymbol{n_w}, \tag{2.3}$$

$$\boldsymbol{w} = \boldsymbol{o} - \boldsymbol{\omega_o} \frac{y_o}{\cos\theta_o} = \boldsymbol{o} - \boldsymbol{\omega_o} \frac{y_o}{\boldsymbol{\omega_o} \cdot \boldsymbol{n_w}}. \tag{2.4}$$

Since we also know the distance between the source and the wall (i.e. y_s), we can also, for a given $\boldsymbol{w}$, link each $\boldsymbol{\omega_s}$ to its corresponding source point using a similar reasoning :

$$\boldsymbol{s} = \boldsymbol{w} + \boldsymbol{\omega_s} \frac{y_s}{\boldsymbol{\omega_s} \cdot \boldsymbol{n_w}}. \tag{2.5}$$

As such, we can rewrite (2.1) as follows :

$$L_r(\boldsymbol{\omega_o}) = L_e(\boldsymbol{\omega_o}) + \int_{\Omega_{\omega_o}} \rho(\boldsymbol{w}, \boldsymbol{\omega_s}, \boldsymbol{\omega_o}) L_r(-\boldsymbol{\omega_s}) \cos\theta_s \, d\boldsymbol{\omega_s}. \tag{2.6}$$

It is worth noting that $\Omega_{\boldsymbol{x}}$ and Ω_{ω_o} are strictly equivalent, since each $\boldsymbol{\omega_o}$ is linked to an unique $\boldsymbol{x}$.

2.1.2 Assumptions and Problem Simplification

Assumptions on Light transport

In section 1.4, the transport operator T was defined in order to find a solution to the rendering equation. We redefine this operator with actualized notations in (2.7).

$$TL := \int_{\omega_o} \rho(\boldsymbol{x}, \boldsymbol{\omega_o}, \boldsymbol{\omega_s}) L(\boldsymbol{s}, -\boldsymbol{\omega_s}) \cos\theta_s \, d\boldsymbol{\omega_s}. \tag{2.7}$$

As shown in section 1.4, the exact solution to the rendering equation is an infinite sum of terms involving this transport operator. Each of these terms corresponds to one bounce of a ray whose origin is the observer. As such, this solution models indirect lighting, and allows for the rendering of realistic light reflection and transmission. However, as previously stated, our goal is to obtain an approximation of this solution using a well-known operator. We cannot possibly take all these terms into account.

For some materials, it is possible to obtain a reasonably good approximation of the solution of the rendering equation by taking only a few terms. To illustrate this, we provide two renders (Figures 2.2 and 2.3) computed by an improved (using details provided by [23]) version of the path-tracer showcased in section 2. This path-tracer is written in C++, and the output is generated using OpenCV.

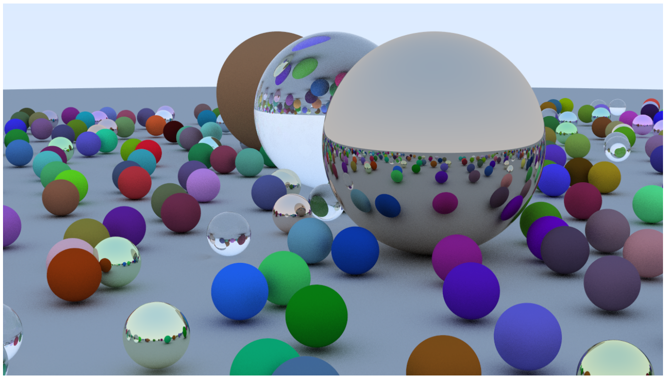

Figure 2.2: Path-tracing result (50 bounces)

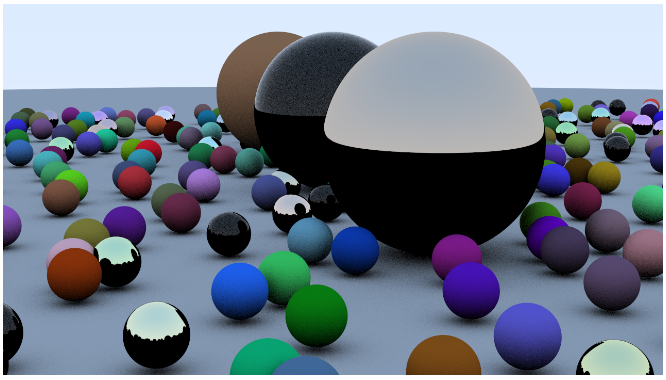

Figure 2.3: Path-tracing result (1 bounce)

The impairments due to taking only the first two terms (one bounce) differ depending on the type of BRDF. Glass-like materials, which rely on transmission and thus on multiple bounces, suffer the most (see the large sphere in the middle of Figure 2.3). Mirror materials use the only bounce at their disposal to differentiate light sources (here, the sky) and geometry elements, but are unable to retrieve the exact color of the latter. On the other hand, diffuse materials are rendered fine, with the exception of the lower part of the spheres, which does not benefits from the reflection of the light on the ground.

However, in our configuration of interest, a single light source Ω_s is placed in front of a diffuse wall Ω_w, and both are illuminated by an all-encompassing light distribution. As such, under the assumption of a diffuse wall, taking only the first two terms of the sum gets us very close to to exact solution. We illustrate this by showcasing two renders (Figures 2.4 and 2.5) of an enlarged

Cornell Box. The wall of interest is the back wall, and a blue light source is oriented towards it. The rendered back wall considering only one term is really similar to the render taking 50 terms into account.

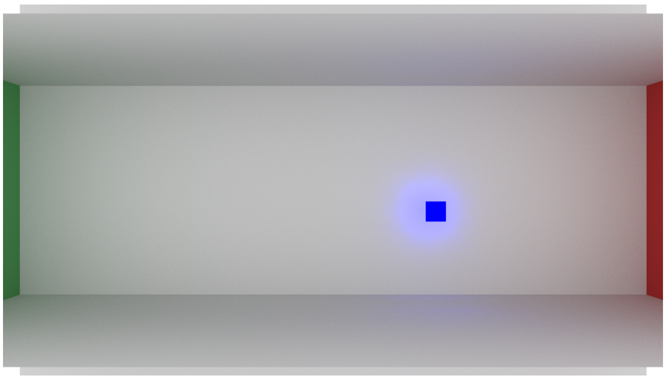

Figure 2.4: Path-tracing result (50 bounces)

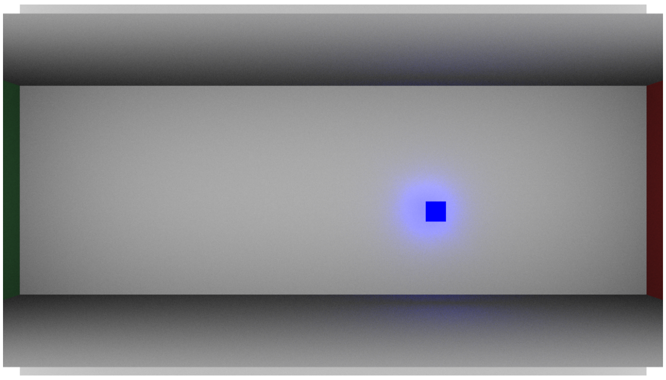

Figure 2.5: Path-tracing result (1 bounce)

We can also extend our reasoning to a mirror BRDF. Given our geometrical configuration, the entirety of the source can be reflected to the observer, even with only one bounce. This is illustrated by two additional renders (Figures 2.6 and 2.7). To better showcase the reflection of the source, the source itself is hidden. The only elements properly reflected are the light sources in the direct line-of-sight of the wall (light source, uniform background light). Fortunately, the geometrical configuration described above only involves a light source in the direct line-of-sight of the wall of interest.

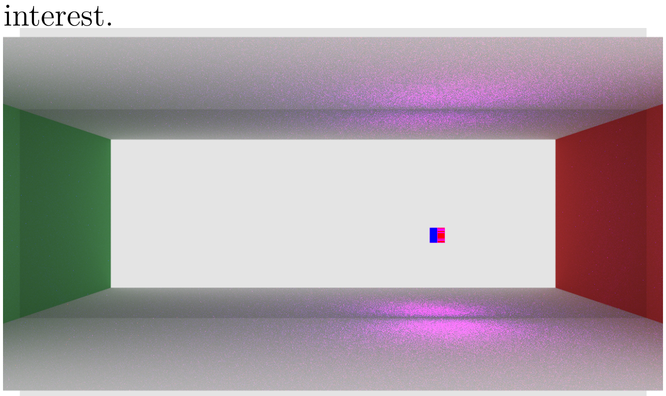

Figure 2.6: Path-tracing result (50 bounces)

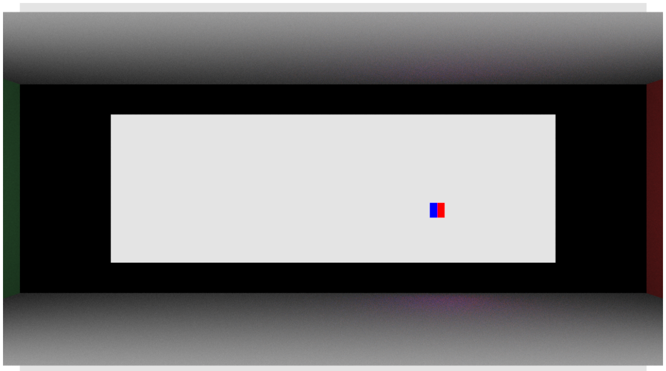

Figure 2.7: Path-tracing result (1 bounce)

Since the one-bounce approximation is satisfactory for both diffuse and specular reflection for our geometrical configuration, we assume that this approximation will yield good results for all BRDFs. The examples above have shown that glass-like materials, involving transmission modelling, cannot be appropriately rendered under the "one-bounce" assumption. However, by definition, the BRDF only contains reflection term. These materials are thus automatically excluded by our decision to model light transport by materials using the BRDF.

Hypothesis 2 - "One-Bounce" approximation : When solving the rendering equation for the reflection of a light source on a wall, the solution is well approximated considering only one light bounce :

$$L_r = L_e + TL_e + T^2L_e + L^3L_e + ... \simeq L_e + TL_e.$$

As such, we can simply express the radiance emitted by the wall point $\boldsymbol{x}$ in the observer direction $\boldsymbol{\omega_o}$ as the sum of the light emitted by the wall itself and a contribution from a source point $\boldsymbol{s}$, emitted in the direction $-\boldsymbol{\omega_s}$.

$$L_r(\boldsymbol{\omega_o}) = L_e(\boldsymbol{\omega_o}) + TL_e(-\boldsymbol{\omega_s}), \tag{2.8}$$

Developing (2.8), we obtain the following solution to the rendering equation :

$$L_r(\boldsymbol{\omega_o}) = L_e(\boldsymbol{\omega_o}) + \int_{\Omega_{\boldsymbol{\omega_o}}} \rho(\boldsymbol{w}, \boldsymbol{\omega_s}, \boldsymbol{\omega_o}) L_e(-\boldsymbol{\omega_s}) \cos\theta_s \, d\boldsymbol{\omega_s}. \tag{2.9}$$

Assumptions on the Wall and its BRDF

We can simplify further (2.9) under the assumption that the wall itself is not a light source.

Hypothesis 3 - Passive reflector : The wall of interest is not a light source.

$$L_e(\boldsymbol{\omega_o}) = 0 \quad \forall \boldsymbol{\omega_o}.$$

Moreover, we also make the arbitrary assumption that the BRDF is identical for all the points of the wall.

Hypothesis 4 - Uniform wall : The BRDF is identical for all wall points.

$$\rho(\boldsymbol{w}, \boldsymbol{\omega_s}, \boldsymbol{\omega_o}) = \rho(\boldsymbol{\omega_s}, \boldsymbol{\omega_o}) \quad \forall \boldsymbol{w}.$$

These two assumptions allow us to rewrite (2.9) as follows :

$$L_r(\boldsymbol{\omega_o}) = \int_{\Omega_{\boldsymbol{\omega_o}}} \rho(\boldsymbol{\omega_s}, \boldsymbol{\omega_o}) L_e(-\boldsymbol{\omega_s}) \cos\theta_s \, d\boldsymbol{\omega_s}. \tag{2.10}$$

Assumption on the Light Source

This last assumption does not contribute to the simplification of the rendering equation, but greatly facilitate the implementation and the description of later algorithms.

Hypothesis 5 - Isotropic light source : All source points emit isotropically : for a given point ("pixel") of the light source, we have :

$$L_e(-\boldsymbol{\omega_s}) = C \quad \forall \boldsymbol{\omega_s} \qquad C \in \mathbb{R}.$$

That assumption does not imply that all source points emit the same light intensity, of that a given point emits light with the same intensity on the whole wall.

2.1.3 2D Formulation of the Direct Problem

Formulating (2.10) in 2D has several advantages :

- Further mathematical developments (notably involving the observer and its associated camera) will be greatly eased, allowing us to focus on building a conceptual solution to both the direct and inverse problem.
- The computational complexity of the algorithms used for our numerical experiments will be reduced.

To proceed, we need to define two more angles related to the observer (see table 2.2). Furthermore, in the BRDF, $\boldsymbol{\omega_s}$ and $\boldsymbol{\omega_o}$ will be respectively replaced by θ_s and θ_o, as these two angles completely describe the inbound and outbound directions for a given $\boldsymbol{o}$, $\boldsymbol{w}$ and $\boldsymbol{s}$ in 2D.

Similarly to the 3D case, we provide a descriptive diagram of the 2D geometrical configuration on Figure 2.8. Finally, we can further simplify (2.10) :

$$L_r(\theta_o) = \int_{\Omega_{\theta_o}} \rho(\theta_s, \theta_o) L_e(\theta_s) \cos\theta_s \, d\theta_s. \tag{2.11}$$

Notation	Meaning
α_n	Negative[1] angle between $-\boldsymbol{n_w}$ and the border of the camera field of view. $\alpha_n \in]-\frac{\pi}{2};0[$
α_1	Positive angle between $-\boldsymbol{n_w}$ and the border of the camera field of view. $\alpha_1 \in]0;\frac{\pi}{2}[$

Table 2.2: Additional notations for 2D formulation

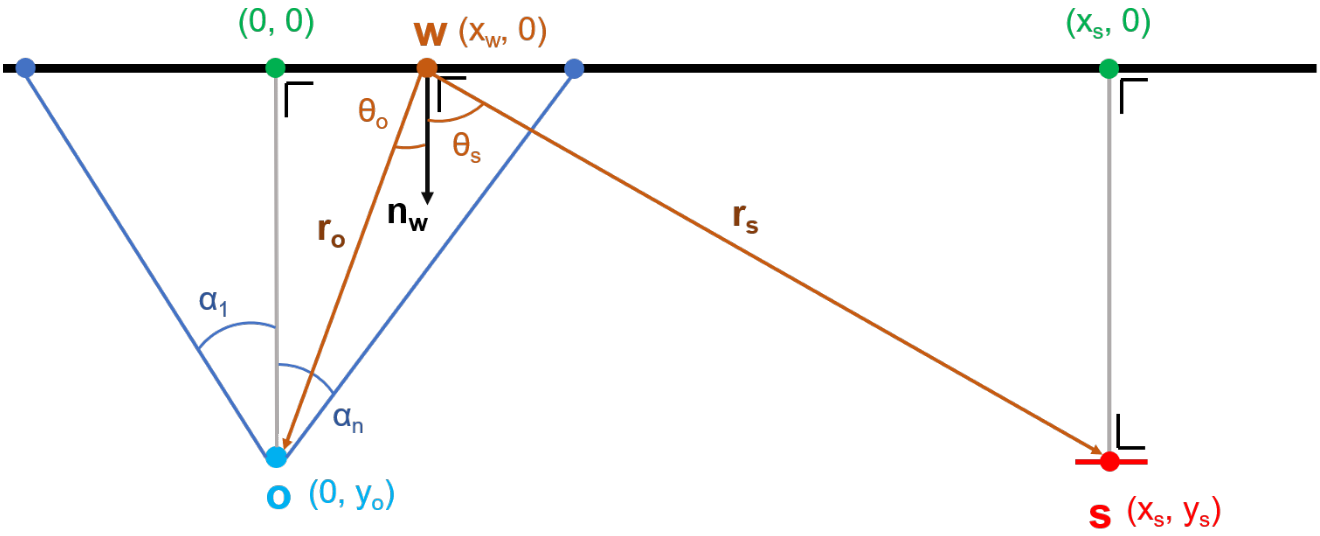

Figure 2.8: Geometrical configuration (2D)

Notice the sign change in from $L_e(-\boldsymbol{\omega_s})$ to $L_e(\theta_s)$.

