# Composing Light
## Designing Freeform lenses for illumination applications using algorithmic differentiation

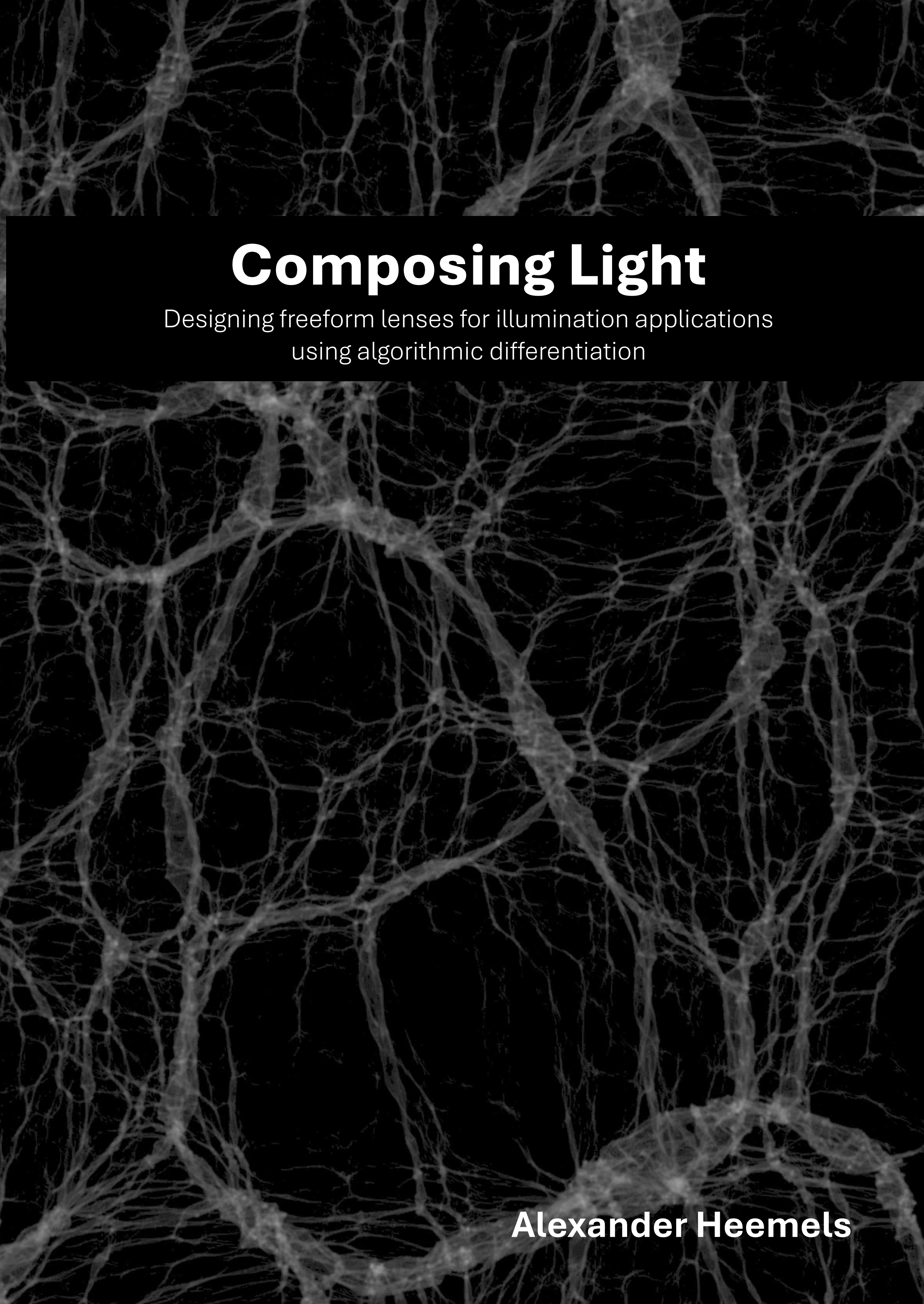
Heemels, A.N.M.

**DOI**
10.4233/uuid:214f151d-3cf5-408a-81d9-803701a66f5d

**Publication date**
2025

**Document Version**
Final published version

**Citation (APA)**
Heemels, A. N. M. (2025). *Composing Light: Designing Freeform lenses for illumination applications using algorithmic differentiation*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:214f151d-3cf5-408a-81d9-803701a66f5d



# Composing Light

Designing freeform lenses for illumination applications using algorithmic differentiation

Alexander Heemels

# Composing Light

Designing freeform lenses for illumination applications using algorithmic differentiation

# COMPOSING LIGHT

## DESIGNING FREEFORM LENSES FOR ILLUMINATION APPLICATIONS USING ALGORITHMIC DIFFERENTIATION

**Dissertation**

for the purpose of obtaining the degree of doctor
at Delft University of Technology
by the authority of the Rector Magnificus, prof. dr. ir. T.H.J.J. van der Hagen,
chair of the Board for Doctorates
to be defended publicly on
17th of November 2025 at 12:30 o'clock

by

**Alexander Nicholas Michael HEEMELS**

Masters of Science in System and Control, Technische Universiteit Delft,
born in Robbinsdale, Minnesota, United States of America.

This dissertation has been approved by the promotors.

Composition of the doctoral committee:

| | |
|---|---|
| Prof. dr. C.R. Kleijn, | Chairman |
| Em. prof. dr. P. Urbach , | Delft University of Technology, promotor |
| Dr. rer. nat. M. Möller , | Delft University of Technology, poromotor |
| Dr. A.J.L. Adam , | Delft University of Technology, copromotor |
| *Independent members:* | |
| Prof. dr. Y. Meuret, | KU Leuven |
| Prof. C. Giannelli, | Univeristy of Florence |
| Prof. dr. W.M.J.M. Coene, | Delft University of Technolog |
| Dr. ir. M.J.H. Anthonissen, | Eindhoven University of Technology |
| *Reserve member:* | |
| Prof. dr. S. Stallinga, | Delft University of Technology |

ISBN 978-94-6522-900-3

*To do things right,*
*first you need love,*
*then technique.*

Antoni Gaudí

# Nomenclature

**Geoemtrical variables**

| | |
|---|---|
| $\mathbf{C}$ | Control point tensor |
| $\mathbf{c}$ | Control point |
| $\mathbf{N}$ | Vector of B-spline basis functions |
| $\mathbf{R}$ | B-spline Refinement matrix |
| $\mathbf{S}$ | B-spline surface |
| $\mathbf{T}$ | Truncation Matrix |
| $\mathbf{X}$ | Characteristic Matrix |
| $\mathscr{B}$ | B-spline vector space |
| $\mathscr{H}$ | Hierarchical B-spline vector space |
| $\mathscr{T}$ | Truncated hierarchical B-spline vector space |
| $K$ | Conic constant of lens |
| $N$ | B-spline basis function |
| $p, q$ | degrees of B-spline |
| $R$ | Radius of curvature of lens |
| $u, v$ | Knot |
| $U, V$ | Knot vectors |

**Optimization variable**

| | |
|---|---|
| $\alpha$ | Step size |
| $\bar{x}$ | Adjoint variable |
| $\mathbf{e}$ | Unit vector |
| $\mathbf{w}$ | Weighted optimization parameters |
| $\mathscr{L}$ | Loss function |
| $\mu$ | Regularization parameter, Langrange multiplier |
| $\nabla$ | Gradient operator |
| $\nabla_{\mathbf{q}}$ | Directional gradient in direction $\mathbf{q}$ |

| | |
|---|---|
| $\boldsymbol{\pi}$ | Optimization parameters |
| $c$ | Optimization constraint |

**Physics constants**

| | |
|---|---|
| $\lambda$ | Wavelength |
| $\mathbf{p}$ | ray momentum vector |
| $\mathbf{x}$ | Phase space vector of a ray |
| $\mathscr{F}$ | Fourier transform operator |
| $\mathscr{F}^{-1}$ | inverse Fourier transform operator |
| $\mathfrak{h}$ | Hemisphere |
| $\mathfrak{R}$ | Fresnel coefficient: reflected light |
| $\mathfrak{T}$ | Fresnel coefficient: transmitted light |
| $\phi$ | Latitude angle |
| $\theta$ | Longitude angle |
| $n$ | Refractive index |
| $s$ | Optical path length |
| $W$ | Measurement function, Reconstruction filter |

**Radiometric Quantities**

| | |
|---|---|
| $\mathscr{U}$ | Étendue |
| $\mathfrak{L}$ | Basic radiance |
| $\Omega$ | Solid Angle |
| $\Omega^{\perp}$ | Projected Solid Angle |
| $\Phi$ | Radiometric Flux |
| $E$ | Irradiance |
| $I$ | Intensity |
| $L$ | Radiance |
| $Q$ | Radiant Energy |
| $Q_p$ | Photon Energy |

# CONTENTS

# 1 INTRODUCTION

In the field of illumination optics, optical engineers design optical elements to direct light from a source (LED, laser, or incandescent lamp) to achieve a desired irradiance (spatial density of radiant flux) or intensity (angular density of the radiant flux) [40]. Freeform lenses and reflectors, which are optics without any symmetry, are attractive to optical designers as they can enable greater flexibility in creating compact illumination solutions.

However, determining the geometry of a freeform lens or reflector that produces a specified irradiance distribution is nontrivial. Many existing techniques assume a zero-étendue source (i.e., a point or collimated source). Under that assumption, the freeform design problem can be formulated as a Monge–Kantorovich mass transport problem and solved either via the Monge–Ampére equation [99, 100, 76, 97, 73] or through optimization [27], for instance using the supporting quadratic method [37, 68]. Another popular approach is ray mapping [32, 11, 12, 25], which first constructs a mapping from rays leaving the source to their locations on the target, and then uses that mapping to compute the geometry freeform optic.

Difficulties arise when extended sources (finite étendue sources) are considered. Methods such as simultaneous multiple surface (SMS) [28, 8, 81] handle this case by mapping edge rays but cannot create arbitrary irradiance distributions. Alternatively, an extended source can be regarded as a perturbation of a zero-étendue problem, in which the finite source size effectively blurs the target distribution. This perspective leads to feedback-type methods that iterate between designing the freeform optic under zero-étendue assumptions and then adjust the target irradiance to progressively approach the desired distribution [53, 57, 16, 91, 36].

Another route is to directly optimize the lens geometry [46, 19]. However, such approaches often get stuck in local minima [93, 92, 101, 63]. For these methods, three components largely determine their effectiveness: how system performance is evaluated, how the geometry of the surface is modeled, and how updates to the surface are computed.

## 1.1. HOW DO WE EVALUATE THE PERFORMANCE OF FREEFORMS?

To evaluate system performance, ray tracing is used. Ray tracing simulates how light traverses the system by computing the paths of rays originating from the source through the optical elements. There are two modes of ray tracing: sequential and non-sequential. Sequential ray tracers, such as Zemax [3] and Code V [83], are mainly used in the design of imaging optics. They trace relatively few rays to evaluate image quality metrics and assume rays travel sequentially from one surface to the next in a fixed order. In non-sequential ray tracers, such as LightTools [84] and Photopia [55], a large number of rays are simulated to determine the optical flux through the system. Few assumptions are made about ray paths or surface interactions, and a single incident ray can split into multiple rays, for example, due to Fresnel reflections or scattering. Programs tailored for illumination applications, however, often require considerable computation time to trace a sufficient number of rays. In computer graphics, rendering is performed in a manner similar to non-sequential ray tracing, however, capable of handling far more rays much more efficiently, primarily by leveraging graphics processing units (GPUs). Historically, these renderers have seen limited use in optical design because some speed-ups rely on approximations that favor visually pleasing results over physical accuracy (e.g., interactions that do not strictly conserve energy). Recently, there has been progress toward physically based rendering [70], enabling optical engineers to benefit from graphics libraries' speed without sacrificing physical correctness.

## 1.2. HOW DO WE MODEL THE SURFACE OF A FREEFORM?

As freeforms do not have symmetry, they require more advanced modeling methods than spherical and aspheric lenses, which are rotationally symmetric. The most common approach is to describe a freeform surface using polynomial bases. Orthogonal polynomial bases such as XY, Chebyshev, or Zernike [59, 18] can represent complex surfaces via the weights (coefficients) of the basis functions. A major issue, however, is that when a minor local change to the lens surface is desired, many coefficients must be adjusted. This can be problematic when designing freeforms for complex irradiance distributions, which typically require many polynomials. Alternatively, B-splines [98] and their nonuniform rational variant, NURBS [96], can be used to describe freeform surfaces. These geometries have control points that allow local adjustment of the surface. In addition, these surfaces can be gradually made more complex—a favorable property when designing freeforms for complex irradiance distributions—as implemented by Wang et al. [90], who use knot insertion [72] to increase the degrees of freedom. However, there is still a catch: each time a knot is added, an entire strip of the freeform surface must be refined, introducing more degrees of freedom in regions where they are not needed. Therefore, alternative spline descriptions can be used, such as LR [26], HB [34], THB [38], U [85], and T-splines [79]. T-splines have been proposed in the context of freeform design [6, 22]. In this work, we make use of *truncated hierarchical B-splines* (THB-splines) [38, 39].

## 1.3. HOW DO WE ADJUST THE SURFACE OF A FREEFORM?

Once it has been established how light propagates through the freeform lens and how the surface is modeled, the remaining question is: how should we adjust the freeform lens to approach the desired irradiance distribution? A common strategy is to compute gradients of the design parameters. Gradients indicate whether changing a parameter improves or degrades performance, making them useful for freeform lens optimization. There are two mainstream ways to obtain these gradients: finite differences and algorithmic (automatic) differentiation. Finite differences are straightforward: perturb each parameter of the freeform lens and evaluate whether the objective improves, repeating this for all parameters. Each perturbation requires one or more simulations to assess how the change affects the result. This may be manageable for simple freeform lenses with few degrees of freedom, however, becomes computationally expensive as the number of parameters grows. To address this, one can use algorithmic/automatic differentiation with specialized libraries such as PyTorch [69], TensorFlow [61], and JAX [14], which have matured rapidly due to machine-learning applications. Differentiable ray tracers have been developed and applied to both sequential [88, 82, 89, 65] and non-sequential settings [66, 47, 50, 90, 52]. The key benefit is that gradients of the freeform lens parameters are computed alongside the ray-tracing simulation, adding little to no overhead to the total runtime.

## 1.4. WHAT NOW?

In this thesis we will combine these developments: fast, physically accurate ray tracing, which is algorithmically differentiable in combination with B-spline surface descriptions to design freeform lenses.

We begin by introducing the physics behind illumination engineering and ray tracing (Chapter 2). Modeling freeform surfaces using B-splines and THB-splines, and how to refine these surfaces, is discussed in Chapter 3. In Chapter 4, we cover strategies for choosing and optimizing parameters, how gradients are computed, and how to enforce additional constraints.

After these foundations, we present our initial implementation for designing freeform lenses with B-spline surfaces (Chapter 5), followed by refinement strategies using THB-splines (Chapter 6). In Chapter 7, we then ask a more fundamental question: what limitations do finite étendue sources impose on achievable irradiance distributions?

Finally, we close by summarizing our main findings and outlining directions for future work.