To conclude this Section, let us formulate the rendering problem in a more formal way :

Direct problem : Rendering reflection on a wall : Given the following elements :

- The position of an observer $\boldsymbol{o}$, placed in front of the wall.
- The BRDF on the wall, $\rho(\theta_s, \theta_o)$
- The position of the light source $\boldsymbol{s}$, and the emitted light intensity $L_e(\theta_s)$.

Find $L_r(\theta_o)$, the intensity of the light reflected by all the points of the wall towards the observer, using (2.11)

As mentioned in the introduction, the inverse problem we aim to solve consists in finding the BRDF on the wall and the intensity of the light emitted by the source using $L_r(\theta_o)$. Depending on the resolution method, additional quantities or assumptions might be required ; we will detail them in the two following chapters, each dedicated to one resolution method.

2.2 Reference Rendering Algorithm

In this Section, we construct and describe the algorithm that provides us with a reflection pattern, given a BRDF for the wall and the light distribution emitted by the light source. The output of this algorithm will fulfill several roles :

- Be used as an input for the (blind) Deconvolution and De-rendering algorithms that will be detailed in later sections.

Notation	Meaning
M	Number of points of the light source. $M \in \mathbb{N}$
N	Number of points on the wall contained within the camera field of view. $N \in \mathbb{N}$
$\boldsymbol{s}_i$	Position of the i^{th} point of the light source. $\boldsymbol{s}_i \in \mathbb{R}^2$
$\boldsymbol{w}_j$	Position of the j^{th} point on the wall contained within the camera field of view. $\boldsymbol{w}_j \in \mathbb{R}^2$
Ω_W	Finite set containing all the points on the wall within the camera field of view.
Ω_S	Finite set containing all the points of the light source.
v_i	Light intensity of the i^{th} point of the light source. $v_i \in [0, 1]$
$\theta_{s,i}$	Angle between $\boldsymbol{n_w}$ and the i^{th} point of the light source. $\theta_{s,i} \in]-\pi/2, \pi/2[$

Table 2.3: Additional notations for discretization

- Be used as a reference against other reflection models (approximation of the rendering equation by a convolution section 3.1, expression of the rendering equation as matrix product in section 4.1).

2.2.1 Problem Discretization

The first step towards the construction of our algorithm is to discretize the reflection model established in (2.11). The observer is already modelled as a single point ; only the wall, the light source, and the integration half-circle Ω_{θ_o} remain to be discretized. To do so, we introduce additional notations in table 2.3.

The "wall" is now modelled by a finite set of T points $\boldsymbol{w}_j$, which are linearly spaced in an interval bounded by the intersections of the two boundaries of the camera field of view. The source is also modelled by a finite set of τ points s_i emitting light isotropically with an intensity v_i.

As previously stated, for a given $\boldsymbol{w}_j$, there is a bijection between the inbound angles $\theta_{s,i}$ and their corresponding source points $\boldsymbol{s}_i$. As such, discretizing the half-circle integration domain amounts to compute the sum of incoming light intensities for each source point $\boldsymbol{s}_i$.

We can thus write the discretized version of the simplified rendering equation described by (2.11) :

$$L_r(\theta_o) = \sum_{i=0}^{M} \rho(\theta_{s,i}, \theta_o) L_e(\theta_{s,i}) \cos \theta_{s,i}. \tag{2.12}$$

To further illustrate this discretization step, we provide a slightly modified version of Figure 2.8 on Figure 2.9, which accounts for the discretization.

2.2.2 Algorithm Description

A formal description of the algorithm is given by 1. As mentioned in the "preparation" block, two sets must be built before proceeding with the computation of the reflected light intensity :

- Ω_S, the set of M points $\boldsymbol{s}_i$ of the light source.
- Ω_W, the set of N points $\boldsymbol{w}_j$ seen by the observer.

Building Ω_S is trivial, but we would like to shed some like on the parameters used to build the light source. Given the width of the light source (the width of the "card" mentioned in the introduction), the position of its center, and the desired number of points M, one still needs to compute the position of the said points. This is made easier by our assumption that the light source is always parallel to the wall. As such, Ω_S is a vector of linearly spaced positions located in an interval

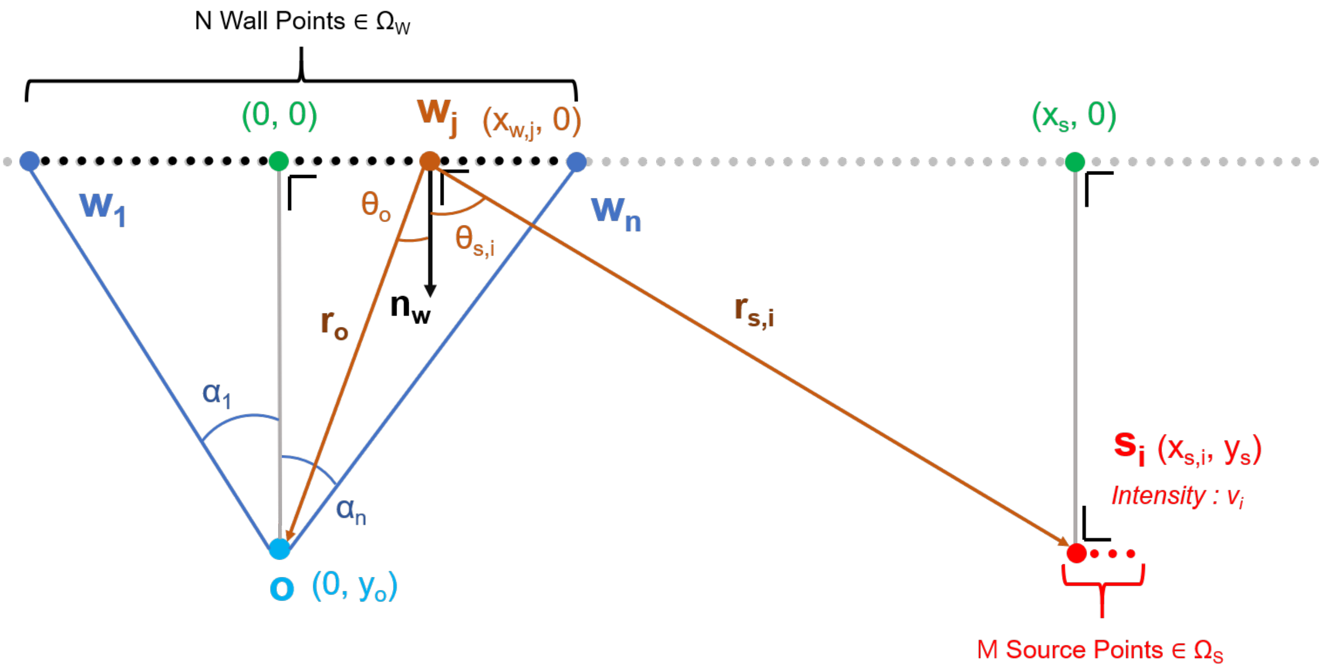

Figure 2.9: Geometrical configuration (2D, discrete)

Algorithm 1: Reference rendering algorithm. For each point of the wall $\boldsymbol{w}_j$, compute the outbound angle θ_o and the inbound angles $\theta_{s,i}$ linked with each point $\boldsymbol{s}_i$. Then, compute the reflected light intensity using (2.12)

Preparation:

Build Ω_S, the set of M points $\boldsymbol{s}_i$ of the light source.
Build Ω_W, the set of N points $\boldsymbol{w}_j$ seen by the observer.

Results:

$L_r(\theta_o)$, the intensity of the light reflected by each of the N points of the wall seen by the observer.

Algorithm:

foreach $\boldsymbol{w}_j \in \Omega_W$ **do**
  $\boldsymbol{\omega_o} \leftarrow \boldsymbol{o} - \boldsymbol{w}_j$
  `// Angle between the wall normal and the outbound direction`
  $\theta_o \leftarrow \mathrm{atan2}(n_{w,y}, n_{w,x}) - \mathrm{atan2}(\omega_{o,y}, \omega_{o,x})$
  $L_r(\theta_o) \leftarrow 0$ **foreach** $\boldsymbol{s}_i \in \Omega_S$ **do**
    $\boldsymbol{\omega_s} \leftarrow \boldsymbol{s}_i - \boldsymbol{w}_j$
    `// Angle between the wall normal and the inbound direction`
    $\theta_{s,i} \leftarrow \mathrm{atan2}(n_{w,y}, n_{w,x}) - \mathrm{atan2}(\omega_{s,y}, \omega_{s,x})$
    $L_r(\theta_o) \leftarrow L_r(\theta_o) + \rho(\theta_{s,i}, \theta_o)\, v_i\, \cos\theta_{s,i}$
  end
end

---

defined by the width of the source and its central position.

To build Ω_W, we make the assumption that the points of the wall seen by the observer are linearly spaced. Given a camera with a grid of linearly spaced photo-sensors, this assumption is true if the following hypothesis is verified :

Hypothesis 6 - Camera plane : The camera plane is parallel to the wall :

$$-\alpha_n = \alpha_1$$

This camera model greatly simplifies further comparisons and developments (especially related to convolutions). We will relax hypothesis 6 and detail what happens when we rotate the observer in order to acquire several "images" of the wall in Section 4.1.1.

As such, Ω_W is a set of linearly based positions located in an interval comprised between the two intersections of the wall with the boundaries of the observer field of view. Our conventions allow us to compute the position of these intersections as follows (y_o is the distance between the observer and the wall) :

$$\boldsymbol{w}_1 = (y_o \tan\alpha_1, 0) \qquad \boldsymbol{w}_n = (y_o \tan\alpha_n, 0). \tag{2.13}$$

Once Ω_W and Ω_S have been obtained, the rendering algorithm computes the inbound/outbound directions and angles in order to directly implement (2.12).

2.2.3 Numerical Experiments

First, let us detail the parameters common to the two renders that will be showcased in this section:

- $\boldsymbol{o}$ and $\boldsymbol{s}$ are located at the same position, $(0, -5)$.
- The field of view of the observer is such that $-\alpha_n = \alpha_1 = \pi/4$.
- $N = 500$, $M = 2$

We start by illustrating the trivial case, which corresponds to the mirror case. The BRDF is such that :

$$\rho_{\text{mirror}}(\theta_{s,i}, \theta_o) = \delta(\theta_o + \theta_{s,i}) \tag{2.14}$$

δ denotes a Dirac delta. Since the BRDF is only a function of the angle sum, we can plot it as a 1D signal. Fig. 2.10 show, from left to right, the source, the BRDF expressed as a function of the angles sum (and not difference as written on the figure), and the resulting render. We simply observe that the render corresponds to the projection of the source on the wall.

We use the second example to introduce the BRDF that we will use for all our future experiments. In the angular domain, this BRDF is a train of 4 gaussians with $\sigma = 1$. Its expression is :

$$\rho_{\text{test}}(\theta_{s,i}, \theta_o) = f_{-\frac{\pi}{12},1}(\theta_o + \theta_{s,i}) + f_{\frac{\pi}{12},1}(\theta_o + \theta_{s,i}) + f_{-\frac{\pi}{6},1}(\theta_o + \theta_{s,i}) + f_{\frac{\pi}{6},1}(\theta_o + \theta_{s,i}) \tag{2.15}$$

In the expression above, $f_{\mu,\sigma}$ is the normalized gaussian function with mean μ and standard deviation σ.

Physically, such a BRDF copies and blurs the light source 4 times (like a blurring retro-reflector with 4 faces). On Fig. 2.11, we rendered the scene using the open-source 3D modelling suite Blender. For the sake of clarity, we programmed the source so that it is invisible to the camera.

On Fig. 2.12, we showcase the render produced by our own algorithm for ρ_{test}. The BRDF is "copied" around the two deltas in a way that is reminiscent of convolution, which will be the focus of the next chapter.

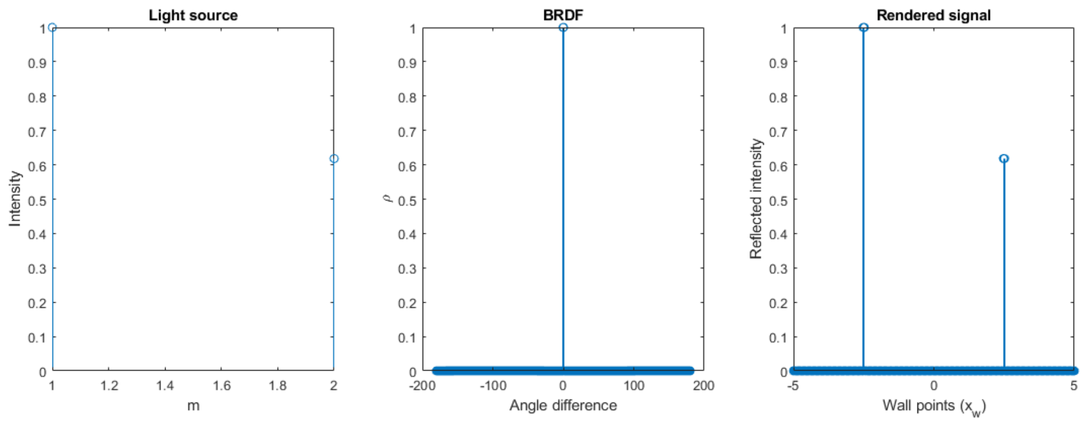

Figure 2.10: Render using Algorithm 1 (mirror case)

Figure 2.11: Real world render of our test BRDF

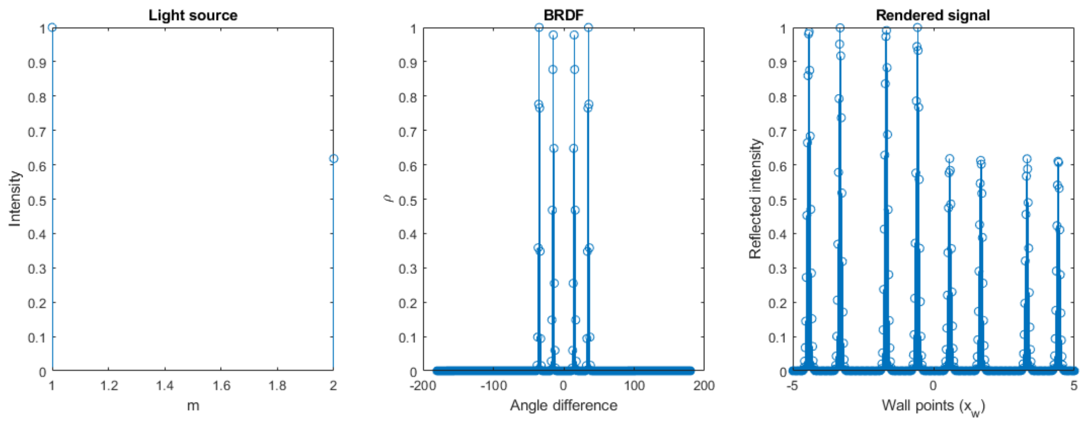

Figure 2.12: Render using Algorithm 1 (test BRDF case)

Chapter 3

From De-Rendering to Deconvolution

In this chapter, we show, under some assumptions, the rendering equation is equivalent to a convolution. Then, we present the deconvolution problem and try to solve the de-rendering problem using tools dedicated to deconvolution.

3.1 Reformulation of Rendering as a Convolution

3.1.1 Rendering as a Convolution on the Sphere

Let us start our reasoning from (2.12), the 2D discrete formulation of the rendering equation :

$$L_r(\theta_o) = \sum_{i=0}^{M} \rho(\theta_{s,i}, \theta_o) L_e(\theta_{s,i}) \cos\theta_{s,i}.$$

Our goal is to reach an expression of the form

$$(f * h)[n] := \sum_{\Omega} f[n-m]h[m]. \tag{3.1}$$

which is the definition of discrete 1D convolution.

To reach such an expression, we need to add two assumptions. The first one aims at factoring the cosine factor out of the summation. We can indeed show that, if the light source is sufficiently far from the wall, $\theta_{s,i}$ is close to constant for all source points $\boldsymbol{s}_i$. This is analog to a far-field approximation, where propagation paths from a source located at a large distance of an object are parallel to each other when reaching the surface of that object.

We already have all the expressions required to complete this intuition with a mathematical reasoning. Given two points $\boldsymbol{s}$ and $\boldsymbol{s}'$, located at random on the light source. Our goal is to show that the difference between θ_s and θ'_s is inversely proportional to the distance of the source from the wall y_s. We can reuse (2.5) to write :

$$\boldsymbol{\omega_s} = \frac{(\boldsymbol{s} - \boldsymbol{w})\cos\theta_s}{y_s} \qquad \boldsymbol{\omega'_s} = \frac{(\boldsymbol{s}' - \boldsymbol{w})\cos\theta'_s}{y_s}. \tag{3.2}$$

The difference between $\boldsymbol{\omega_s}$ and $\boldsymbol{\omega'_s}$, which is, in 2D, morally equivalent to the difference between θ_s and θ'_s, can thus be expressed as being inversely proportional to y_s :

$$\boldsymbol{\omega'_s} - \boldsymbol{\omega_s} = \frac{(\boldsymbol{s}' - \boldsymbol{w})\cos\theta'_s - (\boldsymbol{s} - \boldsymbol{w})\cos\theta_s}{y_s}. \tag{3.3}$$

Intuitively, the difference between the inbound directions $\boldsymbol{\omega'_s}$ and $\boldsymbol{\omega_s}$ can also be reduced by reducing the width of the source, and thus the spacing between its points. All in all, we obtain the following hypothesis :

Hypothesis 7 - "Far-field" approximation : The distance of the source from the wall y_s is large enough so that, for a given wall point $\boldsymbol{w}$, the difference between the $\theta_{s,i}$ linked to each source point $\boldsymbol{s}_i$ is negligible.

Consequently, the cosine factor can be factored out of the summation as a constant factor (denoted by $Q(\theta_o)$) that only depends on the position of the point on the wall, i.e. θ_o :

$$L_r(\theta_o) = Q(\theta_o) \sum_{i=0}^{M} \rho(\theta_{s,i}, \theta_o) L_e(\theta_{s,i}). \tag{3.4}$$

The second assumption is arbitrary, and is that the BRDF is a function of the difference between the outbound angle and the inbound angle :

Hypothesis 8 - BRDF function of the angle difference : Given the outbound angle θ_o and the inbound angle linked with one source element $\theta_{s,i}$, the BRDF is a function of the difference between these angles, i.e. :

$$\rho(\theta_{s,i}, \theta_o) = \rho(\theta_o - \theta_{s,i}).$$

Specular and glossy reflectors satisfy this condition. As a consequence, formulating the de-rendering problem as a convolution is only possible if the wall is a specular or a glossy reflector (like a blurry mirror or a retro-reflector).

We finally obtain the following expression for the rendering equation, which is equivalent to a convolution multiplied by a constant factor :

$$L_r(\theta_o) = Q(\theta_o) \sum_{i=0}^{M} \rho(\theta_o - \theta_{s,i}) L_e(\theta_{s,i}) = Q(\theta_o)(\rho * L_e)(\theta_o). \tag{3.5}$$

In other words, for each wall point, the light intensity reflected in all possible outbound directions is equal to the convolution of the map (over all possible inbound directions) of the inbound light intensity with a map of the BRDF over all possible angle differences. To obtain results similar to those acquired in 2.2.3, one needs to perform that convolution on a hemisphere (on a half-circle in 2D) W times (for each point $\boldsymbol{w}$), then compute the θ_o corresponding to each point $\boldsymbol{w}$. The final render is then obtained by taking one value from each convolution : the element corresponding to the computed θ_o.

On Fig. 3.1, we compare the render obtained with algorithm 1 to the render obtained with the spherical convolution, for a random source with $M = 2$ elements and the BRDF defined in (2.15). After viesual inspection, the renders are close. However, given the way intensities are sampled around the integration half-circle (for a given wall point, the half-circle point with the closest θ_o is chosen), some values are slightly different.

3.1.2 Towards 1D Convolution

The formulation above presents various disadvantages :

- To obtain satisfying results, one must convolve two lengthy vectors (for a resolution of 1 degree, $L_e(\theta_{s,i}$ already has 180 points) as many times as there are points on the wall.
- The rendering process results in a vector that only contains one fraction of the information of each spherical convolution (only one point). As we will explain later, deconvolution is

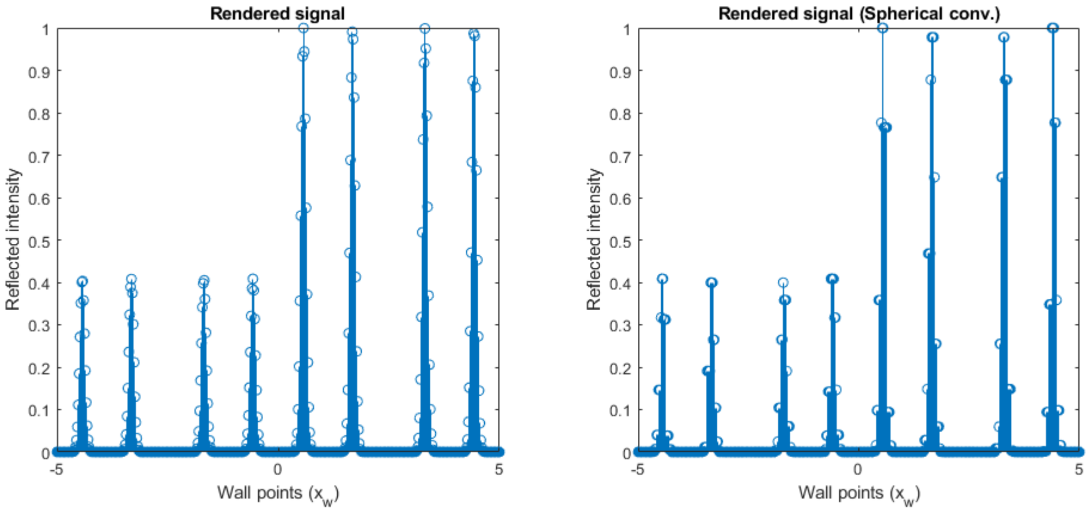

Figure 3.1: Render using Spherical convolution

already an ill-posed problem. Solving for both the BRDF and the source when the only data at our disposal is a fraction of each convolution of that BRDF and source amounts to fighting a losing battle[1].

As such, we attempt to formulate the rendering equation as a single convolution that directly results in the reflection vector $L_r(\theta_o)$ seen by the camera. According to the geometrical configuration defined in Section 2.1.1, doing so results in a 1D convolution.

The intuition behind our reasoning is directly linked to the title of this thesis, "Turning a wall into a mirror". Indeed, reflecting an image in a mirror is equivalent to convolving it with a single Dirac delta : the reflection process outputs the exact projection of the input image on the wall.

Thus, in the mirror case, rendering is simply equivalent to convolving the light distribution of the source by the mirror BRDF, a centered delta. Moreover, shifting the reflection horizontally by a distance d is equivalent to changing the BRDF as follows :

$$\delta(x) \leftarrow \delta(x - \text{atan}(d/y_o)). \tag{3.6}$$

Moreover, rendering with such a BRDF is equivalent to convolve the render obtained with a normal mirror (denoted $L_{r,\text{mirror}}(\theta_o)$) by a delta located in d, which is the "projection" of the BRDF on the wall (denoted $\rho_{\text{wall}}(\theta_o)$).

If we extend that reasoning to BRDFs that satisfy hypothesis 7 and 8 we obtain the following expression :

$$L_r(\theta_o) = Q(\theta_o) \sum_{i=0}^{M} \rho(\theta_o - \theta_{s,i}) L_e(\theta_{s,i}) \approx Q(\theta_o)(\rho_{\text{wall}} * L_{r,\text{mirror}})(\theta_o). \tag{3.7}$$

$L_{r,\text{mirror}}$ can either be obtained using the reference rendering algorithm described in the previous chapter. As for $\rho_{\text{wall}}(\theta_o)$, it is indeed a function of θ_o. To "project" the BRDF on the wall space :

- First, we compute the BRDF in the angular domain, for all possible angle differences.
- Second, we compute the values θ_o corresponding to all the wall points seen by the observer.

[1]However, after developing the algorithm in Chapter 4, we realized a solution based on the spherical convolution might produce acceptable results (cf. conclusion)

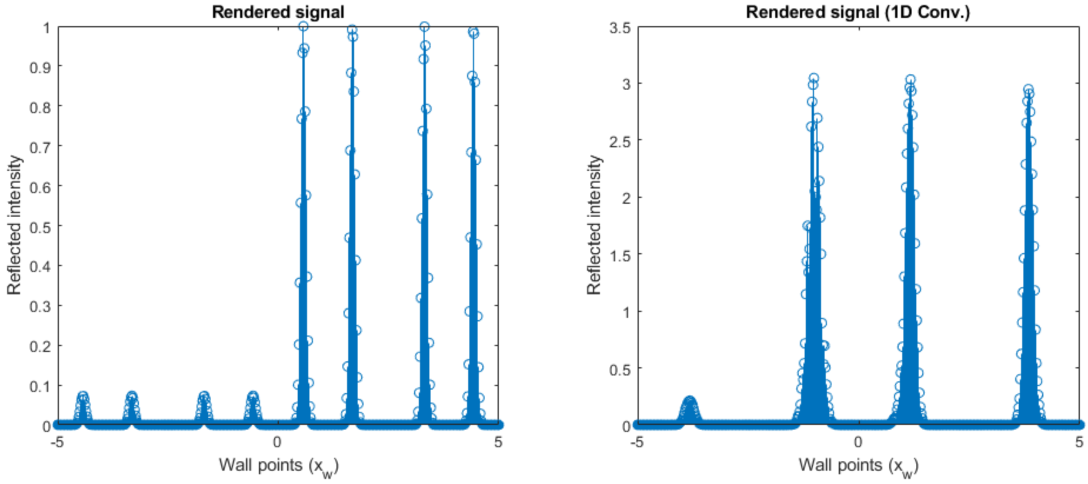

Figure 3.2: Render using 1D convolution (source too close)

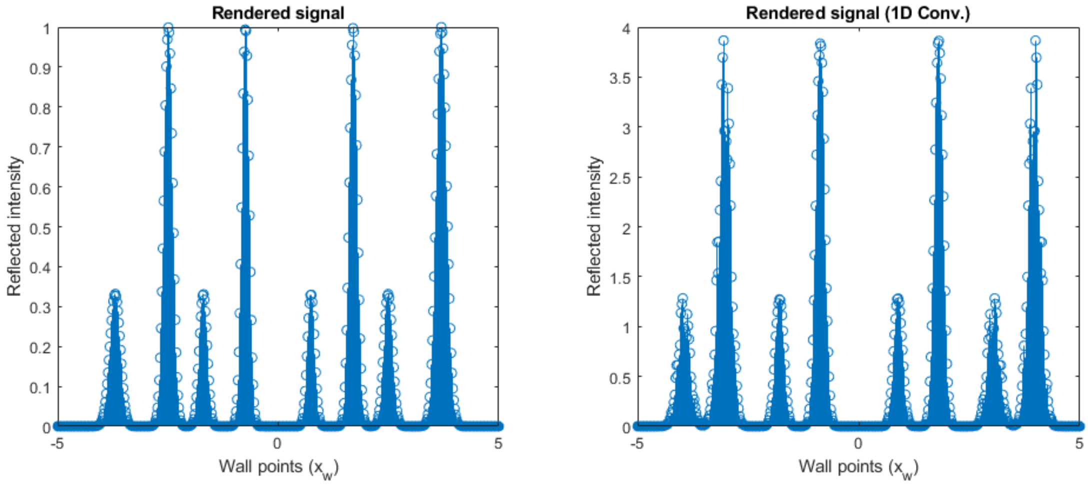

Figure 3.3: Render using 1D convolution

- Then, we assign to each wall point the BRDF value corresponding to an angle difference equal to the θ_o linked to that wall point.

The main advantage of that reasoning is that rendering becomes a convolution of the BRDF with the render obtained in the mirror case. As such, conceptually, solving the deconvolution problem allows for recovery of the render obtained in the mirror case : we are truly "turning a wall into a mirror".

Fig. 3.3 and 3.2 compare the render obtained with Algorithm 1 to the render obtained using 1D convolution. In the case of Fig. 3.2, the 7th hypothesis is not valid (as the source is too close to the wall : $\boldsymbol{s} = (0, -5)$), and the render significantly deviates from the "correct" one. On Fig. 3.3, the distance of the source to the wall has been increased tenfold ($\boldsymbol{s} = (0, -50)$), and two renders are close, even the the position of the gaussians is still slightly off on the convolution product.

3.2 The Deconvolution Problem

3.2.1 Problem Setup and Ill-Posedness

To formulate the deconvolution problem, one must first define the forward problem, i.e. the convolution. We rewrite (3.1) :

$$g[n] = (f * h)[n] := \sum_{\Omega} f[n-m]h[m]. \tag{3.8}$$

Convolution models the impact of a LTI system on a signal. In the expression above, f is the input, h is the impulse response of the considered LTI system, and g is the output. The deconvolution problem [24] consists in finding f given h and g, or to find h given f and g.