# 2 FOUNDATIONS OF GEOMETRICAL OPTICS

To design an illumination system (e.g., for streetlights, car headlights, or indoor luminaires), we must understand how light or radiative energy emitted by a source propagates to a target (such as a desk or a highway), how it interacts with objects along the way, and how to quantify the amount of light arriving at the target. The propagation of radiative energy can be described using wave or ray models. Although the wave model provides the most accurate description of light behavior, its calculations are often too complex for macroscopic applications. Therefore, the ray model, also known as geometrical optics, is typically used to design illumination systems. In geometrical optics, light propagates along curves called rays. It should be noted that the wave nature of light becomes important when designing structures or objects whose dimensions are on the order of the wavelength. Within the scope of this thesis, the ray model is assumed to be sufficiently accurate. Light also has properties such as coherence, which causes interference, and polarization, which defines the direction of its electric field oscillation. In this thesis, we will not consider these properties. All sources will be assumed to be monochromatic (single-wavelength), spatially incoherent (so no interference occurs), and unpolarized (meaning there is no defined oscillation direction). In this chapter, we begin by mathematically defining a ray as a curve in 3D space. We then explain how each curve can be represented in phase space by specifying its coordinates on a plane and its angle with respect to the plane's normal, effectively reformulating it as a point in 4D. Next, we discuss how rays are reflected or refracted at interfaces separating different materials and how the energy carried by each ray is distributed based on the material's properties and the ray's incidence angle. We also introduce the radiometric quantities used to quantify light at the target namely: flux, radiance, irradiance, and intensity. Finally, we explain how these concepts are combined to simulate light propagation through an illumination system via non-sequential ray tracing.

## 2.1. DEFINING A SINGLE RAY

In a homogeneous medium with a constant refractive index $n$, a ray is a straight line that passes through a point $\mathbf{r}_0 = [r_{0,x}\ r_{0,y}\ r_{0,z}]^T$ and has a direction or momentum vector $\mathbf{p} = [p_x\ \ p_y\ \ p_z]^T$ whose length is equal to the refractive index $|\mathbf{p}| = n$ [9, Chapter 3, Equation 25]. The length along the ray's path is called the *optical path length*, denoted by the scalar $s$. The ray equation can then be written as:

$$\mathbf{r}(s) = \mathbf{r}_0 + s\,\mathbf{p}, \tag{2.1}$$

as illustrated in Figure 2.1(a).

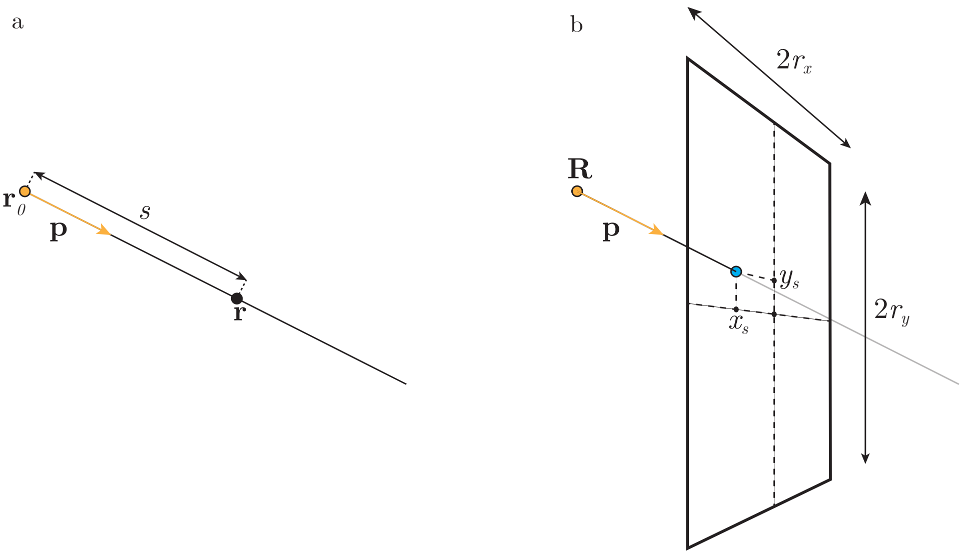


Figure 2.1: (a) A ray with a starting point $\mathbf{r}_0$ and momentum vector $\mathbf{p}$, showing point $\mathbf{r}$ on the ray at optical path length $s$; (b) A plane used to determine the $(x, y)$ coordinates of the ray in phase space.

Often, we need to track how a ray travels between surfaces or planes, such as an LED chip in a streetlight and a section of highway. In these scenarios, it is convenient to specify the ray using four coordinates. To do this, we first define a plane of width $2r_x$ and height $2r_y$, which the ray intersects. We then adopt a local coordinate system $x \in [-r_x, r_x]$, $y \in [-r_y, r_y]$ forming the rectangular area $\mathscr{A} = [-r_x, r_x] \times [-r_y, r_y]$. The point where the ray intersects the plane is denoted $(x_s, y_s)$ and is shown in Figure 2.1(b). Since $|\mathbf{p}| = n$, we have

$$p_z = \pm\sqrt{n^2 - p_x^2 - p_y^2}. \tag{2.2}$$

Taking $p_z > 0$ (assuming the ray propagates in the positive optical-axis direction), we can write $p_x$ and $p_y$ in spherical coordinates:

$$p_x = n \sin(\theta)\cos(\phi), \quad p_y = n \sin(\theta)\sin(\phi), \tag{2.3}$$

where the angles $\theta$ and $\phi$ are illustrated in Figure 2.2. Hence, to specify $\mathbf{p}$, we need two angles with respect to the plane's normal: $\theta \in [0, \pi/2]$ (the longitude) and $\phi \in [0, 2\pi]$ (the latitude). These angles cover the hemisphere $\mathscr{H} = [0,\pi/2] \times [0,2\pi]$. Combining $(x_s, y_s)$ with $(\theta,\phi)$, each ray is defined by $\mathbf{x} = (x, y, \theta, \phi)$ and belongs to the phase space $\mathscr{R} = \mathscr{A} \times \mathscr{H}$.

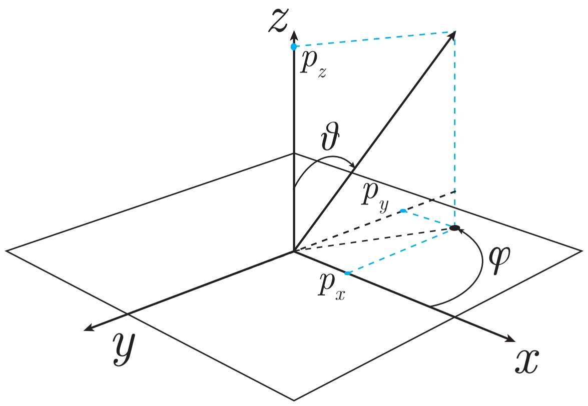


Figure 2.2: Definition of angles: longitude $\theta$ and latitude $\phi$

## 2.2. RAY-SURFACE INTERACTIONS

When a ray encounters an interface between two different media, its trajectory changes based on the optical properties of those media. If the interface is smooth (meaning that the scale over which surface changes occur is much larger than the wavelength), the ray undergoes specular reflection and refraction. In those cases, its path follows the *law of specular reflection* and *Snell's law* for refraction. Additionally, the fraction of refracted versus reflected light is determined by the *Fresnel equations*, which depend on the polarization of the light.

### 2.2.1. REFLECTION AND REFRACTION

Suppose a ray hits an interface with an incident direction $(\theta_i,\phi_i)$. It is reflected into a direction $(\theta_r,\phi_r)$. The longitudinal angle is conserved: $\theta_r = \theta_i$, but the latitudinal angle $\phi_r$ is rotated by $\pi$ radians:

$$\theta_r = \theta_i \quad \text{and} \quad \phi_r = \phi_i + \pi. \tag{2.4}$$

Likewise, if a ray traveling in a material of refractive index $n_i$ encounters an interface with a material of index $n_t$, it refracts into a transmitted direction $(\theta_t,\phi_t)$, according to *Snell's law*:

$$n_i \sin\theta_i = n_t \sin\theta_t \quad \text{and} \quad \phi_t = \phi_i. \tag{2.5}$$

This can be viewed as momentum conservation for the ray, where $n\sin\theta$ remains constant across the interface. However, as noted in Section 2.1, the ray's optical momentum $(|p| = n)$ is proportional to the refractive index. Consequently, if a ray travels from a high-index medium into a low-index medium such that it cannot satisfy $n_i \sin\theta_i = n_t \sin\theta_t$, the ray undergoes *total internal reflection*.

Although total internal reflection may appear to be a discontinuous phenomenon, where rays suddenly reflect inside the material, it is actually a continuous phenomenon: each ray splits into transmitted and reflected components, with the fraction of energy in each part depending on the angle of incidence. As the angle increases, the reflected component grows until it eventually carries all the energy, and the ray is fully reflected.

### 2.2.2. FRESNEL EQUATIONS

When a ray hits an interface between two different media, it splits into a reflected and a refracted ray. The fraction of energy in each ray depends on the ray's incidence angle, the refractive indices on both sides of the interface, and the light's polarization. This dependence is governed by the *Fresnel equations*. While polarization is an electric field property describing the oscillation direction it can be included in geometrical optics by assigning a particular polarization to a ray. Linearly polarized light is commonly assumed when discussing reflection and refraction. If the electric field oscillates in a single plane, we label it: p-polarized when it oscillates parallel to the plane of incidence, or s-polarized (from the German word *senkrecht*) if it is perpendicular to the plane of incidence. Any circular and elliptical polarization can be treated as a combination of those two. The Fresnel equations are derived from boundary conditions enforcing continuity of the tangential electric and magnetic fields across the interface[43, 9]. Let $\mathfrak{R}_s$ and $\mathfrak{T}_s$ denote the s-polarized reflection and transmission coefficients, which are defined as follows:

$$\mathfrak{R}_s = \left| \frac{n_i \cos\theta_i - n_t \cos\theta_t}{n_i \cos\theta_i + n_t \cos\theta_t} \right|^2, \tag{2.6}$$

$$\mathfrak{T}_s = \left| \frac{2 n_i \cos\theta_i}{n_i \cos\theta_i + n_t \cos\theta_t} \right|^2, \tag{2.7}$$

$$\tag{2.8}$$

with $\mathfrak{R}_s + \mathfrak{T}_s = 1$. For p-polarized light,

$$\mathfrak{R}_p = \left| \frac{n_t \cos\theta_i - n_i \cos\theta_t}{n_t \cos\theta_i + n_i \cos\theta_t} \right|^2, \tag{2.9}$$

$$\mathfrak{T}_p = \left| \frac{2 n_i \cos\theta_i}{n_t \cos\theta_i + n_i \cos\theta_t} \right|^2. \tag{2.10}$$

with $\mathfrak{R}_p + \mathfrak{T}_p = 1$. In illumination applications, light is usually unpolarized. Therefore, it is convenient to define an unpolarized reflectance $\mathfrak{R}_u$ and transmittance $\mathfrak{T}_u$ for which: $\mathfrak{R}_u + \mathfrak{T}_u = 1$. The unpolarized reflectance is the average of the reflectances of the two orthogonal linear polarizations:

$$\mathfrak{R}_u = \frac{1}{2}\left(\mathfrak{R}_s + \mathfrak{R}_p\right). \tag{2.11}$$

The reflectances and transmittances for different incidence angles for a ray transitioning from a medium with refractive index $n_i = 1$ to a medium with refractive index $n_t = 1.5$ are shown in Figure 2.3(a), and for a ray transitioning from a medium with refractive index $n_i = 1.5$ to a medium with refractive index $n_t = 1$ in Figure 2.3(b).

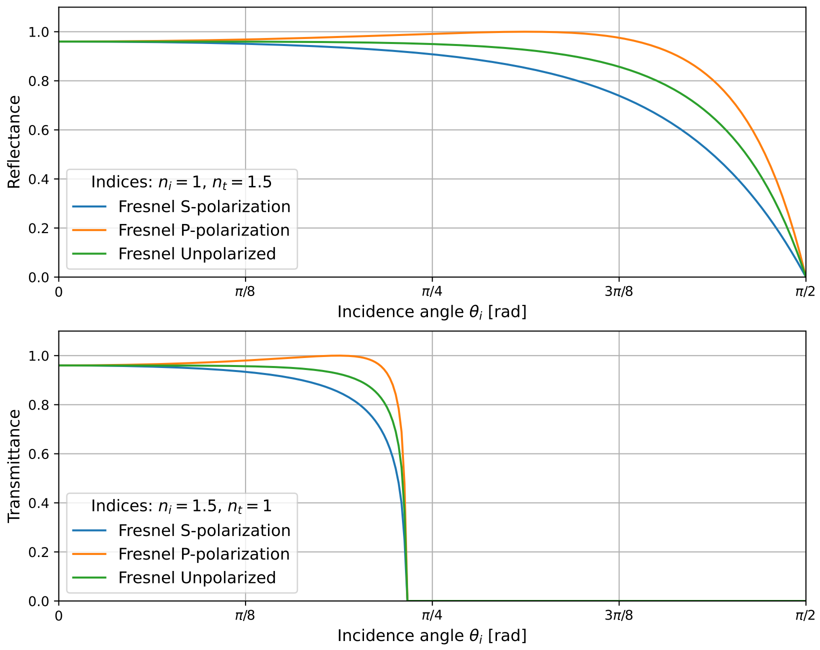


Figure 2.3: Transmission coefficients for a transition from (a) refractive index $n_i = 1$ to $n_t = 1.5$; (b) refractive index $n_i = 1.5$ to $n_t = 1$

### A NOTE ON THE THEORETICAL MAXIMUM EFFICIENCY WITH LENSES

By using Fresnel coefficients, we can get an upper bound of the achievable efficiency of an optical system with multiple lens elements (assuming no anti-reflection coatings). The unpolarized reflection coefficient for normal incidence $\theta_i = 0$ is:

$$\mathfrak{R}_u = \frac{1}{2}\left(\frac{n_i - n_t}{n_i + n_t}\right)^2 + \frac{1}{2}\left(\frac{n_t - n_i}{n_i + n_t}\right)^2. \tag{2.12}$$

If each lens is made of glass with a refractive index of 1.5, then at every interface approximately 96% of the light is transmitted. As each lens has two interfaces, about 92% of the light passes through a single lens. Hence, in a system with $N_l$ lenses, the theoretical maximum efficiency for a source emitting a total flux $\Phi_0$ is

$$\Phi_{\max} = 0.96^{2N_l}\Phi_0. \tag{2.13}$$

## 2.3. WORKING WITH A BUNDLE OF RAYS: RADIOMETRY

Radiometry characterizes the distribution of electromagnetic radiation in space. The most important quantities are *radiant flux*, *irradiance*, *intensity*, and *radiance*. To define these radiometric variables, we first introduce the concepts of *plane angle* and *solid angle*.