The first tool that comes to mind is the convolution theorem. The forward problem can be written as follows in the frequency domain[2] :

$$G[k] = F[k]H[k]. \tag{3.9}$$

We will illustrate why this direct method is not used in practice. Given y and g, we will try to find f using the convolution theorem. Naturally, in the Fourier domain, we have[3] :

$$F[k] = \frac{G[k]}{H[k]}. \tag{3.10}$$

However, problems linked to the presence of noise, however small, in $g(t)$ arise : $g(t)$ might have spectral components where $H(\omega)$ is really low or even zero, which might lead to noise amplification and aberrant values of $F(\omega)$, and thus impair the estimate of $f(t)$. Moreover, in higher dimensions, $H(\omega)$ is not trivially invertible (matrix badly conditioned). It is also worth noting that exact deconvolution is often impossible due to the presence of additive noise in the output y. The deconvolution problem is thus ill-posed[4] [25].

However, approximating a solution to the problem is still possible [26]. Regularization methods provide stable approximations to inverse filters. The Tikhonov regularization discussed below is one of the most ubiquitous regularization methods.

3.2.2 Single-Channel Deconvolution Algorithms

Tikhonov Regularization

In order to approximate a solution to the deconvolution, which is an ill-posed problem, adding information is useful. This process is called regularization. One of the most common regularization methods is the Tikhonov regularization [27].

Let us rewrite (3.8), replacing the convolution operator a multiplication with a Toeplitz matrix (H is the impulse response) to represent our convolution operator :

$$g = Hf. \tag{3.11}$$

The Tikhonov regularization consists in solving the following variational problem :

$$\hat{f}_\alpha = \arg\min_f ||g - Hf||_2^2 + \alpha||f||_2^2. \tag{3.12}$$

[2] here, k is used as a variable in the frequency domain.
[3] It is worth noting that the division is, in this case, element-wise.
[4] The solution of a well-posed problem exists, is unique, and is stable.

In the expression above, a regularization term has been added to the classical least-squares problem. A derivation is available here [28].

$$\hat{f}_\alpha = (H^T H + \alpha I)^{-1} H^T g. \tag{3.13}$$

There are many different approaches to determine the value of α. We usually proceed by trial and error.

To provide an illustration for the purpose of deconvolution in the framework of image processing, We implemented this method in MATLAB. The convolution of an image f with a gaussian point-spread function h results in a blurred image g. Given this double-blocked Toeplitz matrix H corresponding to this PSF (see (3.11)) and the blurred image, it is possible to estimate the original image using (3.13). The results are given by the figures below.

Figure 3.4: Original image

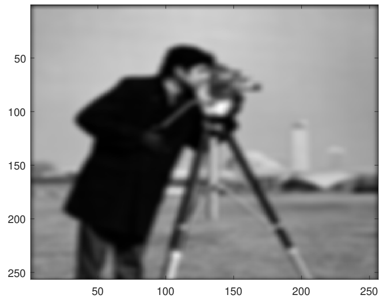

Figure 3.5: Blurred image

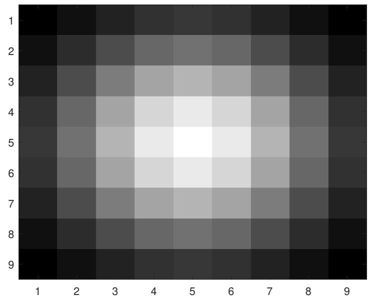

Figure 3.6: Point Spread Function

Figure 3.7: Image estimate, $\alpha = 0.0001$

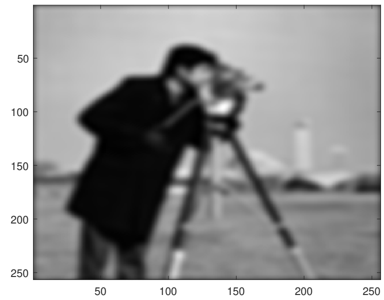

Figure 3.8: Image estimate, $\alpha = 1$

The impact of the value of α is illustrated. If α is too big, the resulting image will be too blurred. Inversely, if α is too small, the problem is closer to the original, ill-posed, deconvolution, and artifacts arise. Moreover, it is important to input the original, uncropped (cropping is often required to to the black contour generated by the convolution), blurred image into the deconvolution algorithm ; else, information loss could arise.

Lucy-Richardson deconvolution

The Lucy-Richardson deconvolution method is an iterative algorithm that was originally designed to restore a signal corrupted by Poisson-distributed shot noise. It belongs to a class of maximum-likelihood algorithms.

Let us take g, the blurred signal, as an initial guess of f. h is known and the iterative scheme is as follows [29] :

$$\hat{f}_{i+1} = \hat{f}_i \cdot \left(h \star \frac{g}{\hat{f}_i * h} \right). \tag{3.14}$$

In the expression above, $\star$ denotes the cross-correlation operator.

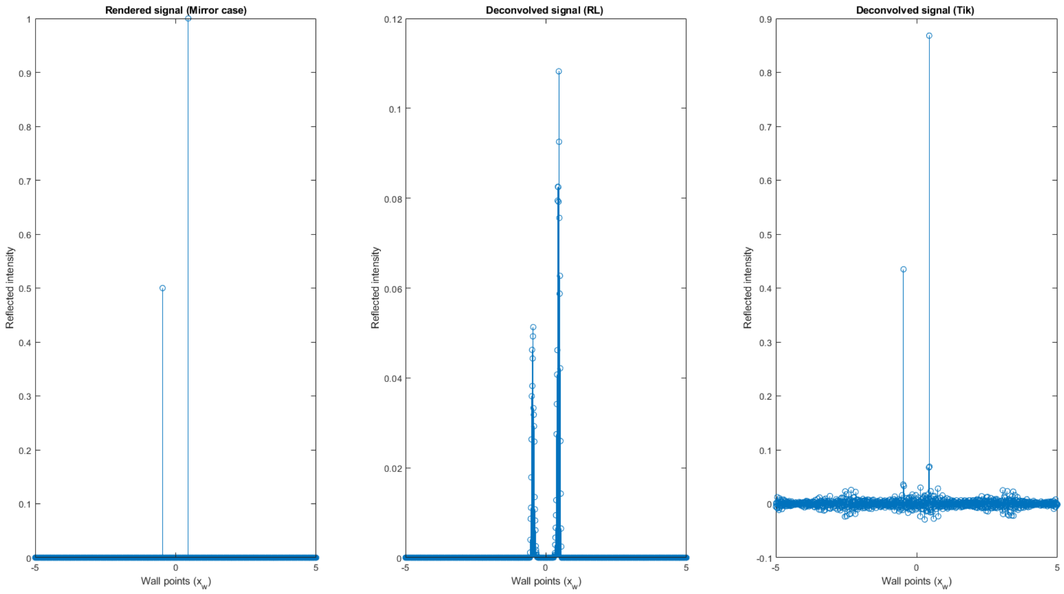

Figure 3.9: Non-blind de-rendering using 1D convolution (RL : 10 iterations, Tik : $\alpha = 1$)

Numerical experiments

Let us now apply these algorithms to the de-rendering problem. We assume that the projection of the BRDF on the wall, ρ_{wall}, is known. As explained in Section 3.1.2, deconvolving ρ_{wall} from $L_r(\theta_o)$ should yield $L_{r,\text{mirror}}(\theta_o)$, the light reflected in the mirror case. The test configuration is the same as in Section 2.2.3, except that $\boldsymbol{s}$ is located further from the wall ($\boldsymbol{s}$ = (0, -50)).

The results are shown on Fig. 3.9. For both the Tikhonov regularization and the Richardson-Lucy algorithms, the relative intensity as well as the delta position is well recovered. The Tikhonov regularization provides thinner impulses, at the expense of artefacts.

3.2.3 Single-channel Blind Deconvolution Algorithms

So far, we have assumed that both the quantities g and h where known. If h is also unknown, the problem to solve is called "blind deconvolution", and consists in simultaneously retrieving f and h, given the output g and assumptions on the impulse response h and the input f.

Existing approaches can be separated in two classes [30] :

- Those which first find an estimate of h, before using it with traditional deconvolution methods, such as the ones cited above.
- Those which simultaneously estimate f and h, which leads to more complex algorithms.

Iterative Blind Deconvolution

In what follows, we will describe the iterative blind deconvolution (IBD) method [31]. This method falls in the second class of deconvolution techniques. It consists in alternating inverse filters while imposing conditions linked to the support of the target quantities. We describe the algorithm in great detail, as alternating minimization techniques are core to this thesis.

In the framework of image processing, deconvolution can be seen as performing a deblurring operation. The blur results from the true image f being convolved with a point spread function

(PSF), that we will call h for the sake of consistency with the developments above. The point spread function is equivalent to the impulse response of an imaging system : it is its response to a point light source.

This method requires the assumptions on f and h are non-negative and of known, finite support. Before starting the iterative method, proper initial guesses for f and h must be provided. For f, the blurred image is used. For h, we make the assumption that the PSF is akin to a delta, or, in the 2D case, a single white pixel in the middle of a black image. Then, the following steps are repeated until satisfactory estimates are obtained for both f and h :

- Compute $\hat{F}$, the 2D Fourier Transform of the original image estimate, $\hat{f}$. If needed, use zero-padding so that $\hat{F}$ has the same dimensions as G, the 2D Fourier Transform of the blurred image.
- Compute $\tilde{H}$, the Fourier Transform of $\tilde{h}$. Its is given by the following formula :
$$\tilde{H} = \frac{G\hat{F}^*}{|\hat{F}|^2 + \alpha/|\hat{H}|^2}. \tag{3.15}$$
All operators are element-wise, $\hat{F}^*$ denotes the conjugate matrix of $\hat{F}$, alpha is an arbitrary factor determined in a similar fashion as the Tikhonov factor.
- Compute $\tilde{h}$. As stated above, it is obtained by computing the inverse Fourier Transform of $\tilde{H}$.
- Compute $\hat{h}$, the estimate of the original PSF. It is obtained by enforcing the assumptions defined above, i.e. by taking the absolute value of all the values of $\tilde{h}$ inside the (known) support of h, and by setting to 0 all the values outside of the support.
- Compute $\hat{H}$, the 2D Fourier Transform of the original image estimate, $\hat{h}$. Again, use zero-padding so that $\hat{F}$ has the same dimensions as G.
- Compute $\tilde{F}$, the Fourier Transform of $\tilde{f}$. Its is given by the following formula, very similar to (3.15) :
$$\tilde{F} = \frac{G\hat{H}^*}{|\hat{H}|^2 + \alpha/|\hat{F}|^2}. \tag{3.16}$$
- Compute $\tilde{f}$. As stated above, it is obtained by computing the inverse Fourier Transform of $\tilde{F}$.
- Compute $\hat{f}$. Again, it is obtained by enforcing the assumptions defined above, i.e. by taking the absolute value of all the values of $\tilde{f}$ inside the (known) support of f, and by setting to 0 all the values outside of the support. Then, go back to the first step.

This method was implemented using MATLAB. The original image, the blurred image and the PSF used to perform the blurring are the same as those feature on figures 6, 7 and 8 respectively. After 100 iterations, we obtain the figures showcased below :

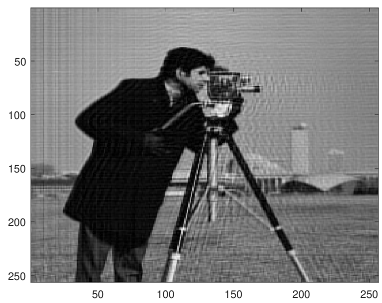

Figure 3.10: Image estimate, $\alpha = 10$

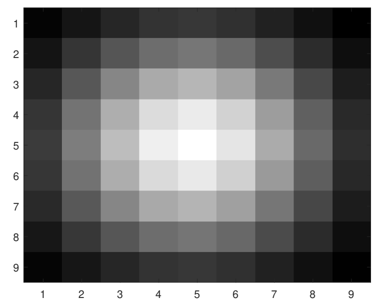

Figure 3.11: PSF estimate, $\alpha = 10$

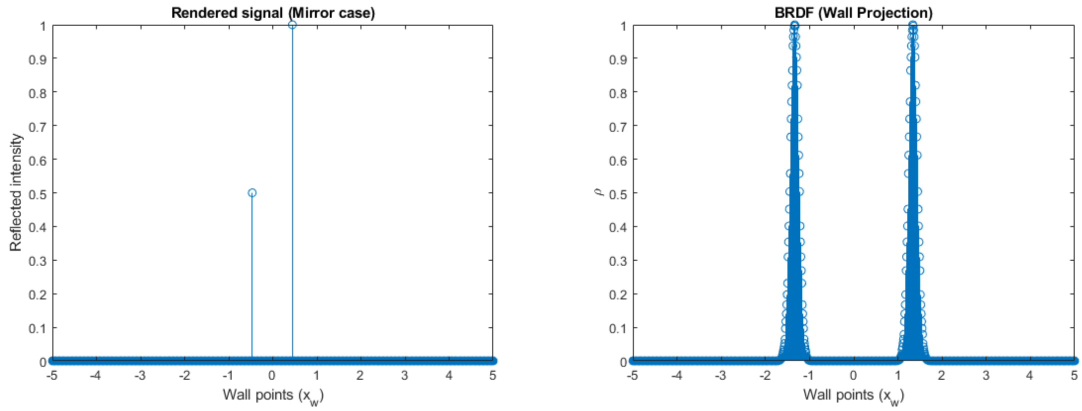

Figure 3.12: True quantities to be obtained by blind deconvolution

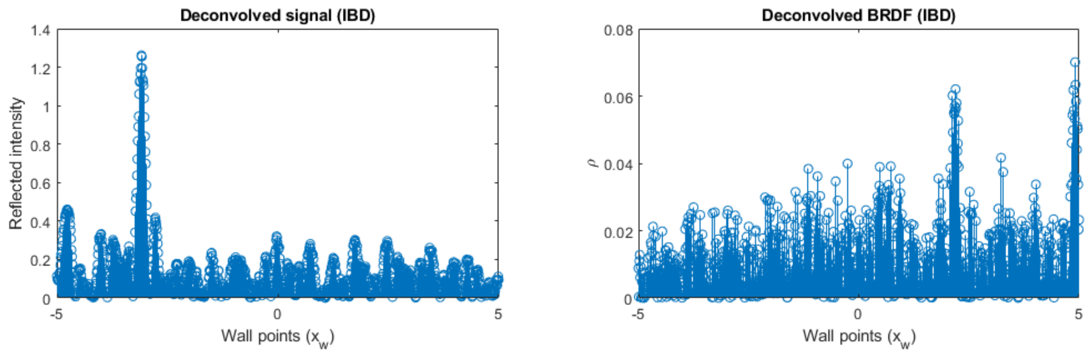

Figure 3.13: Blind de-rendering (IBD, 5 iterations, $\alpha = 0.001$)

As expected, the iterative blind deconvolution yields slightly worse results than those obtained with the Tikhonov regularization : there are more artifacts on the deblurred image, and the estimated PSF is slightly shifted on the left. However, given the low computational complexity of the algorithm, the results are satisfactory.

As detailed in [32], it is also possible to replace the alternating algorithm above by an alternating Richardson-Lucy deconvolution.

Numerical Experiments

We now attempt to solve the blind de-rendering problem using both the Iterative Blind Deconvolution algorithm (IBD) and its Richa rdson-Lucy variant (IRL). The geometrical configuration and the rendering parameters are identical to 3.2.2. The target quantities are shown on Fig. 3.12.

While IBD does not get even close to producing visually acceptable results (see Fig 3.13), IRL (see Fig 3.14) correctly estimates the relative intensity of the source, but the peaks are shifted and the resulting quantity is displayed as...the BRDF. The most important ambiguity of blind deconvolution (being able to distinguish f from h) is not even solved by these algorithms for our problem.

3.2.4 Multi-channel Blind Deconvolution Algorithms

As stated in the introduction, the goal of this thesis is to attempt to determine the conditions on the BRDF that allow for blind de-rendering for the geometrical configuration described in Section 2.1.1. As a starting point, we were given a recently developed blind deconvolution algorithm (Focused Blind Deconvolution [33]), which led to promising results in the field of seismic imaging. As such, attempting to formulate the de-rendering problem as a deconvolution was a logical first

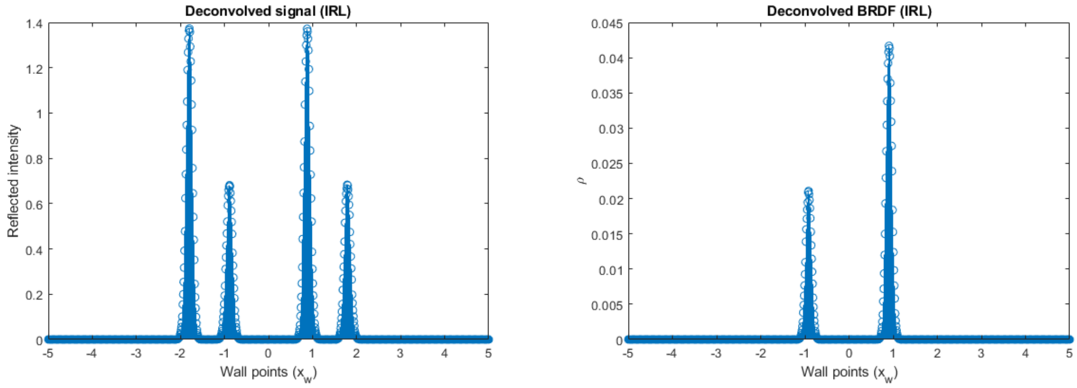

Figure 3.14: Blind de-rendering (IRL, 100 iterations)

step. As shown in the previous chapter, that approach even provided satisfying results in the non-blind case.

The relevance of Focused Blind Deconvolution became much more apparent when single-channel blind deconvolution techniques (see the dedicated section in the previous chapter) did not yield satisfying results for the inverse problem. We needed a multi-channel algorithm in order to leverage diversity to improve our estimations of the BRDF and the source.

Focused blind deconvolution formulates two assumptions on the channels and on their auto-correlations to help recover the channel matrix as well as the original signal. It is divided in two steps :

- The first one attempts to recover the auto-correlation of the signal and the cross-correlations of the channel, under the assumption that the channels are "maximally white", i.e. their the energy of their auto-correlation is focused near zero-lag.
- The second one attempts to recover the phase of the channels and the signal in order to rebuild them from their auto/cross-correlations. It relies on the assumption of the channels are "maximally front-loaded", i.e. their energy is focused around the zero-time of the channel seen by the closest sensor to the source.

Both these assumptions take the form of a regularizing term added the the objective function to minimize. In practice, each step corresponds to an alternative minimization of the cost-function linked to that step, for each quantity (channels, then source).

This method seemed promising at first, especially the first step, which aims at removing the indeterminacy (lack of uniqueness of the solution) that characterizes another well-known multichannel blind deconvolution method, the least-squares blind deconvolution.

However, three main issues arised when trying to adapt that method to our specific case :

- The algorithm relies on interferometric quantities thanks to the following formula :

$$g_{ij}(t) = (g_i \star g_j)(t) = (f_a * h_{ij})(t).$$

  That result of seismic interferometry states that the cross-correlation of the output of two channels is equivalent to the convolution of the autocorrelation of the source signal and the cross-correlation of the linked channels. While we managed to verify that expression in the case of a theoretical 1D convolution, we did not succeed to obtain good results with the quantities used in (3.7).

- The algorithm requires that "*the duration length of the unknown impulse response should be much briefer than that of the channel outputs*". While M, the size of the support of the source, is much smaller than N, the size of the support of the wall, things change after the developments required to obtain (3.7). In that equation, we are indeed convolving two vectors of the same length N.

- The "maximally front-loaded" condition is not verified : if we vary the source k times and assume that the k $L_{r,\text{mirror}}$ correspond to the k channels, the energy can be spread out arbitrarily, depending on the number of source points or on the position of the source. If we move the observer k times and assume that, this time, the k ρ_{wall} are the channels, even our default test BRDF, a train of four gaussians, does not satisfy the condition.

As such, we did not manage to adapt Focused Blind Deconvolution to solve the inverse problem. At that point, we decided to leverage the linear structure of the rendering equation, with much better results (cf. Chapter 4). Nonetheless, this venture into blind deconvolution allowed us to get much more familiar with inverse problems. As such, the alternating structure of the algorithm presented in 4.2 is inspired by both Focused Blind Deconvolution and Iterative Blind Deconvolution techniques.

Chapter 4

Blind De-Rendering : Algorithm and Performance

In this chapter, a build a de-rendering algorithm based on our knowledge of the wall-source geometrical configuration. The performance of this algorithm is then assessed. It is worth noting that our reasoning starts directly where Chapter 2 ends : as such, we do not take into account the assumptions made in Chapter 3.

4.1 Reformulation of Rendering as a Linear System

The de-rendering algorithm presented in this section relies on the formulation of the rendering equation as a linear system, which stems from a matrix product between a matrix containing all the information about the BRDF, and a vector containing the light distribution of the source.

4.1.1 Camera Model

As mentioned in the introduction above, solving the inverse problem in the blind case (both the BRDF and the source light distribution unknown) requires diversity to produce satisfying results. Creating that diversity can be done in two ways :

- Varying the light intensity of the source (but not its position).
- Moving the observer in order to get different snapshots of the wall.

The first method makes sense, from an intuitive point of view : varying the light intensity of the source while keeping the BRDF identical provides the opportunity to better characterize the BRDF, in a way that is reminiscent of multipath channel estimation in wireless communications.

The second method can be refined using the assumptions made in Section 2.1. In (2.12), the BRDF is only a function of θ_o and $\theta_{s,i}$. As such, we do not need to move the observer to get diversity : rotating it already provides different snapshots of the BRDF.

Moreover, for a given rotation step, the BRDF will change in a predictable way. The developments below aim to characterize the evolution of the BRDF with the rotation of the observer. However, doing so requires a better characterization of our rotating camera.

To do so, we introduce some notations relative to the characteristics of the camera in Table 4.1. To better illustrate the usage of these notations, we showcase them on Figure 4.1.
We want our observer to rotate around its position : to do so, we must relax hypothesis 6. Since we do not make the assumption that the photo-sensor P is parallel to the wall, the wall points $\boldsymbol{w}_j$

Notation	Meaning
$\boldsymbol{p}_j$	Position of the j^{th} point on the photo-sensor. $\boldsymbol{p}_j \in \mathbb{R}^2$
P	Finite set containing all the linearly spaced points of the camera photo-sensor.
α_j	Angle between $-\boldsymbol{n_w}$ and the vector going from the observer to the point $\boldsymbol{w}_j$. $\alpha_j \in]-\pi/2; \pi/2[$
γ_j	Angle difference between α_1 and α_j : $\gamma_j = \alpha_1$ - α_j. $\gamma_j \in]-\pi; \pi[$
β	Half of the horizontal field of view of the camera : $\beta = (\alpha_n - \alpha_1)/2$. $\beta \in]0, \pi/2[$
ϕ	Rotation angle. $\phi \in [0, 2\pi]$
l	Length of the photo-sensor. $l \in \mathbb{R}$
d	Distance of the photo-sensor to $\boldsymbol{o}$. $d \in \mathbb{R}$

Table 4.1: Additional notations for observer rotation

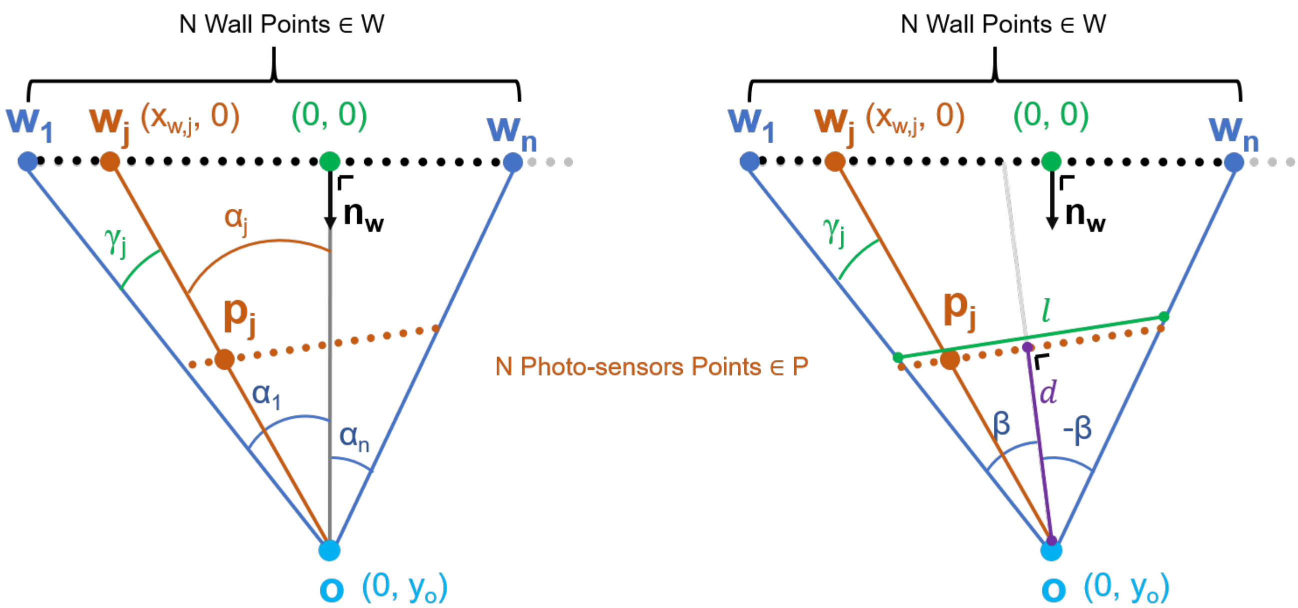

Figure 4.1: Camera geometrical configuration

are not linearly spaced. To obtain their position, we first need to compute the α_j relative to each point of the photo-sensor. To do so, we can compute γ_j as follows :

$$\tan(\beta - \gamma_j) = \frac{\left(\frac{l}{2} - \frac{l(j-1)}{N-1}\right)}{d} \qquad \Longleftrightarrow \qquad \gamma_j = \beta - \operatorname{atan} \frac{\left(\frac{l}{2} - \frac{l(j-1)}{N-1}\right)}{d}. \tag{4.1}$$

It is worth noting since l is a function of d and β, we can simply the expression of γ_j :

$$l = 2d \tan\beta \Longleftrightarrow \gamma_j = \beta - \operatorname{atan}\left(\tan\beta - \frac{2(j-1)\tan\beta}{N-1}\right). \tag{4.2}$$

We finally obtain the following expression for α_j :

$$\alpha_j = \alpha_1 - \gamma_j = \alpha_1 - \beta + \operatorname{atan}\left(\tan\beta - \frac{2(j-1)\tan\beta}{N-1}\right). \tag{4.3}$$

Reusing (2.13), we obtain the position of each wall point :

$$\boldsymbol{w}_j = (y_o \tan\alpha_j, 0). \tag{4.4}$$

If we apply the assumptions enumerated in Section 2.1, ρ is only a function of the difference between θ_o and $\theta_{s,i}$. Since the position of the source and the position of the observer are constant, these angles do not change when the observer rotates for a given wall point. What changes, however, is the set of wall points contained in the horizontal field of view of the observer.

Let us study the following configuration : we obtain the observer $\boldsymbol{o}'$ by rotating the observer $\boldsymbol{o}$ around the point $(0, y_o)$ by an angle ϕ. As a result, we have :

$$(\alpha'_1, ..., \alpha'_n) = (\alpha_1, ..., \alpha_n) + \phi. \tag{4.5}$$

Let us denote by Ω'_W the set of wall points seen by the observer after the rotation. If the rotation angle is small enough (i.e. if $|\phi| < 2|\beta|$), the position of η points $\boldsymbol{w}_j$ are still contained in the interval $[\boldsymbol{w}'_1, \boldsymbol{w}'_n]$. For further use in our derendering algorithm, we formulate the following problem :

Given the above camera model, the observers $\boldsymbol{o}$ and $\boldsymbol{o}'$, the rotation angle ϕ, for each point $\boldsymbol{w}_j$ contained in Ω_W, what is the closest point $\boldsymbol{w}'_k$ contained in W' ?

We solve this problem by assuming[1] that N is large enough so that we can assign the properties of the last η points of W to the first η points of Ω'_W :

$$\forall \epsilon > 0, \exists \delta > 0, N > \delta \Longrightarrow |\boldsymbol{w}'_k - \boldsymbol{w}_{n-\eta+k}| < \epsilon \qquad \boldsymbol{w}_j \in \Omega_W, \boldsymbol{w}'_k \in \Omega'_W, 1 \leq k \leq \eta. \tag{4.6}$$

As such, we can use our new camera model to render $\boldsymbol{o}'$ by only computing points that are outside the field of view of $\boldsymbol{o}$. This will come in handy when estimating the BRDF to solve the inverse problem.

4.1.2 BRDF Matrix Construction

This Section is dedicated to the formulation of the rendering equation as a linear system. As a reminder, we provide (2.12), the 2D rendering equation obtained after imposing the hypothesis 1, 2, 3, 4 and 5.

$$L_r(\theta_o) = \sum_{i=0}^{M} \rho(\theta_{s,i}, \theta_o) L_e(\theta_{s,i}) \cos\theta_{s,i}. \tag{4.7}$$

[1]For this formulation, we assume that ϕ is negative, i.e. we rotate from left to right on Figure 4.1

The cosine can be included into the expression of the BRDF. We obtain :

$$L_r(\theta_o) = \sum_{i=0}^{M} \rho'(\theta_{s,i}, \theta_o) L_e(\theta_{s,i}) \qquad \rho'(\theta_{s,i}, \theta_0) := \rho(\theta_{s,i}, \theta_0) \cos\theta_{s,i}. \tag{4.8}$$

In order to leverage diversity, our notations need to take into account the fact that we rotate the observer and that we use several realizations of the source. Updated notations are provided in table 4.2.