### 2.3.1. PLANE AND SOLID ANGLE

A curve $C$ is said to subtend a *plane angle* $\theta$ at a point **P**, as illustrated in Figure 2.4. To find this plane angle, let **A** and **B** be the starting and ending points of the curve. We then radially project these points onto a circle of radius $r$ centered at **P** and denote the projected points as **A**′ and **B**′. The plane angle $\theta$ is then the ratio of the arc length $s$ between **A**′ and **B**′ and the circle's radius:

$$\theta = \frac{s}{r}. \tag{2.14}$$

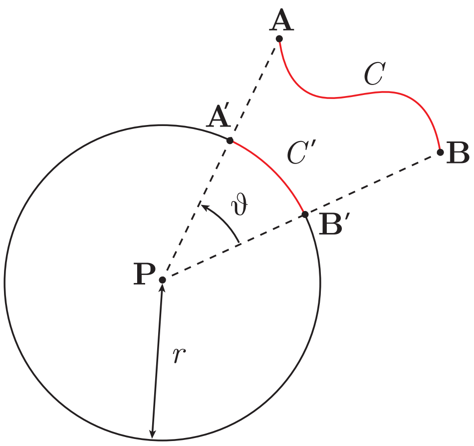


Figure 2.4: Definition of plane angle $\theta$.

A *solid angle* $\Omega$ is subtended by a closed curve $C$ with respect to a point **P** at the center of a sphere of radius $r$. Let $C'$ be the projection of $C$ onto the sphere. This curve $C'$ encloses an area $A$ on the sphere's surface, as shown in Figure 2.5(a). The solid angle is then given by

$$\Omega = \frac{A}{r^2}, \tag{2.15}$$

and is measured in steradians (sr).

The *projected solid angle* $\Omega^{\perp}$ is the solid angle projected onto the plane containing **P**, as shown in Figure 2.5(b). It is obtained by projecting the closed curve $C'$ onto the plane containing **P** yielding the closed curve $C''$. The area enclosed by this curve is the projected solid angle. A differential solid angle $\mathrm{d}\Omega$ then gives a differential projected solid angle:

$$\mathrm{d}\Omega^{\perp} = \cos\theta \mathrm{d}\Omega. \tag{2.16}$$

It can be useful to express solid angle in terms of the longitude $\theta$ and latitude $\phi$. Consider Figure 2.5(b), where the differential area d$A$ on a sphere of radius $r$ is

$$\mathrm{d}A = r^2 \sin\theta \mathrm{d}\theta \mathrm{d}\phi. \tag{2.17}$$

By Equation 2.15, we then obtain

$$\mathrm{d}\Omega = \sin\theta \mathrm{d}\theta \mathrm{d}\phi. \tag{2.18}$$

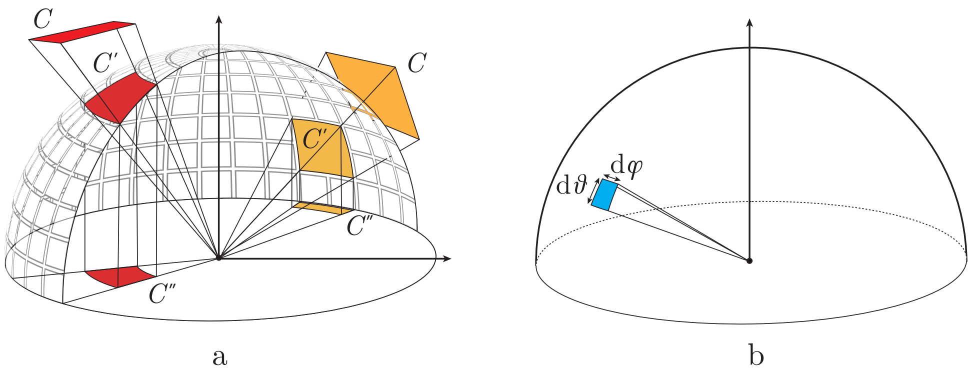


Figure 2.5: (a) Solid angle $\Omega$ and projected solid angle $\Omega^{\perp}$ of a closed curve $C$, which is first projected on the sphere to get $C'$ and then projected onto the plane to obtain $C''$; (b) Differential piece of solid angle in spherical coordinates.

### 2.3.2. RADIANT ENERGY

Radiant energy $Q$ is the total energy propagating onto, through, or emerging from a given surface. A photon of wavelength $\lambda$ carries energy

$$Q_p = \frac{hc}{\lambda}, \tag{2.19}$$

where $c$ is the speed of light ($c = 299,792,458$ m/s), and $h$ is Planck's constant, ($h \approx 6.626 \times 10^{-34}$ m$^2$ kg/s) The total radiant energy depends on the number of photons $N_p$ of a given wavelength measured over a time period:

$$Q = Q_p N_p. \tag{2.20}$$

### 2.3.3. RADIANT FLUX

Radiant flux, $\Phi$, is the radiant energy per unit time:

$$\Phi = \frac{\mathrm{d}Q}{\mathrm{d}t}, \tag{2.21}$$

with units of watts (W = J/s).

### 2.3.4. RADIANCE

Consider a small surface with area d$A$. Let $\theta$ and $\phi$ denote the longitudinal and latitudinal angles of a ray originating from a point $\mathbf{r}$ on this surface. The radiance $L(\mathbf{r},\theta,\phi)$, measured in $\mathrm{W}\cdot\mathrm{m}^{-2}\cdot\mathrm{sr}^{-1}$, is the radiant flux emitted by this area d$\Phi$ into a small solid angle d$\Omega$ around $(\theta,\phi)$, per unit of emitting area d$A$ and per unit of solid angle d$\Omega$:

$$L(\mathbf{r},\theta,\phi) = \frac{\mathrm{d}\Phi}{\cos\theta \mathrm{d}A\mathrm{d}\Omega}. \tag{2.22}$$

In other words, radiance, $L$, is the area and solid angle density of radiant flux: the flux emerging from a point $\mathbf{r}$ into a direction $(\theta,\phi)$. If we assume a lossless transmission from one material to another, incoming radiance $L_i$ travels through an interface from medium with index $n_i$ to one with index $n_t$. According to Snell's law, the solid angle d$\Omega_i$ in the incident medium maps to d$\Omega_t$ in the transmitted medium. Specifically,

$$\mathrm{d}\Omega_i = \sin\theta_i \mathrm{d}\theta_i \mathrm{d}\phi_i, \quad \mathrm{d}\Omega_t = \sin\theta_t \mathrm{d}\theta_t \mathrm{d}\phi_t, \tag{2.23}$$

where $\theta_t$ and $\phi_t$ are related to $\theta_i$ and $\phi_t$ by Equation 2.5. Because $\mathrm{d}\phi_i = \mathrm{d}\phi_t$ and $n_i\mathrm{d}\theta_i = n_t\mathrm{d}\theta_t$ in the small-angle limit, we have

$$n_i^2\mathrm{d}\Omega_i = n_i \sin\theta_i \; n_i\mathrm{d}\theta_i\mathrm{d}\phi_i = n_t^2\mathrm{d}\Omega_t. \tag{2.24}$$

Hence, the quantity

$$\mathfrak{L} = Ln^{-2}, \tag{2.25}$$

is invariant. This weighted radiance is often called the *basic radiance.*

### 2.3.5. IRRADIANCE

The irradiance $E$, with units $\mathrm{W}\cdot\mathrm{m}^{-2}$, is the radiant flux per unit area

$$E(\mathbf{r}) = \frac{\mathrm{d}\Phi}{\mathrm{d}A}. \tag{2.26}$$

Irradiance can also be obtained by integrating the radiance (Equation 2.22) over the entire hemisphere $\mathscr{H}$:

$$E(\mathbf{r}) = \int_{\mathscr{H}} L(\mathbf{r},\theta,\phi)\cos\theta \mathrm{d}\Omega. \tag{2.27}$$

### 2.3.6. INTENSITY

The intensity $I$, with units $\mathrm{W}\cdot\mathrm{sr}^{-1}$, is the flux density per solid angle

$$I(\theta,\phi) = \frac{\mathrm{d}\Phi}{\mathrm{d}\Omega}. \tag{2.28}$$

It can also be obtained by integrating the radiance over an area $\mathscr{A}$ on a surface of interest (e.g., a detector):

$$I(\theta,\phi) = \int_{\mathscr{A}} L(\mathbf{r},\theta,\phi)\cos\theta \mathrm{d}A(\mathbf{r}). \tag{2.29}$$

In some literature, intensity refers to the squared amplitude of the electric field rather than the flux per unit solid angle. In such cases, irradiance is the appropriate term.

### 2.3.7. ÉTENDUE

As discussed in Section 2.1, a ray intersecting a plane can be specified using four parameters $(x, y, \theta, \phi)$ which describe a point in 4D phase space. The étendue $\mathscr{U}$ of a bundle of rays passing through an elemental area $\mathrm{d}A$ on a surface $\mathscr{A}$ and forming an angle $\theta$ with the local normal is:

$$\mathscr{U} = \int_{\mathscr{A}}\int_{\mathscr{H}} n^2 \cos\theta \mathrm{d}A\mathrm{d}\Omega = \int\int\int\int n^2 \cos\theta \sin\theta \mathrm{d}\theta\mathrm{d}\phi\mathrm{d}x\mathrm{d}y = \int \mathrm{d}p_x\mathrm{d}p_y\mathrm{d}x\mathrm{d}y, \quad (2.30)$$

where $p_x$ and $p_y$ are the momentum coordinates defined in Equation 2.3. We can link the flux and radiance through étendue as the radiance represents flux per unit étendue:

$$\mathfrak{L}(x, y, \theta, \phi) = \frac{\mathrm{d}\Phi}{\mathrm{d}\mathscr{U}} = \frac{\Phi}{n^2\mathrm{d}A\cos\theta\mathrm{d}\Omega} = \frac{\mathrm{d}\Phi}{\mathrm{d}A\mathrm{d}p_x\mathrm{d}p_y}. \quad (2.31)$$

Conversely, the flux can be found by integrating the radiance distribution:

$$\Phi = \int \mathfrak{L}(x, y, \theta, \phi)\mathrm{d}\mathscr{U}. \quad (2.32)$$

Étendue is especially valuable in preliminary designs where no scattering, absorption, ray splitting, or wavelength conversion occurs, as under these conditions étendue is conserved [23, Chapter. 18] [94, Appendix. A] [64, Chapter. 2.5]. It sets limits on what is physically achievable in an illumination system.

## 2.4. FOLLOWING A RAY THROUGH A SYSTEM: NON-SEQUENTIAL RAY TRACING

When designing an optical system, we often want to know the intensity, irradiance, or radiance at a particular location or target. Consider a ray specified by $\mathbf{x}_{s,1} = (x_{s,1}, y_{s,1}, \theta_{s,1}, \phi_{s,1})$ with radiance $L_s$, emitted from the source domain $\mathscr{A}_s$ into the hemisphere $\mathscr{H}_s$ above the source. As it travels through the system, it may reach the target plane with coordinate $\mathbf{x}_{t,1} = (x_{t,1}, y_{t,1}, \theta_{t,1}, \phi_{t,1})$ and radiance $L_t$. In principle, we would construct a mapping $\mathscr{S}$ such that

$$L_t(\mathbf{x}_{t,1}) = \mathscr{S}\left\{L_s(\mathbf{x}_{s,1})\right\}. \quad (2.33)$$

However, there is no general way to derive an analytical expression for $\mathscr{S}$. Instead, we rely on ray tracing to simulate the ray's trajectory through the system. In non-sequential ray tracing, we propagate rays $\mathbf{x}_{s,k}$ (for $k = 1, \ldots, N_k$) from the source to the target according to the rules outlined in Section 2.2. By summing the rays at the target, we approximate Equation 2.27 as

$$E_t \approx \sum_{k=1}^{N_k} L_t\left(\mathbf{x}_{t,k}\right). \quad (2.34)$$

In doing so, we must address two key challenges: determining which objects a ray intersects and where, and computing the radiometric quantities at the target.

### 2.4.1. INTERACTION BETWEEN RAYS AND OBJECTS

As a ray propagates through the optical system, we must identify which objects it intersects and determine the points of intersection. Although analytical solutions exist for simple shapes (e.g., spheres, cylinders, and boxes), the complexity increases significantly for more complicated geometries. For surfaces described by cubic polynomials or higher, finding closed-form intersection solutions becomes cumbersome or even impossible. A common simplification is to enclose each object in a bounding box. We first check the intersection between the ray and the bounding box to see if the ray might encounter that object. If it does, we then either use a root-finding algorithm (e.g., Newton's method) to compute the ray-object intersection or approximate the object with a triangular mesh and use a search algorithm to locate the triangle intersected by the ray. Once the intersection point is known, we consult the object's bidirectional scattering distribution function (BSDF) to determine how the ray interacts with that surface. For instance, the ray may reflect when encountering a mirror, refract when the object is a lens, or scatter when it is a diffuser. In this thesis, we restrict ourselves to refraction while accounting for Fresnel coefficients. The process of identifying which object the ray interacts with, computing the intersection point, and determining the ray's subsequent behavior is repeated until the ray reaches the target.

### 2.4.2. DETERMINING THE RADIOMETRIC QUANTITIES AT THE TARGET

To determine radiance, irradiance, or intensity at specific points at the target, we discretize the domain where these quantities are evaluated. This is done by distributing $N_t \times M_t$ points, called pixels, on the target plane. Around each pixel, we define a *measurement function* or *reconstruction filter*, $W$, with finite support $\mathscr{A}_W$. This function dictates how much each ray's radiance contributes to the pixel based on where the ray intersects $\mathscr{A}_W$. Although measurement functions can seem abstract, they mirror real lab measurements. For example, a CCD measures irradiance over a square domain, converting incoming photons into an electric signal. In this case, the measurement function is essentially a *box filter* of width $w$:

$$W(x,y) = \begin{cases} 1 & |x| \leq w/2 \text{ and } |y| \leq w/2, \\ 0 & \text{otherwise.} \end{cases} \tag{2.35}$$

Alternatively, a single-mode fiber measuring local radiance behaves more like a *Gaussian filter*, defined by means $\mu_x$, $\mu_y$ and the standard deviations $\sigma_x$, $\sigma_y$:

$$g(x,y,\mu_x,\mu_y,\sigma_x,\sigma_y) = \frac{1}{2\pi\sigma_x\sigma_y} \exp\left(-\frac{(x-\mu_x)^2}{2\sigma_x^2} - \frac{(y-\mu_y)^2}{2\sigma_y^2}\right). \tag{2.36}$$

Typically, the standard deviations are chosen as $\sigma_x = \sigma_y = \sigma$. To limit the extent of the Gaussian filter to a domain with radius $R$, values beyond $\sqrt{x^2+y^2} > R$ are set to zero:

$$W(x,y) = \begin{cases} g(x,y,0,0,\sigma) - g(R,0,0,0,\sigma) & \sqrt{x^2+y^2} < R, \\ 0 & \text{otherwise.} \end{cases} \tag{2.37}$$

These measurement functions can then be incorporated into Equation 2.34:

$$E_{n,m} = \sum_{k=1}^{N_k} L_{\text{target}}\left(\mathbf{x}_{t,k}\right) W_{n,m}(\mathbf{x}_{t,k}). \tag{2.38}$$

Figure 2.6 shows how the pixel value ($P_2$) is computed on a 1D domain for both box and Gaussian filters, assuming all rays have the same radiance. When using the box filter, all rays within the filter's domain contribute equally. As such, the pixel value is the sum of their radiances. This is shown in Figure 2.6(b1) and Figure 2.6(c1). For the Gaussian filter shown in Figure 2.6(a2), the domains of the filters often overlap with filters from neighboring pixels. The radiance of all the rays falling within the domain of a filter are reweighted based on their distance from the center of the filter and then summed to give the total pixel value. This process is shown in Figure 2.6(b2) and Figure 2.6(c2). For an in-depth discussion on Monte Carlo integration, intersection algorithms, sampling and reconstruction, and other non-sequential ray tracing, the reader is directed to Pharr, Jakob, and Humphreys (2023).

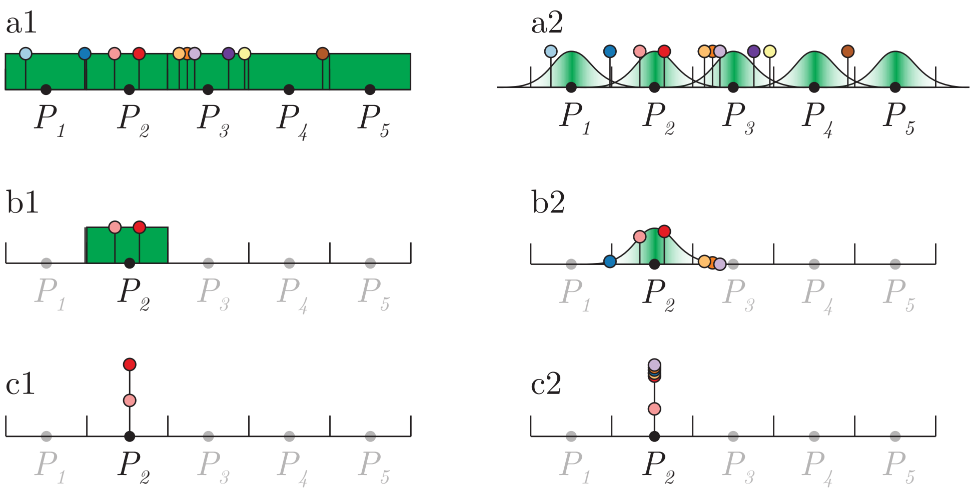


Figure 2.6: Determining a pixel value for a box filter and Gaussian filter: (a1 - a2) Pixels with their measurement functions and ray's radiances; (b1 - b2) Rays contributing to pixel $P_2$ under each filter; (c1 - c2) The final pixel weight is the sum of individual ray contributions

# 3

# THE GEOMETRY OF FREEFORM LENSES

We define a *freeform lens* as a volume with a uniform refractive index enclosed between six surfaces: one front, one rear, and four edge surfaces, as shown in Figure 3.1. The front and rear surfaces, depicted in blue and red in Figure 3.1, are defined on a rectangular domain and are taken as the freeform surfaces, meaning that these surfaces do not exhibit any symmetry. The edge surfaces, depicted in green, are obtained by connecting the boundaries of the front and rear surfaces. An overview of the essential parameters re-

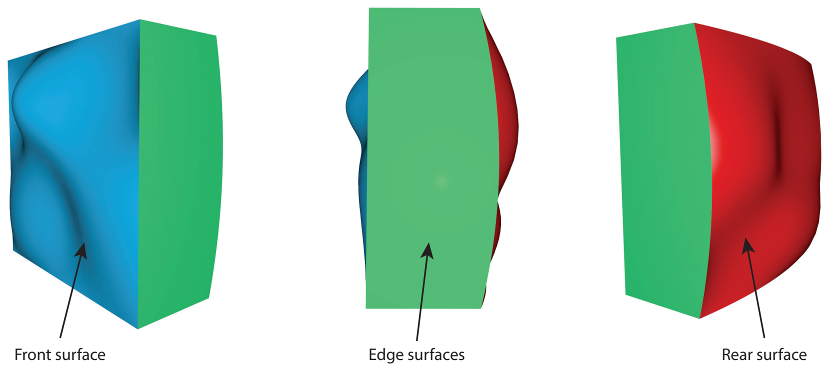


Figure 3.1: Illustration of a freeform lens geometry, consisting of two freeform surfaces: the front (blue) and rear (red), connected by the edge surfaces (green).

quired to define the freeform lens is shown in Figure 3.2(a). The extents of the rectangular domains on which the surfaces are defined are $(x_s, y_s) \in \mathscr{D}_s = [-r_{s,x}, r_{s,x}] \times [-r_{s,y}, r_{s,y}]$, where $s \in \{\text{front}, \text{rear}\}$, with $r_{s,x}$ and $r_{s,y}$ representing the half-width and half-height of the surface. The points $O_{\text{front}}$ and $O_{\text{rear}}$ serve as the local coordinate origins for the surfaces and are located on the optical axis. The distance between these points along the $z$-axis defines the lens thickness $d_1$.

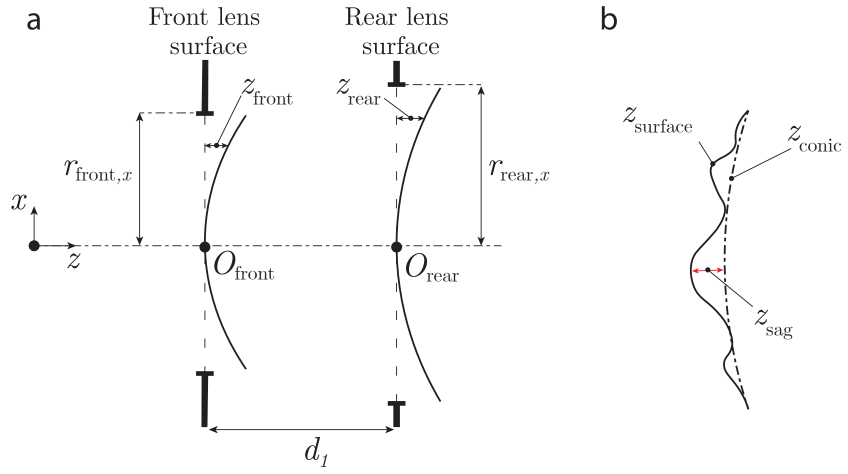


Figure 3.2: Illustration of the parameters required to define a freeform lens, (a) Parameters for the overall lens geometry; (b) Freeform surface definition.

## 3.1. DESCRIPTION OF A FREEFORM SURFACE

We define a freeform surface as the sum of a base conic and a sag component, where the sag represents the deviation from the conic surface

$$z_{\mathrm{surface}}(x, y) = z_{\mathrm{conic}}(x, y) + z_{\mathrm{sag}}(x, y), \tag{3.1}$$

as seen in Figure 3.2(b). The conic term depends on the surface radius $R$ and the conic constant $K$, and is expressed as:

$$z_{\mathrm{conic}}(x, y) = \frac{R}{(1+K)}\left[1 - \sqrt{1-(1+K)\left(x^2+y^2\right)\left(\frac{1}{R}\right)^2}\,\right], \tag{3.2}$$

where

$$x^2 + y^2 < \frac{R^2}{1+K}, \tag{3.3}$$

must hold to ensure a real-valued solution. The sag function can be represented using a basis of polynomial functions $f_i$ defined on the closed, rectangular domain $\mathscr{D}_s$, such as Legendre and Chebyshev-polynomials. In this case, the sag is given by the sum

$$z_{\mathrm{sag}}(x, y) = \sum_i w_i f_i(x, y),$$

where the polynomial weights $\{w_i\}$ are used as parameters to shape the surface. However, a modification to a single weight affects the entire surface. This can be problematic

when local modifications are required, as all weights must be adjusted to maintain the overall shape of the rest of the lens.
To allow localized surface modifications, B-splines can be used [72]. In the following sections, we discuss the construction of a B-spline surface using the B-spline basis functions and control points. We then demonstrate how new degrees of freedom can be added without altering the geometry using knot insertion. However, applying knot insertion to a surface globally introduces degrees of freedom. To allow new local degrees of freedom, we introduce *Truncated Hierarchical B-splines* (THB-Splines) [38, 39].

## 3.2. B-SPLINES

B-spline surfaces $\mathbf{S}(u,v) = [x(u,v),\ y(u,v),\ z(u,v)]$ are piecewise polynomials defined over the parameter domain $(u,v) \in [0,1]^2$ and are given by:

$$\mathbf{S}(u,v) = \sum_{i=0}^{N_u-1}\sum_{j=0}^{N_v-1} \mathbf{c}_{i,j} N_{i,p}(u) N_{j,q}(v), \tag{3.4}$$

where the points $\mathbf{c}_{i,j} = \begin{bmatrix} c^x_{i,j} & c^y_{i,j} & c^z_{i,j} \end{bmatrix}^T$ are called *control points* and influence the shape of the surface through the B-spline basis functions $N_{i,p}$. All the control points together form a *control net*, which determines the overall geometry of the surface.

### 3.2.1. B-SPLINE BASIS FUNCTIONS

The B-spline basis functions are constructed using a *knot vector* $U = \{u_0, \ldots, u_m\}$ and a degree $p$. The knot vector is a sequence of non-decreasing real numbers $u_i \le u_{i+1}$, called *knots*. The interval between neighboring knots $[u_i, u_{i+1})$ is called a *knot span*. Given the degree and knot vector, the B-spline basis functions are constructed using the Cox-de Boor recursion formula [72, Equation 2.5]:

$$N_{i,0}(u) = \begin{cases} 1 & \text{if } u_i \le u < u_{i+1} \text{ and } u_i < u_{i+1} \\ 0 & \text{otherwise} \end{cases}, \tag{3.5}$$

$$N_{i,p}(u) = \frac{u - u_i}{u_{i+p} - u_i} N_{i,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u), \tag{3.6}$$

with $i + p + 1 < m$. If both the numerator and denominator in Equation 3.6 are zero, the fraction is defined as zero ($0/0 = 0$). The basis functions of degrees 0, 1, and 2, constructed using the knot vector $\{0, 0.1, 0.2, 0.3\}$, are illustrated in Figure 3.3. We list several important properties of the B-spline basis functions. A comprehensive list of B-spline basis-functions properties can be found in Section 3.2 of Piegl and Tiller (1996):

**Property 3.2.1 (Local support)** $N_{i,p} = 0$ *outside the interval* $[u_i, u_{i+p+1})$.

**Property 3.2.2** *In any knot span,* $[u_i, u_{i+1})$*, at most* $p+1$ *basis functions are non-zero, namely:* $N_{i-p,p}, \ldots, N_{i,p}$.

**Property 3.2.3 (Non-negativity)** *For all* $i$*,* $p$ *and* $u$*:* $N_{i,p}(u) \ge 0$.

**Property 3.2.4 (Partition of unity)** *For any knot span* $[u_i, u_{i+1})$*,* $\sum_{j=i-p}^{i} N_{j,p}(u) = 1$ *for all* $u \in [u_i, u_{i+1})$.

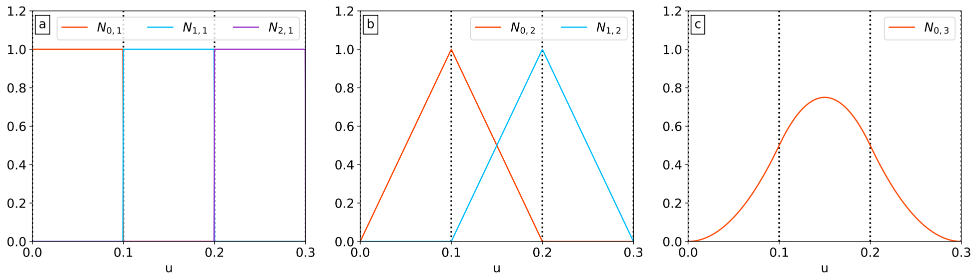


Figure 3.3: B-spline basis functions of various degrees which can be made using the knot vector $\{0, 0.1, 0.2, 0.3\}$: (a) degree 0; (b) degree 1; (c) degree 2.

The basis functions of degree $p$ associated with the knot vector $U$ are linearly independent and thus form a basis for the vector space $\mathscr{B}_{U,p}$.

$$\mathscr{B}_{U,p} = \langle N_{0,p}, \dots, N_{N_u-1,p} \rangle \,. \tag{3.7}$$

The dimension of this vector space, $\dim(\mathscr{B})$, equals the number of basis functions $N_u = m - p - 1$.

## Types of knot vectors

Knot vectors can be classified based on the spacing and repetition of the knots. One classification based on the spacing between knots distinguishes between *uniform* and *non-uniform* knot vectors. A knot vector is uniform when the distance between knots is equally spaced by a value $\Delta u$, such that $u_{i+1} = u_i + \Delta u$. In a non-uniform knot vector, the distance between knots is not constant and can vary arbitrarily. Another classification is based on the repetition of the first and last knots. In this case, a knot vector can be *clamped* (also referred to as open or non-periodic knot vectors). A clamped knot vector has repeated first and last knots. This ensures that the start and end of the B-spline coincide with the first and last control points. For a B-spline of degree $p$, the first and last knots are repeated $p + 1$ times

$$U = \{\underbrace{0, \dots, 0}_{p+1}, u_{p+1}, \dots, u_{m-p-1}, \underbrace{1, \dots, 1}_{p+1}\}. \tag{3.8}$$

Alternatively, a knot vector can be *periodic*, meaning it does not repeat its first and last knots. In this thesis, we primarily use the clamped uniform knot vectors:

$$U = \{\underbrace{0, \dots, 0}_{p+1}, \Delta u, \dots, (m-1)\Delta u, \underbrace{1, \dots, 1}_{p+1}\}, \tag{3.9}$$

as this type of knot vector has a simple relationship between the number of control points $N_u$ and the number of knots $m = N_u + p + 1$. This approach simplifies specifying the number of control points without explicitly defining the knot vector. Thus, specifying the number of control points and the degree $p$ is enough to calculate the knot values using Equation 3.9, where $\Delta u = 1/(N_u - p - 1)$.