Notation	**Meaning**
$\theta_{o,ja}$	Angle between $\boldsymbol{n_w}$ and the direction pointing from the j^{th} wall point to the observer, for the a^{th} rotation of the observer. $\theta_{o,ja} \in]-\pi/2, \pi/2[$
$\theta_{s,jia}$	Angle between $\boldsymbol{n_w}$ and the direction pointing from the j^{th} wall point to the i^{th} source point, for the a^{th} rotation of the observer. $\theta_{s,jia} \in]-\pi/2, \pi/2[$
$L_{r,ab}(\theta_{o,ja})$	Total intensity of the light reflected for the a^{th} observer and the b^{th} source realization by the wall point corresponding to $\theta_{o,ja}$. $L_{r,ab}(\theta_{o,ja}) \in \mathbb{R}$
$L_{e,b}(\theta_{s,jia})$	Intensity of the light emitted for the b^{th} source realization by the source point corresponding to $\theta_{s,jia}$. $L_{e,b}(\theta_{s,jia}) \in \mathbb{R}$
v_{ib}	Intensity of the light emitted for the b^{th} source by its i^{th} point. $v_{ib} \in \mathbb{R}$
$\boldsymbol{L}_{r,a}$	"Render matrix" containing all the $L_{r,ab}(\theta_{o,ja})$ for the a^{th} observer.
$\boldsymbol{B}_a$	"BRDF matrix" containing all the $\rho'(\theta_{s,jia}, \theta_{o,ja})$ for the a^{th} observer.
$\boldsymbol{V}$	"Source matrix" containing all the v_{ib}.
A	Number of observer rotations. $A \in \mathbb{N}$
B	Number of source realizations. $B \in \mathbb{N}$

Table 4.2: Additional notations for formulation of rendering as a linear system

As hinted by the new notations, we can express the total intensity of the light reflected for the a^{th} observer and the b^{th} source as follows :

$$L_{r,ab}(\theta_{o,ja}) = \sum_{i=0}^{M} \rho'(\theta_{s,jia}, \theta_{o,ja}) L_{e,b}(\theta_{s,jia}) \qquad \rho'(\theta_{s,jia}, \theta_{o,ja}) := \rho(\theta_{s,jia}, \theta_{o,ja}) \cos\theta_{s,jia}. \tag{4.9}$$

As explained in Section 2.1.1, all $\theta_{s,jia}$ linked to a wall point $\boldsymbol{w}_j$ is related by a bijection to each of the source points $\boldsymbol{s}_i$. As such, we can replace $L_{e,b}(\theta_{s,jia})$ in 4.9 by the unique, corresponding v_{ib} :

$$L_{r,ab}(\theta_{o,ja}) = \sum_{i=0}^{M} \rho'(\theta_{s,jia}, \theta_{o,ja}) v_{ib}. \tag{4.10}$$

We can now formally define the Source matrix $\boldsymbol{V}$, as well as the BRDF matrix $\boldsymbol{B}_a$ and the Render matrix $\boldsymbol{L}_{r,a}$.

The BRDF matrix $\boldsymbol{B}_a$ contains all the information about the BRDF for the a^{th} rotation of the observer. Its M columns correspond to each of the M points of the light source. Its N lines correspond to each of the N wall points seen by the observer during one rotation.

$$\boldsymbol{B}_a := \begin{bmatrix} \rho'(\theta_{s,11a}, \theta_{o,1a}) & \dots & \rho'(\theta_{s,1Ma}, \theta_{o,1a}) \\ \vdots & \ddots & \vdots \\ \rho'(\theta_{s,N1a}, \theta_{o,Na}) & \dots & \rho'(\theta_{s,NMa}, \theta_{o,Na}) \end{bmatrix} \qquad \boldsymbol{B}_a \in \mathbb{R}^{N \times M}.$$

The Source matrix $\boldsymbol{V}$ contains the intensity of light emitted by each point of the source. Its M lines correspond to the M points ("pixels") of the source, and the B columns correspond to the B

random realizations of the source.

$$\boldsymbol{V} := \begin{bmatrix} v_{11} & \dots & v_{1B} \\ \vdots & \ddots & \vdots \\ v_{M1} & \dots & v_{MB} \end{bmatrix} \qquad \boldsymbol{V} \in \mathbb{R}^{M \times B}.$$

The Render matrix $\boldsymbol{L}_{r,a}$ the intensity of the light reflected by all wall points for the a^{th} rotation of the observer. Its N lines correspond to the N points of the wall seen by the observer, and its B columns correspond to the B random realizations of the source.

$$\boldsymbol{L}_{r,a} := \begin{bmatrix} L_{r,a1}(\theta_{o,1a}) & \dots & L_{r,aB}(\theta_{o,1a}) \\ \vdots & \ddots & \vdots \\ L_{r,a1}(\theta_{o,Na}) & \dots & L_{r,aB}(\theta_{o,Na}) \end{bmatrix} \qquad \boldsymbol{V} \in \mathbb{R}^{N \times B}.$$

Having defined these three matrices, we can rewrite (4.10) as a matrix product :

$$\boldsymbol{L}_{r,a} = \boldsymbol{B}_a \times \boldsymbol{V}. \tag{4.11}$$

In conclusion, if we rotate the observer A times and take B realizations of the source, we will obtain $A \times B$ renders : the rotating observer scans the whole wall B times, once for each source realization. This translates into the following matrices :

- A matrices $\boldsymbol{L}_{r,a}$, each corresponding to a rotation a of the observer.
- A matrices $\boldsymbol{B}_a$, each corresponding to a rotation a of the observer.
- The matrix V, which already contains all the information about the source.

For convenience, we also define $\boldsymbol{L}_r$, which concatenates horizontally all of the $\boldsymbol{L}_{r,a}$. The matrix $\boldsymbol{B}$ concatenates horizontally all of the $\boldsymbol{B}_a$:

$$\boldsymbol{L}_r := \begin{bmatrix} \boldsymbol{L}_{r,1} & \dots & \boldsymbol{L}_{r,A} \end{bmatrix} \quad \boldsymbol{L}_r \in \mathbb{R}^{N \times AB}. \qquad \boldsymbol{B} := \begin{bmatrix} \boldsymbol{B}_1 & \dots & \boldsymbol{B}_A \end{bmatrix} \quad \boldsymbol{B} \in \mathbb{R}^{N \times AM}.$$

4.2 An Algorithm for Blind De-Rendering

4.2.1 Multiple Observers, Multiple Sources : Leveraging Diversity

Let us recall the matrix product formulation of the direct rendering problem :

$$\boldsymbol{L}_{r,a} = \boldsymbol{B}_a \times \boldsymbol{V} \qquad 1 \leq a \leq A. \tag{4.12}$$

To solve the blind inverse problem, we must find both $\boldsymbol{B}_a$ and $\boldsymbol{V}$, starting from $\boldsymbol{L}_{r,a}$. In what follows, we will asume that $N > M$, i.e. the number of points on the wall is higher than the number of points on the source. If there is only one source realization and one observer rotation ($A = 1$, $B = 1$). This problem is ill-posed, and admits an infinity of solutions : there are not enough constraints on the possible values of $\boldsymbol{B}_a$ and $\boldsymbol{V}$ to restrict the set of solutions.

To stabilize the solution to the inverse problem, we leverage diversity. This is where the formulation of the forward problem described in the previous section comes into play. Solving the problem for B random realizations of the light source certainly adds more unknowns, but constrains the BRDF : the M BRDF coefficients linked to a single wall point.

Indeed, for $B = 1$, for a single wall point, we have only one equation, M unknowns v_i and M unknown BRDF coefficients :

$$L_{r,a}(\theta_{o,1a}) = \rho'(\theta_{s,11a}, \theta_{o,1a})v_1 + ... + \rho'(\theta_{s,1Ma}, \theta_{o,1a})v_M. \tag{4.13}$$

However, for $B > 1$, we have $B - 1$ equations which only add M unknowns, since the BRDF coefficients remain unchanged. For $B = 1$, the ratio between equations and unknowns is equal to $\frac{1}{2M}$. Asymptotically, for $B \to \infty$, we reach a ratio of $\frac{1}{M}$, which is an improvement, but is still far from enough.

The other method used to restrict the set of solutions has been explored in the previous section, and consists in rotating the observer. In a sense, this method is the opposite of the previous one : this time, we add more unknown BRDF coefficients in order to stabilize the value of the source unknowns.

However, under the assumption that the geometrical configuration is well known alongside the Render matrix, we can show that a chunk of $\boldsymbol{B}_a$, BRDF matrix of the a^{th} rotation of the observer, can be predicted using $\boldsymbol{B}_{a-1}$. That way, while still adding new equations to the system (because we add new points), we only add unknowns for the wall points whose BRDF coefficients cannot be predicted from the previous observer iteration.

Hypothesis 6bis - Quantities required to solve the inverse problem : Alongside the full render matrix $\boldsymbol{L}_r$, the following quantities are known :

- M, the number of source points, their position $\boldsymbol{s}_i$, and the number of realizations B.
- The position of the observer $\boldsymbol{o}$, the number of rotations A, the horizontal field of view $\alpha_1 - \alpha_n$, but also the rotation angle[a] ϕ.

[a] In practice, we determine a wider interval that must be scanned by the observer after A iterations, and compute ϕ from these parameters.

Hypothesis 7bis - Large N : N is large enough for (4.6) to be verified.

The combination of those two hypothesis allows us to estimate a fraction of each BRDF matrix $\boldsymbol{B}_a$ using $\boldsymbol{B}_{a-1}$. As a consequence, the full BRDF matrix $\boldsymbol{B}$ can now be approximated by a matrix containing only $\boldsymbol{B}_1$, and the coefficients of each $\boldsymbol{B}_a$ that cannot be estimated using $\boldsymbol{B}_{a-1}$. We define that structuring matrix as follows :

$$\boldsymbol{B}_{\text{diff}} := \begin{bmatrix} \boldsymbol{B}_1 \\ \boldsymbol{C}_2 \\ \vdots \\ \boldsymbol{C}_A \end{bmatrix} \qquad \boldsymbol{B}_a \in \mathbb{R}^{(N+\xi)\times M}. \tag{4.14}$$

In the expression above, we denote by $\boldsymbol{C}_a$ the coefficients of $\boldsymbol{B}_a$ that cannot be estimated using $\boldsymbol{B}_{a-1}$, and by ξ the combined number of lines of all $\boldsymbol{C}_a$.

We have mentioned earlier that a chunk of $\boldsymbol{B}_a$, BRDF matrix of the a^{th} rotation of the observer, can be predicted using $\boldsymbol{B}_{a-1}$. This prediction relies of the hypothesis 7bis, and consists in :

- Finding the set of points contained in the intersection of the fields of view of $\boldsymbol{o}_a$ and the fields of view $\boldsymbol{o}_{a-1}$.

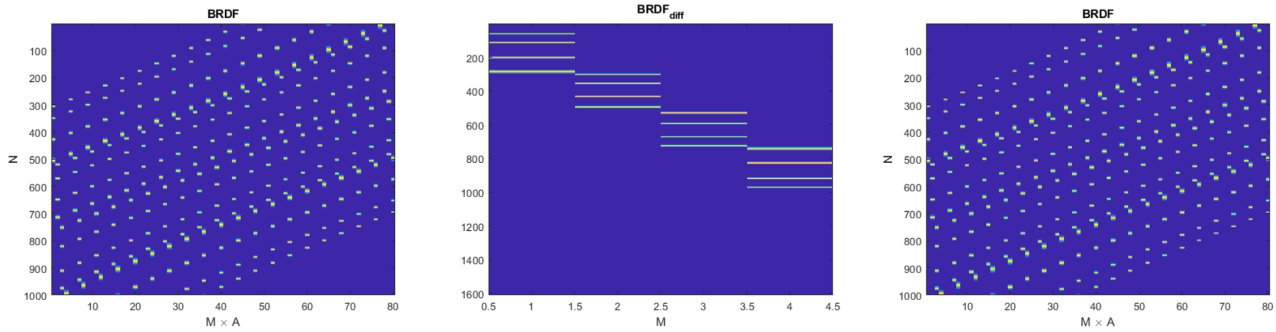

Figure 4.2: From left to right : $\boldsymbol{B}$, $\boldsymbol{B}_{\text{diff}} = U(\boldsymbol{B})$, $\boldsymbol{B}' = u^{-1}(U(\boldsymbol{B}))$

- Assign to each wall point $\boldsymbol{w}_a$ contained in the intersection the value of the closest $\boldsymbol{w}_{a-1}$.

This is close to a translation, but not quite, as the size of the intersection can vary due to the wall points not being linearly spaced. Please note once again that a large N is required for this prediction to be accurate.

In order to build our final cost function, we also need to define the operator U :

$$\boldsymbol{B}_{\text{diff}} := U(\boldsymbol{B}) \qquad U^{-1}(\boldsymbol{B}_{\text{diff}}) = \boldsymbol{B}. \tag{4.15}$$

The operator U is described by algorithm 2. We complete that description with Fig. 4.2. The most interesting diagram is the representation of $\boldsymbol{B}_{\text{diff}}$ that in the center. We can notice two characteristics of $\boldsymbol{B}_{\text{diff}}$:

- $\boldsymbol{B}_{\text{diff}}$ is sparse.
- Each column of $\boldsymbol{B}_{\text{diff}}$ corresponds to one source point. There is a structure to be noticed, linked to the position of each source point in the scene. This structure is due to the fact that ρ_{test} is a function of the angle sum. To understand the usefulness of such a structure, one must recall that even if we correctly solve the inverse problem and recover all coefficients, our current formulation does not provide any guarantee of the estimated source points being at their original index. The structure observed on $\boldsymbol{B}_{\text{diff}}$ allows us to post-process the result to obtain a well-ordered solution. **Hyp.8, defined in Chapter 3, is thus necessary to the obtention of a well-ordered solution.**

There is also a slight error between $\boldsymbol{B}$ and $\boldsymbol{B}'$, due to the approximations necessary to implement U.
$U^{-}1$ simply consists in reverting that operation, which is straightforward thanks to hypothesis 6bis : since we know all the quantities required to compute the points of the wall for each observer rotation, we can affect back the $\boldsymbol{C}_a$ to the points $\in \Omega_{W,a}$ that are outside of the interval bounded by the bounds of $\Omega_{W,a-1}$.

4.2.2 Objective Function and Algorithm Description

In this section, estimates are denoted as follows : $\hat{\boldsymbol{B}}$ is an estimate of $\boldsymbol{B}$.

The inverse problem can be formulated as an optimization problem, whose objective functions stem naturally from (4.11).We use a typical least-squares formulation to try to minimize the estimation error. As mentioned in the previous section, the operator U is used to enforce constraints on the BRDF linked to the geometrical configuration of the problem. However, $\boldsymbol{B}_{\text{diff}}$ also contains much less coefficients than $\boldsymbol{B}$, which allows for lower computational complexity. Enforcing sparsity on $\boldsymbol{B}_{\text{diff}}$ allows for a more stable solution. A typical α is 0.1.

Algorithm 2: Implementation of the U operator. Creates $\boldsymbol{B}_{diff}$ by concatenating $\boldsymbol{B}_1$ with the new coefficients of each $\boldsymbol{B}_a$

Preparation:

```
// Made possible by hypothesis 6bis
```

Compute the position of the set of wall points W_a for each observer rotation.

Compute $\hat{\boldsymbol{B}}$, the BRDF matrix.

Results:

$\boldsymbol{B}_{\text{diff}}$, the structured BRDF matrix.