### 3.2.2. CONSTRUCTING A B-SPLINE SURFACE

Now that the B-spline basis functions have been defined, we can construct the surface. To do so, we need to specify the degrees $p$ and $q$, the knot vectors $U$ and $V$, and the control points $\mathbf{c}_{i,j} = \begin{bmatrix} c^x_{i,j} & c^y_{i,j} & c^z_{i,j} \end{bmatrix}^T$ where $i \in \{0, \ldots, N_u - 1\}$ and $j \in \{0, \ldots, N_v - 1\}$. The B-spline surface is given by Equation 3.4. By collecting the B-spline basis functions in the vectors

$$\mathbf{N}_p(u) = \begin{bmatrix} N_{0,p}(u) & N_{1,p}(u) & \ldots & N_{N_u-1,p}(u), \end{bmatrix}^T$$
$$\mathbf{N}_q(v) = \begin{bmatrix} N_{0,q}(v) & N_{1,q}(v) & \ldots & N_{N_v-1,q}(v). \end{bmatrix}^T$$

The B-spline surface can be written in matrix form [72, Equation 3.13]

$$\mathbf{S}(u, v) = \mathbf{N}_p(u)^T \mathbf{C} \mathbf{N}_q(v), \tag{3.10}$$

where

$$\mathbf{C} = \begin{bmatrix} \mathbf{c}_{0,0} & \mathbf{c}_{0,1} & \cdots & \mathbf{c}_{0,N_u-1} \\ \mathbf{c}_{1,0} & \mathbf{c}_{1,1} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ \mathbf{c}_{N_v-1,0} & \mathbf{c}_{N_v-1,1} & \cdots & \mathbf{c}_{N_v-1,N_u-1} \end{bmatrix}. \tag{3.11}$$

### 3.2.3. MAPPING $(u, v)$ TO $(x_s, y_s)$

To ensure the B-spline surface uses the coordinates of the lens surface $(x_s, y_s)$, we need an analytically invertible mapping $(u, v) \mapsto (x(u, v), y(u, v))$. To achieve this, the coordinates of the control points are chosen such that the parametrizations $x$ and $y$ are linear:

$$x : u \mapsto (2u - 1) r_x \in [-r_x, r_x], \tag{3.12a}$$
$$y : v \mapsto (2v - 1) r_y \in [-r_y, r_y]. \tag{3.12b}$$

To enforce linearity, we use the nodal representation of the B-spline basis functions [24, Equation (23)]:

$$u = \sum_{i=0}^{N_u-1} u^*_{i,p} N_{i,p}(u), \quad u \in [0, 1], \quad u^*_{i,p} = \frac{u_{i+1} + \ldots + u_{i+p}}{p}, \tag{3.13}$$

which provides a specific knot vector-dependent linear combination of the basis functions. The values $u^*_{i,p}$ are known as the Greville abscissae [30, sec. 8.6]. Assuming $c^x_{i,j}$ is independent of $j$, such that $c^x_{i,0} = c^x_{i,1} = \ldots = c^x_{i,N_u-1}$, we obtain the following definition of $x$:

$$x(u, v) = \sum_{i=0}^{N_u-1} \sum_{j=0}^{N_v-1} c^x_{i,0} \, N_{i,p}(u) N_{j,q}(v) \tag{3.14a}$$
$$= \sum_{i=0}^{N_u-1} c^x_{i,0} \, N_{i,p}(u) \underbrace{\sum_{j=0}^{N_v-1} N_{j,q}(v)}_{=1}. \tag{3.14b}$$

Here, $\sum_{i=0}^{N_u-1} N_{i,p}(u) = 1$ by the partition of unity property (Property 3.2.4). If we define $c^x_{i,j} = u^*_{i,p}$, then $x(u) = u$. Hence, applying the mapping $u \mapsto (2u-1)r_{s,x}$ to both sides of Equation (3.13) yields

$$(2u-1)r_x = \sum_{i=0}^{N_u-1} (2u^*_{i,p}-1)r_{s,x}N_{i,p}(u). \tag{3.15}$$

We can interpret this by expanding 1 into the sum over all $N_{i,p}(u)$ by using the partition of unity. Consequently, if we set $c^x_{i,j} = (2u^*_{i,p}-1)r_{s,x}$ and, similarly, $c^y_{i,j} = (2v^*_{j,q}-1)r_{s,y}$ then Equations (3.12a) and (3.12b) are satisfied. For the arguments of $z(u,v)$ the inverses of $x$ and $y$ are used:

$$u = x^{-1}(x_s) = \frac{1}{2}\left(\frac{x_s}{r_{s,x}}+1\right), \quad v = y^{-1}(y_s) = \frac{1}{2}\left(\frac{y_s}{r_{s,y}}+1\right), \tag{3.16}$$

which allow us to express the B-spline surface in terms of $x_s$ and $y_s$

$$z(x_s,y_s) = \sum_{i=0}^{N_u-1}\sum_{j=0}^{N_v-1} c^z_{i,j}N_{i,p}(x_s)N_{j,q}(y_s). \tag{3.17}$$

## 3.3. B-SPLINE REFINEMENT

In this section, we cover two methods for increasing the number of degrees of freedom in a B-spline surface. The first and most straightforward method is *knot insertion* [72, Chapter 5.2 and 5.3], with which a knot is added to an existing knot vector, thereby increasing the dimension of the associated vector space by one. However, knot insertion has two main drawbacks: it can introduce excessive degrees of freedom and create ambiguity regarding how to refine the surface. An example illustrating these issues will be provided in Section 3.3.2. To address these problems, alternative refinement procedures have been developed, including T-splines [79], U-splines [85], LR-splines [26], *Hierarchical B-splines* (HB-splines) [34], and *Truncated Hierarchical B-splines* (THB-splines) [38, 39]. The last two of these approaches will be covered in Section 3.3.3.

### 3.3.1. KNOT-INSERTION

Consider a knot vector $U = \{u_0,\dots,u_m\}$, which spans a vector space $\mathscr{B}_{U,p}$ of dimension $N_u$. By inserting a unique knot $\tilde{u} \in [u_k, u_{k+1})$ we obtain a new knot vector

$$\widetilde{U} = \left\{\tilde{u}_0 = u_0,\dots,\tilde{u}_k = u_k, \tilde{u}_{k+1} = \tilde{u}, \tilde{u}_{k+2} = u_{k+1},\dots,\tilde{u}_{m+1} = u_m\right\}, \tag{3.18}$$

which spans a vector space $\mathscr{B}_{\widetilde{U},p} = \{\widetilde{N}_{i,p}\},\ i = 0,\dots,N_u+1$. The basis functions $\widetilde{N}_{i,p}$ in $\mathscr{B}_{\widetilde{U},p}$ can be related to the original basis functions $N_{i,p}$ in $\mathscr{B}_{U,p}$ with the following relationship:

$$N_{i,p}(u) = \frac{\tilde{u}-\tilde{u}_{\tilde{i}}}{\tilde{u}_{\tilde{i}+p+1}-\tilde{u}_{\tilde{i}}}\widetilde{N}_{\tilde{i},p}(u) + \frac{\tilde{u}_{\tilde{i}+p+2}-\tilde{u}}{\tilde{u}_{\tilde{i}+p+2}-\tilde{u}_{\tilde{i}+1}}\widetilde{N}_{\tilde{i}+1,p}(u). \tag{3.19}$$

For convenience, we define:

$$\alpha_{\tilde{i}} = \frac{\tilde{u} - \tilde{u}_{\tilde{i}}}{\tilde{u}_{\tilde{i}+p+1} - \tilde{u}_{\tilde{i}}}, \quad \beta_{\tilde{i}} = \frac{\tilde{u}_{\tilde{i}+p+2} - \tilde{u}}{\tilde{u}_{\tilde{i}+p+2} - \tilde{u}_{\tilde{i}+1}}, \tag{3.20}$$

so that

$$N_{i,p}(u) = \alpha_{\tilde{i}} \widetilde{N}_{\tilde{i},p}(u) + \beta_{\tilde{i}} \widetilde{N}_{\tilde{i}+1,p}(u). \tag{3.21}$$

Using Equation 3.19, we can construct a *refinement matrix* ($\mathbf{R}_{\tilde{u}} \in \mathbb{R}^{N_u \times N_u+1}$) that relates the vector of original basis functions $\mathbf{N}_{U,p}$ to the vector of refined basis functions $\mathbf{N}_{\widetilde{U},p}$:

$$\mathbf{N}_{U,p} = \mathbf{R}_{\tilde{u}} \mathbf{N}_{\widetilde{U},p}. \tag{3.22}$$

To ensure the geometry remains unchanged, we combine Equation 3.10 and Equation 3.22:

$$\mathbf{S}(u,v) = \mathbf{N}_{\widetilde{U},p}^T \mathbf{R}_{\tilde{u}}^T \mathbf{C} \mathbf{N}_{V,q}. \tag{3.23}$$

The new control points are then given by $\widetilde{\mathbf{C}} = \mathbf{R}_{\tilde{u}}^T \mathbf{C}$.

### EXAMPLE: KNOT INSERTION

To demonstrate how knot insertion works, we begin with the uniform clamped knot vector

$$U = \{0 \quad 0 \quad 0 \quad 0.25 \quad 0.5 \quad 0.75 \quad 1 \quad 1 \quad 1\},$$

into which we insert the knot $\tilde{u} = 0.4$. This gives the new knot vector

$$\widetilde{U} = \{0 \quad 0 \quad 0 \quad 0.25 \quad 0.4 \quad 0.5 \quad 0.75 \quad 1 \quad 1 \quad 1\}.$$

The basis functions of the two knot vectors are shown in Figure 3.4(a) and Figure 3.4(b).

Using Equation 3.21, we construct the refinement matrix $\mathbf{R}_{\tilde{u}}$, which relates the refined basis $\widetilde{\mathbf{N}}$ to the original basis $\mathbf{N}$:

$$\mathbf{R}_{\tilde{u}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \alpha_1 & \beta_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & \alpha_2 & \beta_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & \alpha_3 & \beta_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3.24}$$

with values $\alpha_1 = 1$, $\alpha_2 = 0.8$, $\alpha_3 = 0.3$, $\beta_1 = 0.2$, $\beta_2 = 0.7$, $\beta_3 = 1$. These values are illustrated in Figure. 3.5, which shows how the refined basis functions $\widetilde{N}_{1,3}$, $\widetilde{N}_{2,3}$, $\widetilde{N}_{3,3}$, and $\widetilde{N}_{4,3}$ combine to recover the original basis functions $N_{1,3}$, $N_{2,3}$, and $N_{3,3}$.

## 3.3.2. PROBLEM: OVER-REFINEMENT

When applying knot insertion to 2D B-splines or surfaces, over-refinement can become problematic. To illustrate this, we consider a B-spline surface of degree 3 in both $u$ and $v$, with knot vectors $U = \{0, 0, 0, 0.25, 0.5, 0.75, 1, 1, 1\}$ and $V = \{0, 0, 0, 0.25, 0.5, 0.75, 1, 1, 1\}$.

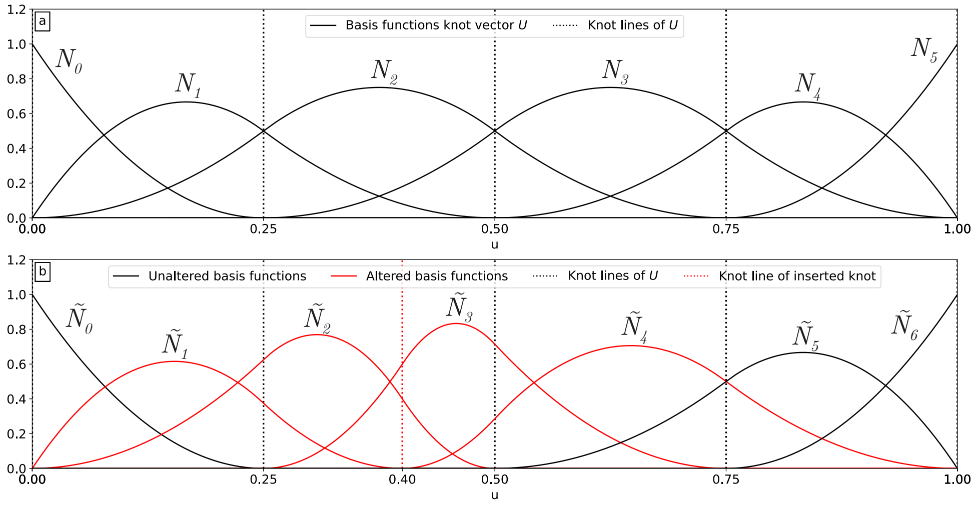


Figure 3.4: (a) The original B-spline basis functions; (b) The basis functions after a knot is inserted.

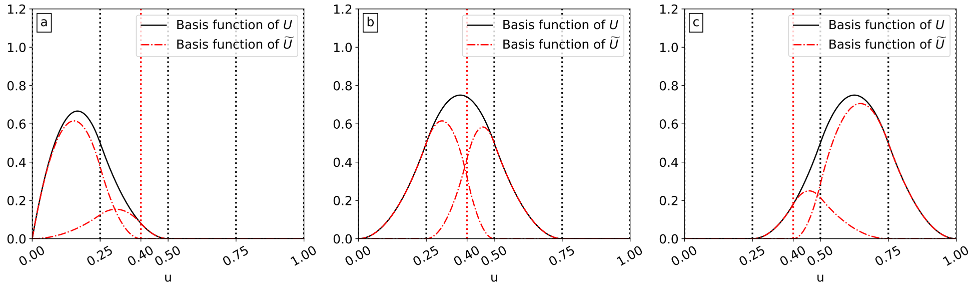


Figure 3.5: (a) Refined basis function $\widetilde{N}_{1,3}$, $\widetilde{N}_{2,3}$ combined to give the unrefined basis function $N_{1,3}$ (b) Refined basis function $\widetilde{N}_{2,3}$, $\widetilde{N}_{3,3}$ combined to give the unrefined basis function $N_{2,3}$ (c) Refined basis function $\widetilde{N}_{3,3}$, $\widetilde{N}_{4,3}$ combined to give the unrefined basis function $N_{3,3}$

Both spaces $\mathscr{B}_U$ and $\mathscr{B}_V$ are 6-dimensional, and their combined spline space $\mathscr{B}_{U\times V}$ has dimension 36. The initial knot span is shown in Figure 3.6(a).
Suppose we want to refine the elements $[0,0.25)\times[0,0.25)$ and $[0.25,0.5)\times[0.5,0.75)$, highlighted in red in Figures 3.6(a-c). One way to accomplish this is by inserting $\tilde{u}=0.4$ into the knot vector $U$. This raises the dimension of $\mathscr{B}_U$ from 6 to 7, so the dimension of $\mathscr{B}_{U\times V}$ then becomes 42. Subsequently, inserting $\tilde{v}=0.1$ into the knot vector $V$ increases the dimension of $\mathscr{B}_{U\times V}$ to 49.
Thus, in order to refine two elements, we end up refining seven elements, adding 13 degrees of freedom when only two were desired. Moreover, there are multiple ways to insert the new knots. For example, we could have inserted $\tilde{u}=0.1$ and $\tilde{v}=0.6$ instead. The issue is that inserting a knot refines an entire row or column of knot spans, introducing redundant control points. Therefore, a method to limit refinement strictly to the desired knot span would be preferable. This is precisely where HB-splines [34] and THB-splines [38] come into play.

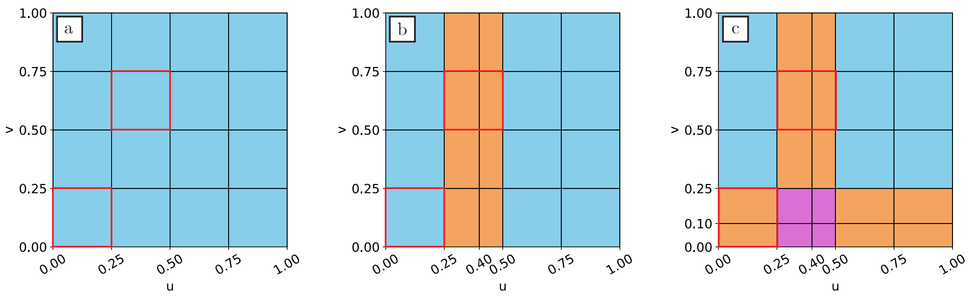


Figure 3.6: Knot spans of the B-spline surface at different refinement steps. The red regions highlight where refinement is desired, orange shows spans refined once, and purple indicates knot spans refined twice: (a) Original knot span; (b) Knot span after the knot $\tilde{u}=0.4$ is inserted; (c) Knot span after the knot $\tilde{v}=0.1$ is inserted;

### 3.3.3. Truncated hierarchical B-splines (THB-splines)

We now walk through the refinement and truncation steps involved in constructing a THB-spline [38, 39]. Specifically, we construct a THB-spline of degree $p=3$ using the knot vector $U^0=\{0,0,0,0.25,0.5,0.75,1,1,1\}$, the level $\ell=0$ knot vector. If we bisect each knot span, dividing it into two equal parts, we obtain the level $\ell=1$ knot vector:

$$U^1=\{0,0,0,\color{red}{0.125},0.25,\color{red}{0.375},0.5,\color{red}{0.625},0.75,\color{red}{0.875},1,1,1\}. \tag{3.25}$$

Using the Cox-de Boor relationship Equation 3.6, we can construct the B-spline basis functions of levels 0 and 1, shown in Figure 3.7. The resulting spaces are:

$$\mathscr{B}^0_u=\{N^0_0(u),\ N^0_1(u),\ \dots,\ N^0_5(u)\}, \tag{3.26}$$
$$\mathscr{B}^1_u=\{N^1_0(u),\ N^1_1(u),\ \dots,\ N^1_9(u)\}. \tag{3.27}$$

Because B-splines are refinable, the basis functions at level $\ell=0$ can be expressed in terms of those at level $\ell=1$. By constructing a refinement matrix Equation 3.22 for the

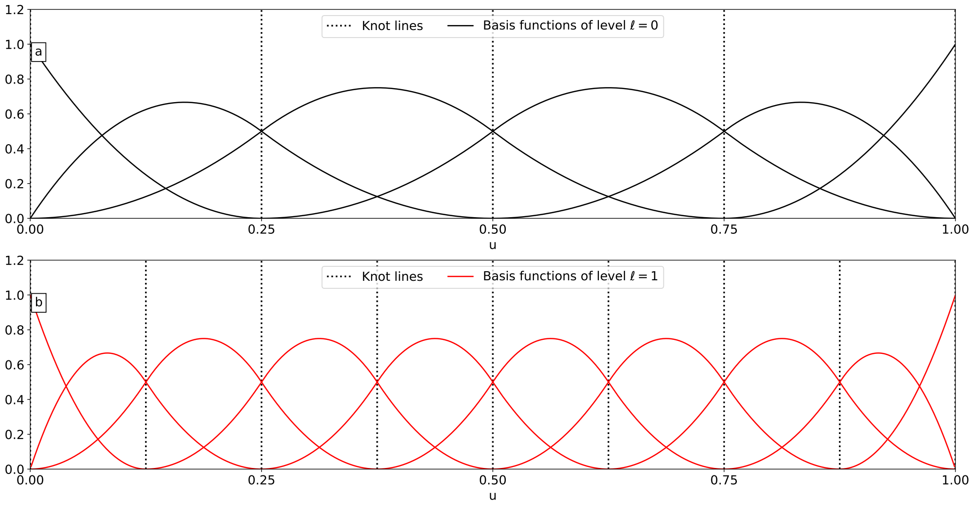


Figure 3.7: B-spline basis functions generated using the Cox-de Boor recursion equation (Equation 3.26) for (a) level $\ell = 0$ with knot vector $U^0$; (b) level $\ell = 1$ with knot vector $U^1$.

inserted knots $\tilde{u}_1 = 0.125$, $\tilde{u}_2 = 0.375$, $\tilde{u}_3 = 0.625$, and $\tilde{u}_4 = 0.875$ we relate $\mathscr{B}_u^0$ to $\mathscr{B}_u^1$ via:

$$\mathbf{N}^0 = \mathbf{R}_{\tilde{u}_1}\mathbf{R}_{\tilde{u}_2}\mathbf{R}_{\tilde{u}_3}\mathbf{R}_{\tilde{u}_4}\mathbf{N}^1. \tag{3.28}$$

For example, the basis functions $N_0^0$, $N_1^0$, and $N_2^0$ decompose into level $\ell = 1$ basis functions as follows:

$$N_0^0(u) = N_0^1(u) + \frac{1}{2}N_1^1(u), \tag{3.29}$$
$$N_1^0(u) = \frac{1}{2}N_1^1(u) + \frac{3}{4}N_2^1(u) + \frac{1}{2}N_3^1(u), \tag{3.30}$$
$$N_2^0(u) = \frac{1}{4}N_2^1(u) + \frac{3}{4}N_3^1(u) + \frac{3}{4}N_4^1(u) + \frac{1}{4}N_5^1(u), \tag{3.31}$$

as illustrated in Figure 3.8.

Suppose we want to refine the domain $[0, 0.5]$. First, we identify which $\ell = 0$ basis functions overlap this region. In our example, the supports of $N_0^0$ and $N_1^0$ are fully contained in $[0, 0.5]$ as shown in Figure 3.9. *Hierarchical B-splines* (HB-splines) refine this region simply by removing those level 0 basis functions and replacing them with level $\ell = 1$ functions, as shown in Figure 3.10:

$$\mathscr{H}_u = \{N_0^1(u), \dots, N_3^1(u), N_2^0(u), \dots, N_5^0(u)\}.$$

However, HB-splines violate the partition of unity property because $N_2^1$ and $N_3^1$ appear twice: once in their own level-1 basis, and again within the decomposition of $N_2^0$. As a result, these functions can have a weight greater than one, which can cause the spline to overshoot control points. Furthermore, the HB-spline basis is not orthogonal, as changing $N_2^0$ also affects $N_2^1$ and $N_3^1$.

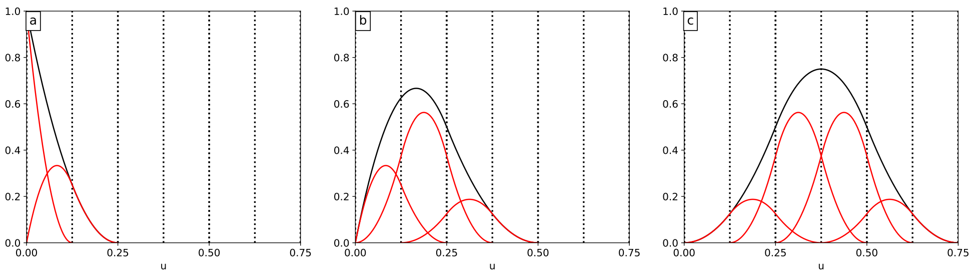


Figure 3.8: (a) The basis function $N_0^0$ decomposed into $N_0^1$ and $N_1^1$; (b) The basis function $N_1^0$ decomposed into $N_1^1$, $N_2^1$ and $N_3^1$; (c) The basis function $N_2^0$ decomposed into $N_2^1$, $N_3^1$, $N_4^1$ and $N_5^1$.

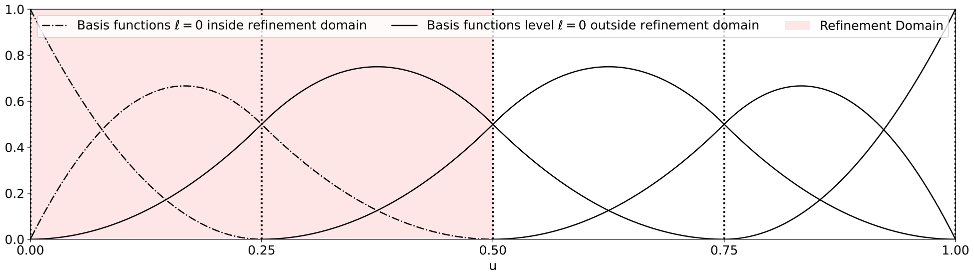


Figure 3.9: B-spline basis functions at level $\ell = 0$ that fall in the refinement domain $[0,\ 0.5]$

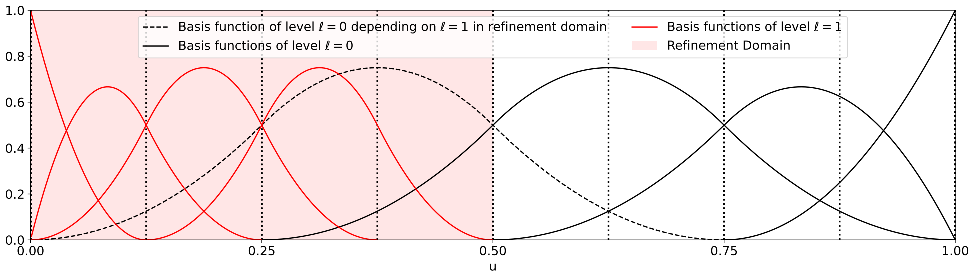


Figure 3.10: Hierarchical B-spline basis functions.

To address these issues, Truncated Hierarchical B-splines (THB-splines) set certain higher-level weights to zero in the lower level functions. In our example, $N_2^0$ depends on $N_2^1$ and $N_3^1$, which fall inside $[0, 0.5]$. We *truncate* $N_2^0$ by zeroing those weights:

$$\operatorname{trunc}\{N_2^0\}(u) = 0N_2^1(u) + 0N_3^1(u) + \frac{3}{4}N_4^1(u) + \frac{1}{4}N_5^1(u). \tag{3.32}$$

See Figure 3.11 and Figure 3.12 for a visual depiction of the process.

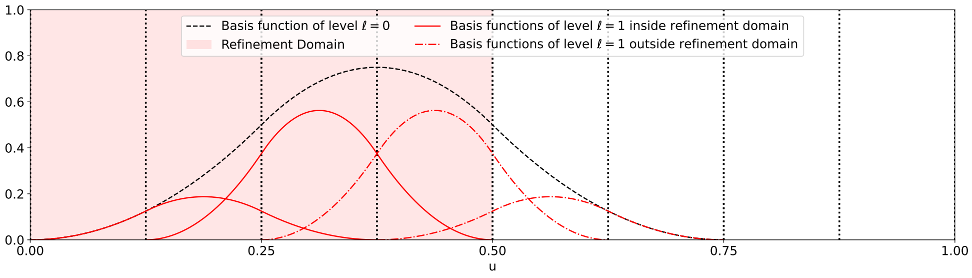


Figure 3.11: Decomposition of $N_2^0$ into level $\ell = 1$ functions. Those inside [0, 0.5] (red solid line) will have weights set to zero; those outside (red dotted line) will remain.

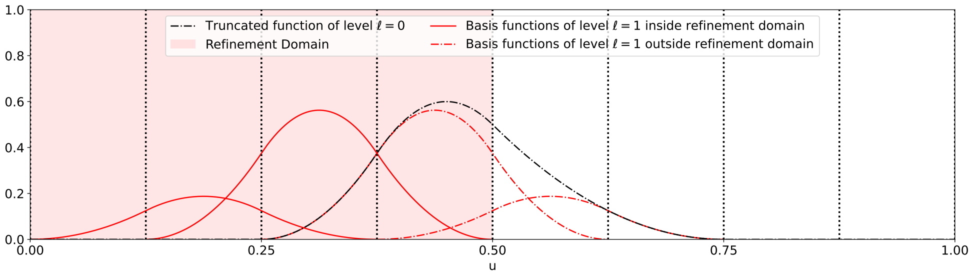


Figure 3.12: The truncated basis function after setting weights of $N_2^1$ and $N_3^1$ to zero.

After truncation, the THB-spline basis replaces $N_2^0$ with $\operatorname{trunc}\{N_2^0\}$. Hence, the final Truncated Hierarchical B-spline basis becomes

$$\mathcal{T}_u = \left\{N_0^1(u), \dots N_3^1(u), \operatorname{trunc}\{N_2^0\}(u), N_3^0(u), N_4^0(u), N_5^0(u)\right\},$$

depicted in Figure 3.13.

## FORMAL DEFINITION OF THB-SPLINE

For a generalized expression of THB-splines, we require a sequence of nested spline spaces of levels $\ell \in \{0, 1, \dots, L\}$, $\mathcal{B}_u^0 \subset \mathcal{B}_u^1 \subset \dots \subset \mathcal{B}_u^L$, over a domain $\Omega_u^0 \subseteq [0,1]$. The highest level $L$ is chosen according to the desired refinement. We start with a knot vector at level $\ell = 0$. Higher-level knot vectors

$$U^{\ell+1} = \left\{u_0^{\ell+1}, \dots, u_m^{\ell+1}\right\}, \tag{3.33}$$

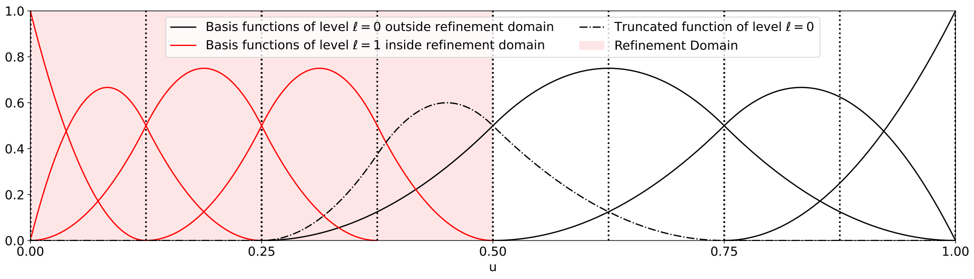


Figure 3.13: Final Truncated Hierarchical B-spline basis functions.

are obtained by bisecting each knot span of the lower level knot vector:

$$U^{\ell+1} = \Big\{u_0^{\ell+1} = u_0^\ell, u_1^{\ell+1} = \frac{u_0^\ell + u_1^\ell}{2}, u_2^{\ell+1} = u_1^\ell, \dots,$$
$$u_{2m-2}^{\ell+1} = u_{m-1}^\ell, u_{2m-1}^{\ell+1} = \frac{u_{m-1}^\ell + u_m^\ell}{2}, u_{2m}^{\ell+1} = u_m^\ell\Big\}. \quad (3.34)$$

We denote the vector of B-spline basis functions of level $\ell$ by $\mathbf{N}^\ell$. Using the refinement matrices (Equation 3.22) we define a matrix $\mathbf{R}_u^{\ell+1} \in \mathbb{R}^{2^\ell N_u \times 2^{\ell+1} N_u}$ as

$$\mathbf{R}_u^{\ell+1} = \prod_{k \in \mathcal{K}} \mathbf{R}_k, \quad \text{with} \quad \mathcal{K} = \left\{\frac{u_0^\ell + u_1^\ell}{2}, \dots, \frac{u_{m-1}^\ell + u_m^\ell}{2}\right\}, \quad (3.35)$$

where $\mathbf{R}_k$ is the refinement matrix related to inserting the $k$th knot. We then relate the level-$\ell$ and level-$\ell+1$ basis vectors by:

$$\mathbf{N}^\ell(u) = \mathbf{R}_u^{\ell+1}\mathbf{N}^{\ell+1}(u). \quad (3.36)$$