Initialization:

$\boldsymbol{B}_{\text{diff}} \leftarrow 0_{AN,M}$

Index $\leftarrow 0$

Algorithm:

foreach $\boldsymbol{B}_a \in \boldsymbol{B}$ **do**

  if $\boldsymbol{B}_a = \boldsymbol{B}_1$ **then**

    `// Assign B1 to the N first lines of Bdiff`

    $\boldsymbol{B}_{\text{diff},1\ldots N} \leftarrow \boldsymbol{B}_a$

    $i \leftarrow N+1$

  else

    $c \leftarrow$ {Number of points $\in W_a$ that are outside of the interval bounded by the bounds of $W_a - 1$ }

    $\boldsymbol{C}_a \leftarrow \boldsymbol{B}_{a,N-c+1\ldots N}$

    $\boldsymbol{B}_{\text{diff},i\ldots i+c-1} \leftarrow \boldsymbol{C}_a$

    $i \leftarrow N+1$

  end

  $\boldsymbol{B}_{\text{diff}} \leftarrow \boldsymbol{B}_{\text{diff},1\ldots i}$ `// Truncation to get rid of unwanted zeros`

end

Objective functions

$$\arg\min_{\hat{\boldsymbol{B}}_{\text{diff}}} ||\boldsymbol{L}_{r,a} - \hat{\boldsymbol{B}}_a \hat{\boldsymbol{V}}||_2^2 + \alpha ||\hat{\boldsymbol{B}_{\text{diff}}}||_1 \quad \hat{\boldsymbol{B}} = U^{-1}(\hat{\boldsymbol{B}_{\text{diff}}}), \quad 1 \leq a \leq A, \quad 0 \leq \hat{\boldsymbol{B}}_{\text{diff}} \leq 1. \tag{4.16}$$

$$\arg\min_{\hat{\boldsymbol{V}}} ||\boldsymbol{L}_{r,a} - \hat{\boldsymbol{B}}_a \hat{\boldsymbol{V}}||_2^2 \quad \hat{\boldsymbol{B}} = U^{-1}(\hat{\boldsymbol{B}_{\text{diff}}}), \quad 1 \leq a \leq A, \quad 0 \leq \hat{\boldsymbol{V}} \leq 1. \tag{4.17}$$

Drawing inspiration from iterative blind deconvolution methods, we solve the problem by performing an alternative minimization. The full blind de-rendering algorithm is described in Algorithm 3.

Two practical details are worth mentioning :

- We obtain better results when we start building $\boldsymbol{B}_{\text{diff}}$ based on $\boldsymbol{B}_{A/2}$ instead of $\boldsymbol{B}_1$ (intuitively, we start from the observation located at the middle of the total interval scanned, instead of starting at the extremity of the interval). That requires slight modifications to Algorithm 2 : one must first add the $\boldsymbol{C}_a$ corresponding to the first half of the interval, and then do the same of the second half of the interval.
- To further increase the performance of the algorithm, we can leverage the sparsity of $\boldsymbol{B}_{\text{diff}}$ to reduce the number of coefficients.

4.2.3 Numerical Results and Performance

Our algorithm was implemented in MATLAB, notably using the solver`fmincon` of the optimization toolbox. To introduce this section, we provide typical estimates of the source (Fig. 4.3) and the BRDF (Fig. 4.4) for a successful algorithm iteration. The parameters used to obtain these results are given in Table 4.3. Please note that these parameters are also the default ones used for the performance analysis.

On Fig. 4.3, the source estimate looks reasonably close to the true source. The extrema and the relative intensity are recovered.

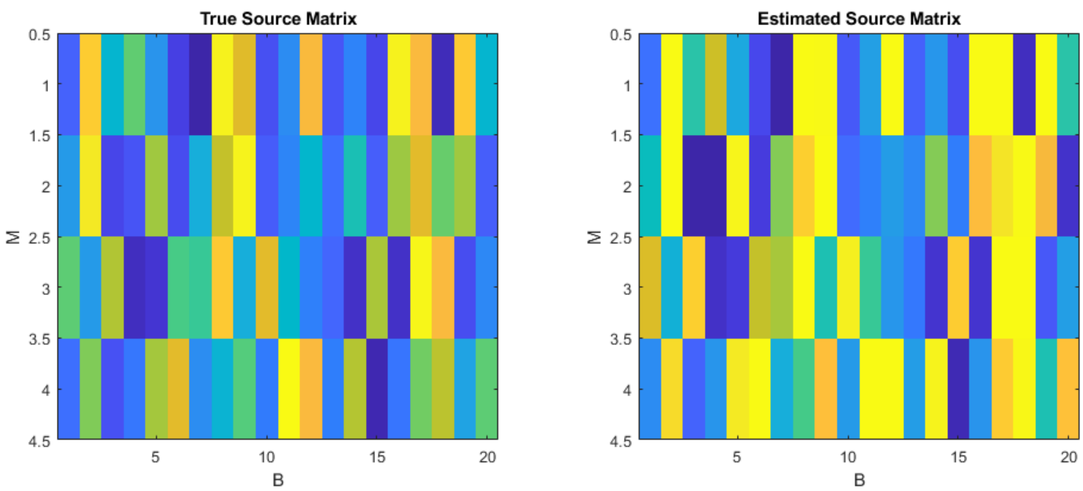

Figure 4.3: Source estimate ($\hat{\boldsymbol{V}}$) visual inspection

On Fig. 4.4, the BRDF estimate also looks reasonably close to the true source. As intended, the structure is preserved. However, some of the nuance in the values is lost. This is due to the fact that, in practice, we accelerate the algorithm by leveraging the sparsity of $\hat{\boldsymbol{B}}_{\text{diff}}$ and considering

Algorithm 3: Blind de-rendering algorithm. Based on alternating minimization of cost functions developed in (4.16).

Preparation:

Compute the position of the set of wall points W_a for each observer rotation.

Obtain the Render matrix $\boldsymbol{L}_r$.

Results:

$\hat{\boldsymbol{V}}$, an estimator of the source matrix, and $\hat{\boldsymbol{B}}$, an estimator of the BRDF matrix

Parameters (with examples):

Tolerance for convergence $\epsilon = 10^{-2}$

$\alpha = 0.1$

Initialization:

```
foreach B̂_a ∈ B̂ do
    // Initialize B by averaging Lr over the source realizations.
        B̂_a ← [mean(L_{r,a}), ..., mean(L_{r,a})]
end
// Initialize V at random (uniform distribution)
    V̂ ← Z    Z ∈ R^{N×B}    Z ~ U(0,1)
    // Initialize Bdiff using the operator U
    B̂diff ← U(B̂)
    D1 ← ∞, D2 ← ∞, D1p ← D1, D2p ← D2, ΔD ← ∞
```

$\hat{\boldsymbol{B}}_a \leftarrow [\overline{\boldsymbol{L}_{r,a}}, ..., \overline{\boldsymbol{L}_{r,a}}]$

$\hat{\boldsymbol{V}} \leftarrow Z \quad Z \in \mathbb{R}^{N \times B} \quad Z \sim \mathcal{U}(0,1)$

$\hat{\boldsymbol{B}_{\text{diff}}} \leftarrow U(\hat{\boldsymbol{B}})$

$D_1 \leftarrow \infty,\ D_2 \leftarrow \infty,\ D_{1p} \leftarrow D_1,\ D_{2p} \leftarrow D_2,\ \Delta D \leftarrow \infty$

Algorithm:

while $\Delta D > \epsilon$ **do**

`// Source update`

$\hat{\boldsymbol{V}} \leftarrow \arg\min_{\hat{\boldsymbol{V}}} ||\boldsymbol{L}_{r,a} - \hat{\boldsymbol{B}}_a \hat{\boldsymbol{V}}||_2^2, \quad 0 \le \hat{\boldsymbol{V}} \le 1, \quad \hat{\boldsymbol{B}} = U^{-1}(\hat{\boldsymbol{B}_{\text{diff}}})$

$D_{1p} \leftarrow D_1,\ D_1 \leftarrow ||\boldsymbol{L}_{r,a} - \hat{\boldsymbol{B}}_a \hat{\boldsymbol{V}}||_2^2, \quad \hat{\boldsymbol{B}} = U^{-1}(\hat{\boldsymbol{B}_{\text{diff}}})$

`// BRDF update.`

$\hat{\boldsymbol{B}}_{\text{diff}} \leftarrow \arg\min_{\hat{\boldsymbol{B}}_{\text{diff}}} ||\boldsymbol{L}_{r,a} - \hat{\boldsymbol{B}}_a \hat{\boldsymbol{V}}||_2^2 + \alpha||\hat{\boldsymbol{B}_{\text{diff}}}||_1, \quad 0 \le \hat{\boldsymbol{B}}_{\text{diff}} \le 1, \quad \hat{\boldsymbol{B}} = U^{-1}(\hat{\boldsymbol{B}_{\text{diff}}})$

$D_{2p} \leftarrow D_2,\ D_2 \leftarrow ||\boldsymbol{L}_{r,a} - \hat{\boldsymbol{B}}_a \hat{\boldsymbol{V}}||_2^2, \quad \hat{\boldsymbol{B}} = U^{-1}(\hat{\boldsymbol{B}_{\text{diff}}})$

$\Delta D \leftarrow D_2 - D_1$

end

$\hat{\boldsymbol{B}} \leftarrow U^{-1}(\hat{\boldsymbol{B}_{\text{diff}}})$

Parameter	Value
N	500
BRDF	Train of four gaussians, $\sigma = 1$
$\boldsymbol{o}$	[0, -5]
$\boldsymbol{s}$	[0, -1]
A	20
B	20
M	4
α	0.1
β	$\pi/6$

Table 4.3: Default blind derendering parameters

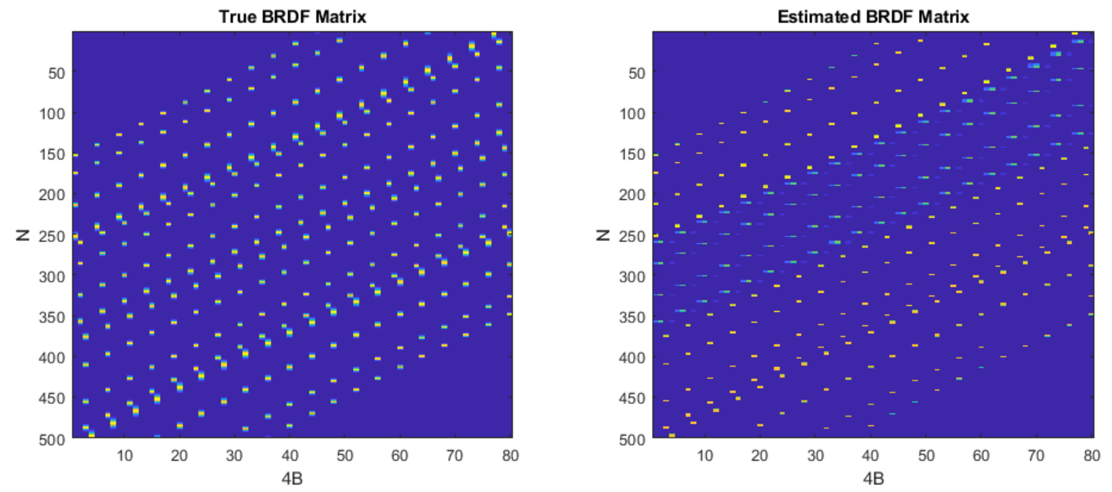

Figure 4.4: Source estimate ($\hat{\boldsymbol{B}}$) visual inspection

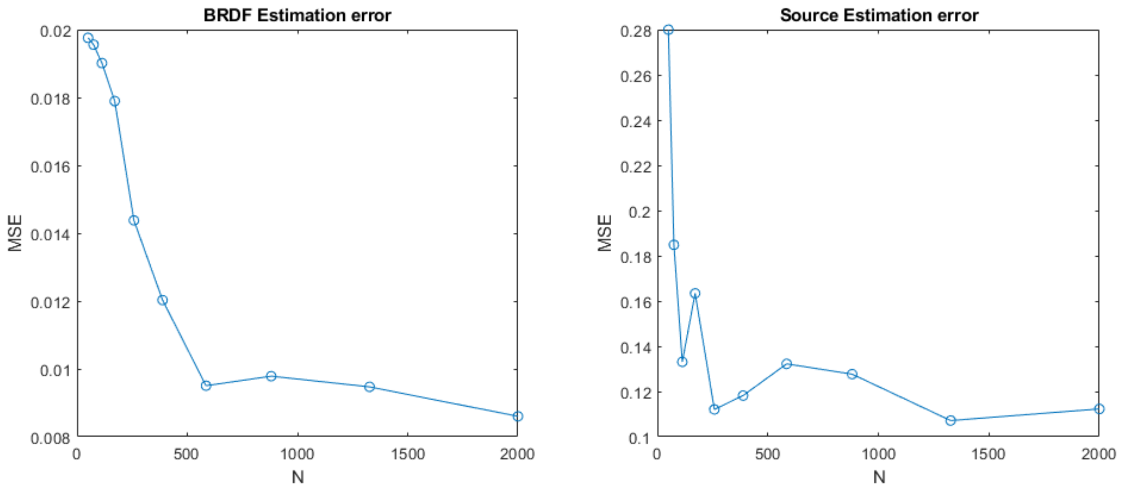

Figure 4.5: Performance analysis : Number of wall points N

only a fraction of its coefficients.

For this performance analysis, we observe the evolution of the estimation error on the source and BRDF (MSE) with respect to parameters critical to our algorithm. The MSE was averaged over 96 iterations of the algorithm.

On Fig. 4.5, we observe the performance of the algorithm with respect to N, the number of points. Our ability to leverage our geometrical configuration (i.e. the validity of hypothesis 7bis) is directly tied to N, and that translates into a lower error when N is higher. $N = 500$ seems to be a satisfying threshold.

On Fig. 4.6, we observe the performance of the algorithm with respect to A, the number of observer rotations. As expected, A primarily benefits the estimation of the source. Past a certain threshold ($A = 10$), performance gets worse. The more the observer rotates, the more unknowns are added to the BRDF, and much most importantly, the more the solution relies on the validity of hypothesis 7bis, which might not be completely guaranteed if N is on the low side.

On Fig. 4.7, we observe the performance of the algorithm with respect to B, the number of source realizations. B clearly benefits to the estimation of the BRDF. The performance of the source estimation also increases with B, but not without hiccups. $B = 25$ seems to be a good threshold.