For a 2D lens surface, define $\Omega^0 = \Omega_u^0 \times \Omega_v^0$. We use nested domains $\Omega^0 \supseteq \Omega^1 \supseteq \ldots \supseteq \Omega^L$ to indicate where we want higher-level spline functions to be active. Figure 3.14(a2) and Figure 3.14(c2), illustrate an example of nested domains $\Omega_0, \Omega_1, \Omega_2$. The lens domain $\Omega_0$ is shown in Figure 3.14(a2) together with its knot lines, indicating where $\Omega_u$ and $\Omega_v$ are subdivided. Overlaying all these domains produces a hierarchical mesh of the lens surface, shown in Figure 3.14(a1). Next, we define characteristic matrices $\mathbf{X}_u^\ell$ and $\mathbf{X}_v^\ell$ for the basis vectors $\mathbf{N}^\ell(u)$ and $\mathbf{N}^\ell(v)$. These matrices, are used to indicate which basis function fall fully within $\Omega^\ell$ but not in $\Omega^{\ell+1}$. Formally,

$$\mathbf{X}_u^\ell = \operatorname{diag}(x_{u,i}^\ell), \quad x_{u,i}^\ell = \begin{cases} 1, \text{if supp } N_i^\ell(u) \subseteq \Omega_u^\ell \text{ and supp } N_i^\ell(u) \nsubseteq \Omega_u^{\ell+1}, \\ 0, \text{otherwise.} \end{cases} \quad (3.37)$$

In other words, $x_{u,i}^\ell = 1$ when $N_i^\ell(u)$ is fully contained within $\Omega^\ell$ and does not extend into $\Omega^{\ell+1}$. Otherwise, it is 0. Using $\mathbf{X}_u^\ell$ and $\mathbf{X}_v^\ell$ we define the truncation matrix $\mathbf{T}_u^\ell$ [39] as

$$\mathbf{T}_u^\ell = \mathbf{R}_u^\ell(\mathbf{I}^\ell - \mathbf{X}_u^\ell), \quad (3.38)$$

where $\mathbf{I}^\ell$ is the identity matrix. Analogously, we have $\mathbf{T}_v^\ell$ in the $v$-direction. The truncation matrices are used to calculate the THB-coefficient matrix $\mathbf{D}^\ell$

$$\mathbf{D}^\ell = \mathbf{T}_u^{\ell T}\mathbf{D}^{\ell-1}\mathbf{T}_v^\ell + \mathbf{X}_u^{\ell T}\mathbf{C}^\ell\mathbf{X}_v^\ell, \tag{3.39}$$

starting with $\mathbf{D}^0 = \mathbf{C}$. Thus, a THB-spline $\mathbf{S}_{\mathrm{THB}}$ at the highest level $L$ is expressed via the level-$L$ basis:

$$\mathbf{S}_{\mathrm{THB}}(u,v) = \mathbf{N}^L(u)^T\mathbf{D}^L\mathbf{N}^L(v). \tag{3.40}$$

An example of a hierarchical mesh is depicted in Figure 3.14(a1), with the corresponding nested domains in Figure 3.14(a2-c2). The resulting THB-spline is shown in Figure 3.14(b1), where the higher refinement levels exhibit finer surface details.

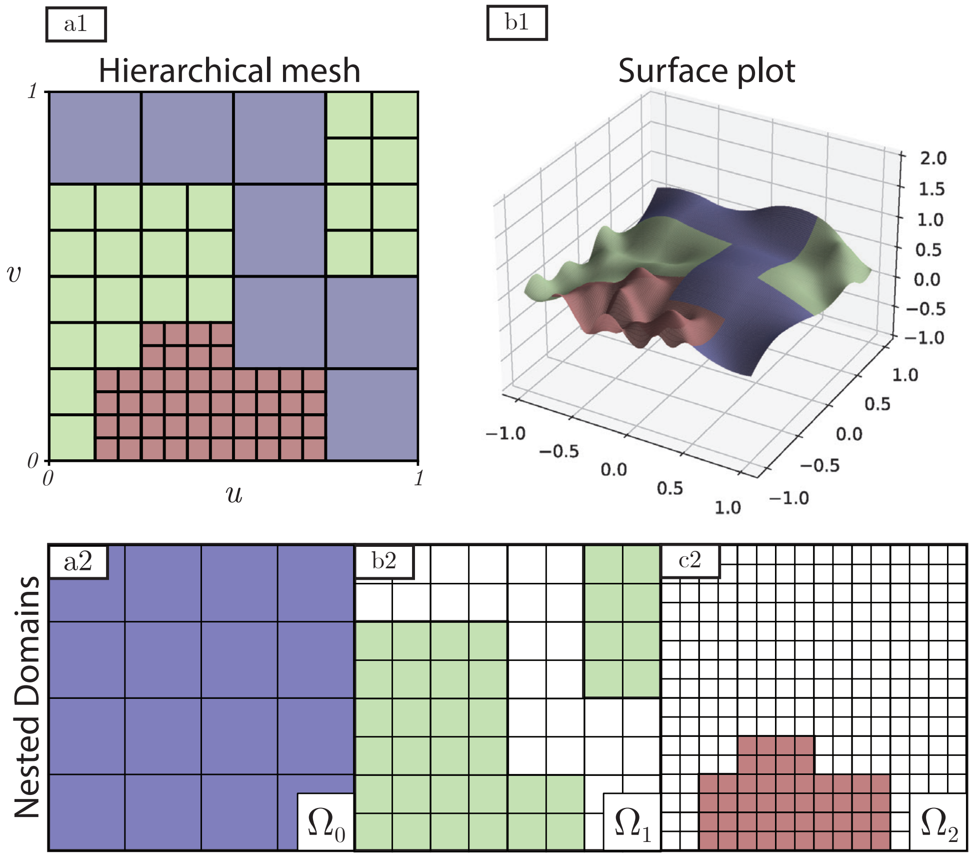


Figure 3.14: (a1) Hierarchical mesh; (b1) A THB-spline surface with random $z$-positions of control points; (a2-c2) The nested domains $\Omega_0$, $\Omega_1$, $\Omega_2$. Colored regions indicate where higher-level domains are active.

### NOTE: REFINEMENT WITHOUT NEW DEGREES OF FREEDOM

Sometimes, refining a knot span does not introduce new degrees of freedom. This happens when the support of the THB-spline of the current level and the lower level do not fall within the support of the refinement domain.

For instance, consider the example discussed in Section 3.3.3, but now we refine the area $\Omega_r = [0.25, 0.5)$. If none of the basis functions at levels $\ell = 0$ or $\ell = 1$ lie within $\Omega_r$, the knot span is subdivided, but no truncation or addition of new basis functions occurs.

However, if we refine the domain $\Omega_r$ to level $\ell = 2$, there are basis functions of level $\ell = 2$ which fall fully within $\Omega_r$, and we gain new control points.

### 3.3.4. SUMMARY

In this chapter, we have shown how freeform surfaces can be described using B-splines, HB-splines, and THB-splines. We began by defining B-splines, outlining how to construct their basis functions from a chosen knot vector and degree, and explaining how these functions, together with control points, form a B-spline surface. We then demonstrated how to map B-spline spaces back to the physical coordinates of the lens surface.

Next, we discussed how to refine B-spline surfaces via knot insertion, highlighting the potential over-refinement problem it can create. To address this, we introduced HB-splines and THB-splines, provided a step-by-step explanation of their construction, and presented a more formal definition of THB-splines. This theory will be used in later chapters to model freeform surfaces and to allow local adjustments and refinement.

# 4

# Optimization and Algorithmic Differentiation

Optimization is the process of finding the best possible solution for a given objective function. This objective can range from finding the shortest route between two locations to maximizing the return on a stock portfolio. To achieve this, we define a set of parameters that influence the outcome, such as the available roads while navigating or the selection of stocks in a portfolio. The next step is to establish a quantitative measure, often called an objective function, that allows us to evaluate and compare different parameter combinations. Finally, an optimization algorithm is employed to systematically explore and identify the combination of parameters that yields the best possible outcome. In this chapter, we begin by defining unconstrained optimization and introducing the primary optimization algorithm used in this thesis: gradient descent. We will discuss different strategies for selecting the step size and methods for computing gradients, including analytical derivation, numerical differentiation, and algorithmic differentiation. We will conclude with a brief discussion on constrained optimization and various techniques for incorporating constraints, such as penalty methods and parameter weighting.

## 4.1. Mathematical formulation

The *objective function*, $\mathscr{L}$, is a scalar function depending on $N_p$ parameters, $\boldsymbol{\pi} \in \mathbb{R}^{N_p}$. It can take any form, but often we want to compare an outcome $\mathbf{g}$, which depends on $\boldsymbol{\pi}$, with some desired outcome $\mathbf{g}_{\text{des}}$. Often, this is done by using norms. The most common one for vectors is the $L_2$-norm, which for a vector $\mathbf{a} = [a_1,\ a_2,\ \dots,\ a_N]$ is defined as follows:

$$\|\mathbf{a}\|_2 \equiv \sqrt{\sum_{i=1}^{N} a_i^2}. \tag{4.1}$$

An example of an objective function using the $L_2$ norm is

$$\mathscr{L}(\boldsymbol{\pi}) = \left\| \mathbf{g}_{\text{des}} - \mathbf{g}(\boldsymbol{\pi}) \right\|_2 . \tag{4.2}$$

To compare matrices, we use the *Frobenius norm*, which for a matrix

$$\mathbf{A} = \begin{bmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,N} \\ a_{2,1} & a_{2,2} & \dots & a_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{M,1} & a_{M,2} & \dots & a_{M,N} \end{bmatrix}, \tag{4.3}$$

is defined as

$$\|\mathbf{A}\|_F = \sqrt{\sum_{i=1}^{N} \sum_{j=1}^{M} a_{i,j}^2}. \tag{4.4}$$

Once we have chosen the objective function, our goal is to find the set of parameters $\boldsymbol{\pi}^*$ that minimizes the function $\mathscr{L}$,

$$\boldsymbol{\pi}^* = \arg\min_{\boldsymbol{\pi}} \mathscr{L}(\boldsymbol{\pi}). \tag{4.5}$$

We can distinguish between two types of minimizers: *global minimizer* and *local minimizer*. A point $\boldsymbol{\pi}^*$ is a local minimizer if there is a neighborhood $\mathscr{N}$ of $\boldsymbol{\pi}^*$ such that $\mathscr{L}(\boldsymbol{\pi}^*) \le \mathscr{L}(\boldsymbol{\pi})$ for all $\boldsymbol{\pi} \in \mathscr{N}$. A global minimizer is a point $\boldsymbol{\pi}^*$ such that $\mathscr{L}(\boldsymbol{\pi}^*) \le \mathscr{L}(\boldsymbol{\pi})$ for all $\boldsymbol{\pi}$. Ideally, we would always find the global minimum. However, no method guarantees that the global minimum will be found, and often, there is no way to show that a minimum is indeed the global minimizer. Therefore, most gradient-based optimization algorithms are geared towards finding local minima. While methods such as simulated annealing, particle swarm optimization and genetic algorithms aim to find the global minimum, they are beyond the scope of this thesis. Instead, we focus on gradient descent, a widely used method for local optimization.

## 4.2. GRADIENT DESCENT

*Gradient descent*, in its simplest form, finds local minima by iteratively stepping in the opposite direction of the gradient, as this is the direction of steepest decrease of $\mathscr{L}$. At iteration $i$, the parameter vector $\boldsymbol{\pi}_i$ is updated as:

$$\boldsymbol{\pi}_{i+1} = \boldsymbol{\pi}_i - \alpha_i \nabla \mathscr{L}(\boldsymbol{\pi}_i), \tag{4.6}$$

where $\alpha_i$ is the *step size* at the $i$th iteration.

Two essential components for implementing gradient descent are computing the gradient

$$\nabla \mathscr{L}(\boldsymbol{\pi}) = \begin{bmatrix} \frac{\partial \mathscr{L}}{\partial \pi_1} & \frac{\partial \mathscr{L}}{\partial \pi_2} & \cdots & \frac{\partial \mathscr{L}}{\partial \pi_{N_p}} \end{bmatrix}^T, \tag{4.7}$$

(discussed in Section 4.3) and choosing a strategy for the step size $\alpha_i$.

The simplest approaches consider a constant or decaying step size. A constant step size is achieved by keeping $\alpha$ fixed and decaying step size is achieved by updating the step size every iteration with: $\alpha_{i+1} = \alpha_0 a^i$ with the decay factor $0 < a < 1$. However, large step sizes might risk divergence or strong oscillation, while small ones lead to slow convergence. In certain problems, information about previous gradients can help smooth out updates. One straightforward example is the Momentum [29], which keeps a moving average of past gradients:

$$m_{i+1} = \beta_1 m_i + (1-\beta_1)\nabla\mathscr{L}(\boldsymbol{\pi}_i), \tag{4.8}$$

with $\beta_1 \in [0,1)$ the momentum decay. Each iteration the parameters are then updated as follows:

$$\boldsymbol{\pi}_{i+1} = \boldsymbol{\pi}_i - \alpha_0 m_{i+1}. \tag{4.9}$$

This method reduces oscillations and accelerates in consistent descent directions.

Adaptive step size approaches, such as RMSprop [45], address the situation where different parameters can have very different sensitivities. By tracking a moving average of the squared gradients,

$$v_{i+1} = \beta_2 v_i + (1-\beta_2)\left[\nabla\mathscr{L}(\boldsymbol{\pi}_i)\right]^2, \tag{4.10}$$

with $\beta_2 \in [0,1)$ the decay factor for the squared gradients. Each parameter is updated proportionally to $1/\sqrt{v_{i+1}}$, balancing out large and small gradients:

$$\boldsymbol{\pi}_{i+1} = \boldsymbol{\pi}_i - \frac{\alpha_0}{\sqrt{v_{i+1}}+\epsilon}\nabla\mathscr{L}(\boldsymbol{\pi}_i). \tag{4.11}$$

The Adam method [49] combines the Momentum and RMSProp approaches. Because the moving averages of both the gradient and squared gradient are initialized to zero, they are biased toward zero for the first few iterations. To counteract this, unbiased estimates are introduced: $\hat{m}_i i + 1 = m_{i+1}/(1-\beta_1^{i+1})$, $\hat{v}_{i+1} = v_{i+1}/(1-\beta_2^{i+1})$ and the parameters updated as follows

$$\boldsymbol{\pi}_{i+1} = \boldsymbol{\pi}_i - \alpha_0\frac{\hat{m}_{i+1}}{\sqrt{\hat{v}_{i+1}}+\epsilon}, \tag{4.12}$$

where $\epsilon$ is a small number preventing the division by zero for the case that $\hat{v}_i \approx 0$ .
Finally, a line search can be performed to determine the optimal step size by minimizing $\mathscr{L}$ along the current gradient direction:

$$\alpha_i = \arg\min_{\alpha_i} \mathscr{L}\left(\boldsymbol{\pi}_i - \alpha_i\nabla\mathscr{L}\left(\boldsymbol{\pi}_i\right)\right). \tag{4.13}$$

Although line searches tend to give good step sizes, they can be computationally more expensive per iteration as multiple function evaluations are required. In practice, problems with a large number of parameters make use of constant, decaying, momentum-based, or adaptive approaches for their simplicity and efficiency.