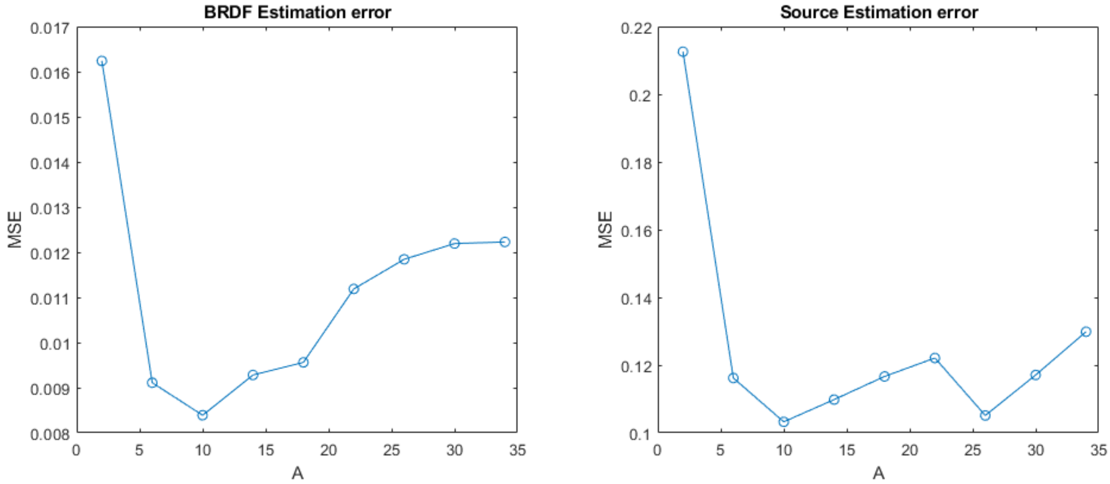

Figure 4.6: Performance analysis : Number of observer rotations A

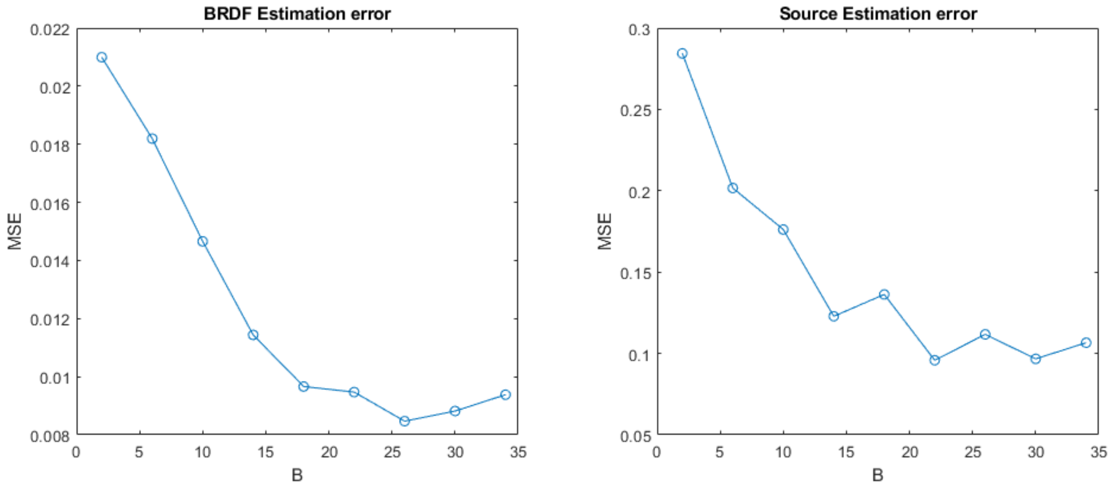

Figure 4.7: Performance analysis : Number of source realizations B

On Fig. 4.8, we observe the performance of the algorithm with respect to both A and B. The most interesting diagram is the left-handed one, dedicated to the BRDF. We can clearly see how increasing B improves its estimator, and how increasing A worsens the performance.

On Fig. 4.9, we observe the performance of the algorithm with respect to both M. Both the source and the BRDF estimation error increase with M. Of course, more source points mean more unknowns, and thus more underdeterminacy. However, a threshold seems to appear at $M = 10$, where the error ceases to increase.

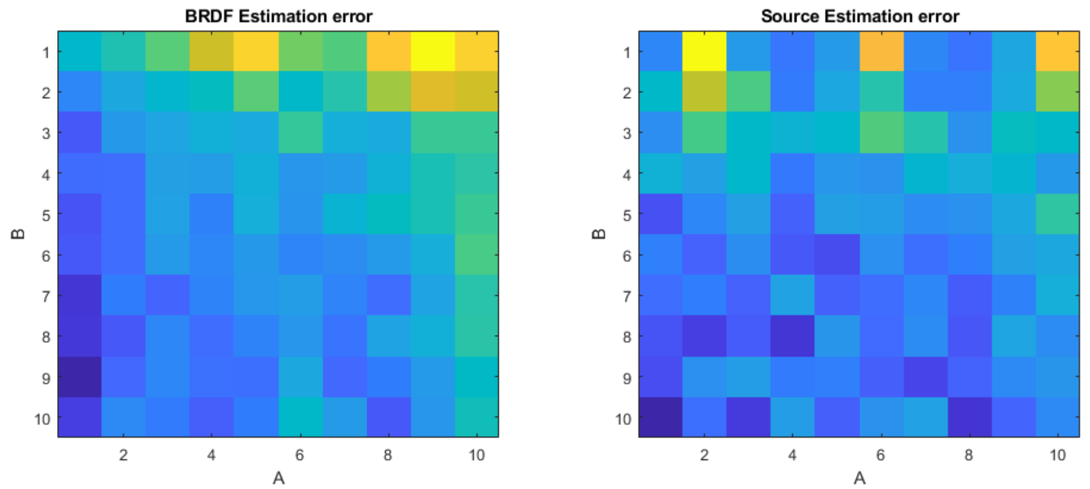

Figure 4.8: Performance analysis : Crossing A with B

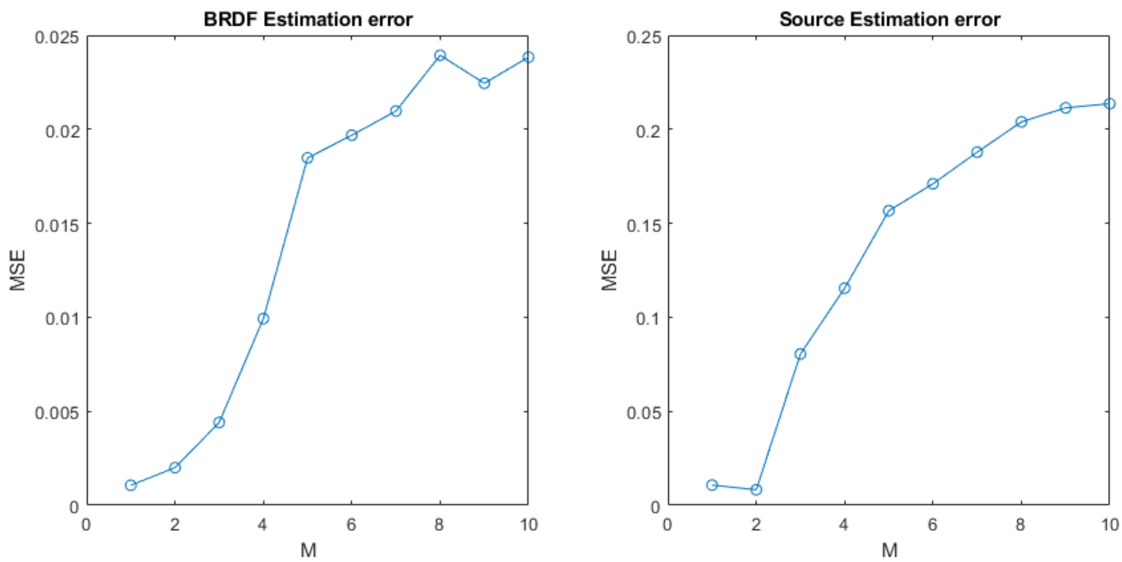

Figure 4.9: Performance analysis : Source support size M

Conclusion

Let us come back to the two goals of this master thesis that we defined in the introduction :

- Identify an algorithm that allows for the simultaneous recovery of the BRDF and the incoming light distribution, given the outgoing light distribution.
- Given that algorithm, formulate the assumptions on the source and the BRDF that ensure their recovery.

The performance analysis shows that the first goal is successfully reached. A visual inspection of the estimates (cf Fig. 4.4, 4.3) of the source and the BRDF confirm that the algorithm is able to recover both desired quantities.

The second objective is also fulfilled. We have determined thresholds for the most important parameters of the algorithm. For $N = 500$, $A = 10$ and $B = 25$, our algorithm is able to retrieve acceptable estimates for the BRDF and the source. Together with the hypothesis formulated throughout this document, these thresholds compose a set of conditions that must be met to maximize the robustness of the algorithm.

Numerous improvement perspectives exist. In section 3.1.1, we explored the equivalence of the rendering equation and the convolution on the sphere, and decided not to build on these foundations. However, after leveraging our geometrical configuration to solve the de-rendering problem, a similar reasoning could be applied to the convolution on the sphere.

The main reason we shifted from the convolution on the sphere to 1D convolution was that a render only provided us with a fraction of the full spherical convolution at each point. However, under the hypothesis formulated in chapter 3, the incoming light distribution and the BRDF are similar for all wall points up to a rotation angle...that we can actually predict, as most geometrical parameters are known to us !

1D blind deconvolution also feels like a missed opportunity, especially since non-blind deconvolution provides good results, assuming that all the hypotheses are valid. In the framework of this thesis, we explored several single-channel blind deconvolution algorithms such as [34], [35] or [36]. The lack of results further convinced us of the necessity to use multi-channel algorithms. Our inability to adapt Focus Blind Deconvolution to de-rendering does not mean that no deconvolution algorithm is suitable to solve the inverse problem formulated in Chapter 3...

And of course, extending the de-rendering algorithm to a 3D geometrical configuration would surely allow for interesting and more visually striking results. However, the present algorithm is already resource-intensive (all the diagrams showcased in the performance analysis took more than a week to compute), and heavy optimization is required to significantly increase performance. The full code is available at `https://github.com/jlvoiseux/Master-Thesis-Code`.

Bibliography

[1] James T. Kajiya. The rendering equation. *SIGGRAPH Comput. Graph.*, 20(4):143–150, August 1986.

[2] Miika Aittala, Prafull Sharma, Lukas Murmann, Adam Yedidia, Gregory Wornell, Bill Freeman, and Fredo Durand. Computational mirrors: Blind inverse light transport by deep matrix factorization. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 14311–14321. Curran Associates, Inc., 2019.

[3] M. Baradad, V. Ye, A. B. Yedidia, F. Durand, W. T. Freeman, G. W. Wornell, and A. Torralba. Inferring light fields from shadows. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6267–6275, 2018.

[4] Katherine L. Bouman, Vickie Ye, Adam B. Yedidia, Fredo Durand, Gregory W. Wornell, Antonio Torralba, and William T. Freeman. Turning corners into cameras: Principles and methods. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[5] Ravi Ramamoorthi and Pat Hanrahan. A signal-processing framework for inverse rendering. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 117–128, 2001.

[6] Frédo Durand, Nicolas Holzschuch, Cyril Soler, Eric Chan, and François X. Sillion. A frequency analysis of light transport. *ACM Trans. Graph.*, 24(3):1115–1126, July 2005.

[7] Eric W Weisstein. *Spherical Coordinates.* MathWorld - A Wolfram Web Resource.

[8] *Spherical Coordinates.* Wikipedia, the free encyclopedia.

[9] *Why is* $|\cos\theta d\omega|$ *the projection of the differential solid angle d onto the (x,y)-plane?* Stack-Exchange - Mathematics.

[10] Jean Taine. *Transferts thermiques - Notes de cours.* Ecole centrale Paris, September 2016.

[11] Daniel Ritchie Ravi Ramamoorthi. *Illumination and Reflection - CS 294-13, Advanced Computer Graphics, Lecture Notes.* University of California, Berkeley, September 2009.

[12] R.G. Grainger. *An Atmospheric Radiative Transfer Primer*, pages 43–80. CRC Press, 2013.

[13] Warren J. Smith. *Modern Optical Engineering: The Design of Optical Systems, Fourth Edition*, pages 255–256. The McGraw-Hill Companies, Inc., 2008.

[14] Michal Haindl and Jiří Filip. *Visual Texture: Accurate Material Appearance Measurement, Representation and Modeling*, pages 119–145. Springer London, London, 2013.

[15] Tong Lai Yu. *CSE 420 Computer Graphics.* School of Computer Science and Engineering, California State University, San Bernardino, November 2018.

[16] D. Guarnera, G. C. Guarnera, A. Ghosh, C. Denk, and M. Glencross. Brdf representation and acquisition. EG '16, pages 625–650, Goslar Germany, Germany, 2016. Eurographics Association.

[17] Jaroslav Krivanek. *Computer graphics III – Light reflection, BRDF.* Computer Graphics Group, Charles University, Praha, November 2015.

[18] Roman Durikovic. *Computer Graphics - Light Transport BRDFs Shading.* Department of Applied Informatics, Comenius University in Bratislava.

[19] Jaroslav Krivanek. *Computer graphics III – Rendering equation and its solution.* Computer Graphics Group, Charles University, Praha, November 2015.

[20] A.D. Polyanin and A.V. Manzhirov. *Handbook of Integral Equations.* Handbooks of mathematical equations. CRC Press, 1998.

[21] Albert Liu Ravi Ramamoorthi. *Global Illumination and the Rendering Equation - CS 294-13, Advanced Computer Graphics, Lecture Notes.* University of California, Berkeley, September 2009.

[22] Kevin Beason. *smallpt: Global Illumination in 99 lines of C++.* 2011.

[23] Peter Shirley. *Ray-Tracing, the rest of your life.* 2019.

[24] Manuel T. Silva. *Handbook of Digital Signal Processing - Engineering Applications.*

[25] Subhasis Chaudhuri, Rajbabu Velmurugan, and Renu Rameshan. *Blind Image Deconvolution: Methods and Convergence*, pages 11–35. Springer International Publishing, 2014.

[26] Frédéric Cohen Tenoudji. *Analog and Digital Signal Analysis: From Basics to Applications*, pages 321–335. Springer International Publishing, 2016.

[27] José Bioucas-Dias. *Inverse Problems in Imaging.* Instituto de Telecomunicacoes - Instituto Superior Técnico, Lisboa, September 2016.

[28] Mark Davenport. *ECE 6254: Statistical Machine Learning : Regression, least squares, and Tikhonov regularization.* School of Electrical and Computer Engineering, Georgia Tech, 2017.

[29] Jiunn-Lin Wu, Chia-Feng Chang, and Chun-Shih Chen. An adaptive richardson-lucy algorithm for single image deblurring using local extrema filtering. *[U+6DE1][U+6C5F][U+7406][U+5DE5][U+5B78][U+520A]*, 16(3):269–276, 2013.

[30] D. Kundur and D. Hatzinakos. Blind image deconvolution. *IEEE Signal Processing Magazine*, 13(3):43–64, May 1996.

[31] G. R. Ayers and J. C. Dainty. Iterative blind deconvolution method and its applications. *Optics Letters*, 13:547–549, July 1988.

[32] D. A. Fish, A. M. Brinicombe, E. R. Pike, and J. G. Walker. Blind deconvolution by means of the richardson–lucy algorithm. *J. Opt. Soc. Am. A*, 12(1):58–65, Jan 1995.

[33] P. Bharadwaj, L. Demanet, and A. Fournier. Focused blind deconvolution. *IEEE Transactions on Signal Processing*, 67(12):3168–3180, June 2019.

[34] Sunghyun Cho and Seungyong Lee. Fast motion deblurring. In *ACM SIGGRAPH Asia 2009 papers*, pages 1–8. 2009.

[35] Ali Ahmed, Benjamin Recht, and Justin Romberg. Blind deconvolution using convex programming, 2012.

[36] D. Krishnan, T. Tay, and R. Fergus. Blind deconvolution using a normalized sparsity measure. In *CVPR 2011*, pages 233–240, 2011.

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
École polytechnique de Louvain
Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve, Belgique | **www.uclouvain.be/epl**