## 4.3. CALCULATING GRADIENTS

There are several ways to calculate the gradient for gradient descent, such as symbolic differentiation, numerical differentiation, and algorithmic differentiation. We will use an example to demonstrate how these methods work. In this example, we scan a surface with varying heights, as depicted in Figure 4.1. At every position, the focus must be adjusted so that the image is formed on the detector at a distance $d_i$ behind the lens. We use an adjustable lens with a tunable focal length $f$ to achieve this. We choose the ob-

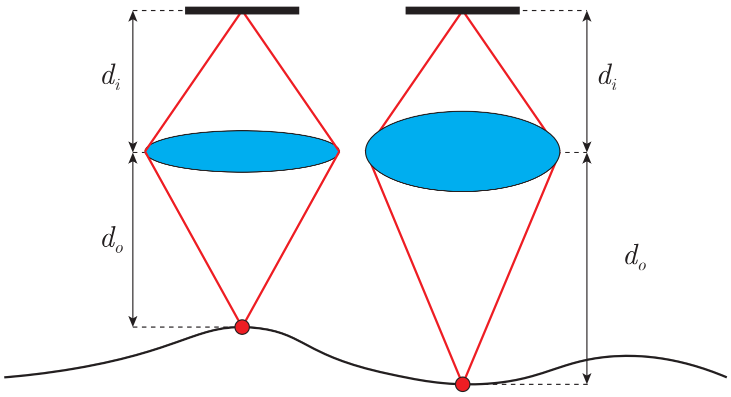


Figure 4.1: Example problem of varying image and focal length positions.

jective function to be the squared difference between the current image $d_i$ position and the desired image position $d_i^*$

$$\mathcal{L}\left(f, d_o\right) = \left(d_i^* - d_i\right)^2. \tag{4.14}$$

The desired image position follows from the thin lens equation:

$$d_i^* = \frac{d_o f}{d_o - f}, \tag{4.15}$$

where the parameters $\boldsymbol{\pi} = [f, d_o]$ are used to obtain the desired image position.

### 4.3.1. SYMBOLIC DIFFERENTIATION

For some problems, the gradient can be calculated by hand, allowing an explicit implementation in code. This is the case for the example problem, where the gradients with respect to the parameters $f$ and $d_o$ are:

$$\frac{\partial \mathcal{L}}{\partial f} = \frac{2d_o^2}{(d_o - f)^2}\left[\frac{d_o f}{d_o - f} - d_i\right], \tag{4.16}$$

$$\frac{\partial \mathcal{L}}{\partial d_o} = \frac{-2f^2}{(d_o - f)^2}\left[\frac{d_o f}{d_o - f} - d_i\right]. \tag{4.17}$$

In addition, having an analytic expression for the gradient can provide valuable insights into parameter sensitivity and how they affect each other. For simple cases, explicit derivatives can reveal how different parameters influence the objective function. However, for problems with many parameters or highly nonlinear objective functions, computing derivatives by hand can be impractical or even impossible. In these cases, programs like Maple [60] and Mathematica [95] are efficient tools for calculating symbolic derivatives.

## 4.3.2. NUMERICAL DIFFERENTIATION

If an objective function cannot be symbolically differentiated or involves a numerical simulation, its derivative can be approximated using the finite-difference approximation. Small perturbations of the parameter vector $\delta\boldsymbol{\pi}$ are used to approximate the gradient, based on the Taylor expansion of the loss function:

$$\mathscr{L}(\boldsymbol{\pi}+\delta\boldsymbol{\pi}) = \mathscr{L}(\boldsymbol{\pi}) + \nabla\mathscr{L}(\boldsymbol{\pi})^T\delta\boldsymbol{\pi} + \frac{1}{2}\delta\boldsymbol{\pi}^T\nabla^2\mathscr{L}(\boldsymbol{\pi})\delta\boldsymbol{\pi}. \tag{4.18}$$

By setting a bound $L$ on $\|\nabla^2\mathscr{L}\|$ in the region of interest, the last term of Equation 4.18 is bounded by $L/2\|\delta\boldsymbol{\pi}\|^2$, and:

$$\|\mathscr{L}(\boldsymbol{\pi}+\delta\boldsymbol{\pi}) - \mathscr{L}(\boldsymbol{\pi}) - \nabla\mathscr{L}(\boldsymbol{\pi})^T\delta\boldsymbol{\pi}\| \leq \frac{L}{2}\|\delta\boldsymbol{\pi}\|^2. \tag{4.19}$$

If we choose the perturbation as $\delta\boldsymbol{\pi} = \varepsilon\mathbf{e}_i$, where $\mathbf{e}_i$ is the $i$th standard basis vector and $\varepsilon$ is a small increment, we get:

$$\nabla\mathscr{L}(\boldsymbol{\pi})^T\delta\boldsymbol{\pi} = \varepsilon\nabla\mathscr{L}(\boldsymbol{\pi})^T\mathbf{e}_i = \varepsilon\frac{\partial\mathscr{L}}{\partial\pi_i} \tag{4.20}$$

As $\|\delta\boldsymbol{\pi}\|^2 = \varepsilon^2$, we can rearrange Equation 4.18 into

$$\frac{\partial\mathscr{L}}{\partial\pi_i}(\boldsymbol{\pi}) = \frac{\mathscr{L}(\boldsymbol{\pi}+\varepsilon\mathbf{e}_i) - \mathscr{L}(\boldsymbol{\pi})}{\varepsilon} + \frac{L}{2}\varepsilon. \tag{4.21}$$

If $\varepsilon$ approaches zero, the higher order terms become negligible, and Equation 4.21 will approximately become

$$\frac{\partial\mathscr{L}}{\partial\pi_i}(\boldsymbol{\pi}) \approx \frac{\mathscr{L}(\boldsymbol{\pi}+\varepsilon\mathbf{e}_i) - \mathscr{L}(\boldsymbol{\pi})}{\varepsilon}. \tag{4.22}$$

However, a couple of issues have to be addressed when using this method. If $\varepsilon$ is too small, round-off error becomes dominant. Conversely, if $\varepsilon$ is chosen too large, the approximation deviates too much from the true gradient, as shown in Figure 4.2(a). A more accurate approximation, such as the central difference method, can reduce these errors. In this case, the gradient is approximated as follows:

$$\frac{\partial\mathscr{L}}{\partial\pi_i}(\boldsymbol{\pi}) \approx \frac{\mathscr{L}(\boldsymbol{\pi}+\varepsilon\mathbf{e}_i) - \mathscr{L}(\boldsymbol{\pi}-\varepsilon\mathbf{e}_i)}{2\varepsilon}. \tag{4.23}$$

As shown in Figure 4.2(b), the central difference method reduces errors compared to the forward difference. However, for a function with $n$ parameters, the forward difference method requires $n+1$ function evaluations (one for each perturbed parameter plus the baseline evaluation), while the central difference method requires $2n$ evaluations, making it computationally more expensive.

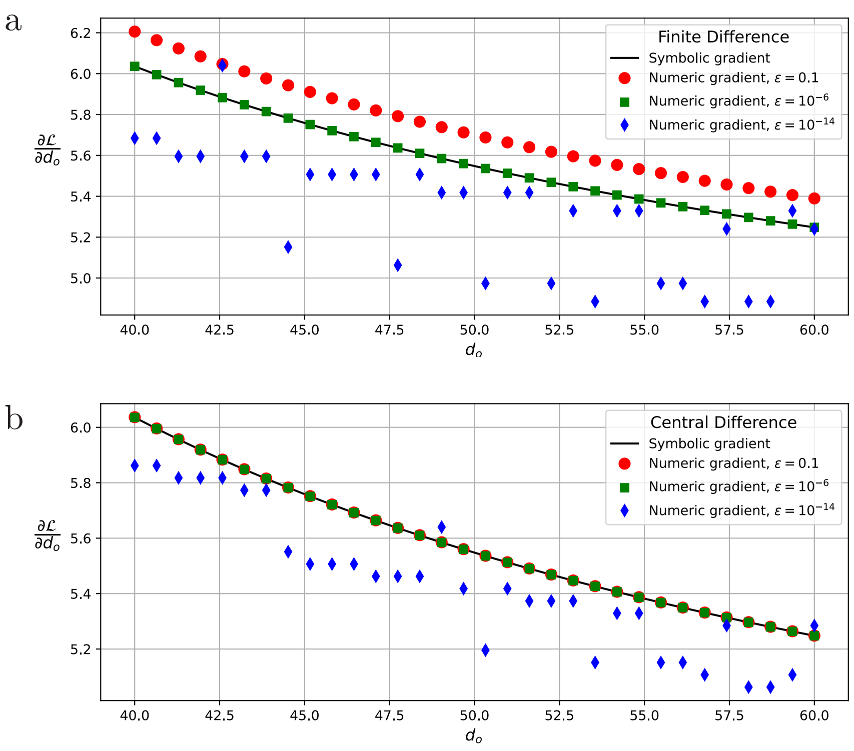


Figure 4.2: Numerical gradient for the focal length in the example problem, for different values of $d_o$, a current focal length $f = 4$, and $d_i = 2$ (a) using finite difference; (b) using central difference.

### 4.3.3. ALGORITHMIC DIFFERENTIATION

*Algorithmic differentiation* or *automatic differentiation* calculates gradients using specialized software libraries, such as PyTorch [69], TensorFlow [61], and JAX [14]. Algorithmic differentiation is based on two core concepts. First, most functions can be expressed using simple operations on one or more variables, including addition, multiplication, logarithms, powers, and trigonometric functions. Second, the chain rule allows us to compute derivatives of composite functions. The chain rule states that if $h$ is a function of a vector $\mathbf{y} \in \mathbb{R}^m$, which depends on $\mathbf{x} \in \mathbb{R}^n$, we can compute the gradient of $h$ with respect to $\mathbf{x}$ as:

$$\nabla h(\mathbf{y}(\mathbf{x})) = \sum_{i=1}^{m} \frac{\partial h}{\partial y_i} \nabla y_i(\mathbf{x}). \tag{4.24}$$

A computational graph is a useful tool for visualizing algorithmic differentiation, as depicted in Figure 4.3. It breaks down the function into its elementary operations and illustrates dependencies between parameters and gradients. We define the independent variables $c = -d_i$, $v_1 = d_o$, $v_2 = f$, and at each intermediate step, we introduce intermediate variables $v_3, v_4, \ldots, v_8$. The final variable, $v_9$, on the right of the graph, is the function value $\mathscr{L}(\boldsymbol{\pi})$. An overview of all the variables and their full expression is given in Table 4.1.

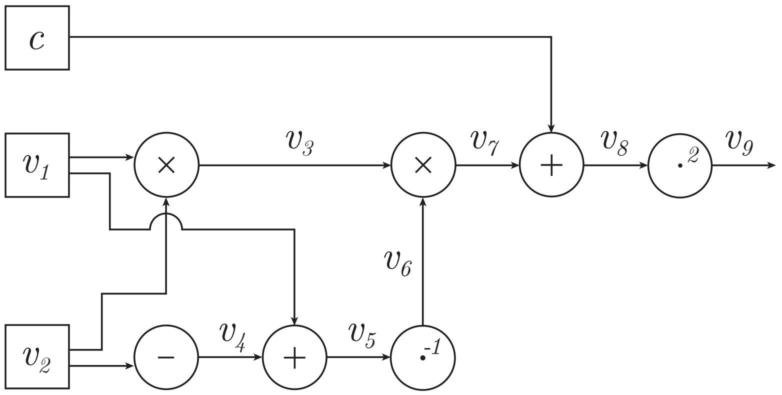


Figure 4.3: Computational graph for the example problem, Equation 4.14.

We limit the discussion on algorithmic differentiation to an overview of the mathematics for both the *forward mode differentiation* and the *reverse mode differentiation*. For a much more in-depth discussion on mathematics and computer science, the interested reader is referred to Griewank and Walther (2008).

#### FORWARD MODE DIFFERENTIATION

In forward mode differentiation, we evaluate the directional derivative $\nabla_{\mathbf{q}}$, along $\mathbf{q}$, as follows:

$$\nabla_{\mathbf{q}} \mathscr{L}(\boldsymbol{\pi}) = \mathbf{q} \cdot \nabla \mathscr{L}(\boldsymbol{\pi}). \tag{4.25}$$

The directional derivative can be evaluated simultaneously with the function $\mathscr{L}$ itself. Thus, when the value of $v_i$ at any node is known, its directional derivative $\nabla_{\mathbf{q}} v_i$ can also be computed using the chain rule. As the computation progresses, both parameters, $v_i$

Table 4.1: All the independent and intermediate variables of the example problem.

| Intermediate variables | Expressed in intermediate variables | Full expression |
|---|---|---|
| $c$ | $-d_i$ | $-d_i$ |
| $v_1$ | $d_o$ | $d_o$ |
| $v_2$ | $f$ | $f$ |
| $v_3$ | $v_1 \times v_2$ | $d_o \times f$ |
| $v_4$ | $-v_2$ | $-f$ |
| $v_5$ | $v_1 + v_4$ | $d_o - f$ |
| $v_6$ | $(v_5)^{-1}$ | $\frac{1}{d_o - f}$ |
| $v_7$ | $v_3 \times v_6$ | $\frac{d_o \times f}{d_o - f}$ |
| $v_8$ | $v_7 + c$ | $\frac{d_o \times f}{d_o - f} - d_i$ |
| $v_9$ | $(v_8)^2$ | $\left(\frac{d_o \times f}{d_o - f} - d_i\right)^2$ |

and their gradients $\nabla v_i$ propagate through the computational graph simultaneously, as illustrated in Figure 4.4. Consider computing the directional derivative of $v_3 = v_1 \times v_2$. The gradient is then given by:

$$\nabla v_3 = \frac{\partial v_3}{\partial v_1}\nabla v_1 + \frac{\partial v_3}{\partial v_2}\nabla v_2. \tag{4.26}$$

As the values of $v_1$, $v_2$, $\nabla_{\mathbf{q}} v_1$ and $\nabla_{\mathbf{q}} v_2$ are known we can evaluate the directional derivative

$$\nabla_{\mathbf{q}} v_3 = v_2 \nabla_{\mathbf{q}} v_1 + v_1 \nabla_{\mathbf{q}} v_2, \tag{4.27}$$

simultaneously with evaluating $v_3$. By recursively applying this process, we eventually obtain $\nabla_{\mathbf{p}} \mathscr{L}$. The gradients of all intermediate variables in the computational graph are summarized in Table 4.2. Forward mode differentiation is most efficient when the number of objective functions exceeds the number of parameters. To compute the full gradient vector $\nabla \mathscr{L}$, we must traverse the computational graph once per parameter, computing the directional derivative along each direction $\mathbf{q} = \mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_n$.

### 4.3.4. REVERSE MODE DIFFERENTIATION

Reverse mode differentiation does not compute gradients while evaluating $\mathscr{L}$. Instead, it computes the derivatives of $\mathscr{L}$ with respect to each parameter $\pi_i$ by performing a reverse sweep. $\nabla \mathscr{L}$ is obtained by accumulating all partial derivatives $\bar{v}_i = \partial \mathscr{L} / \partial v_i$, called

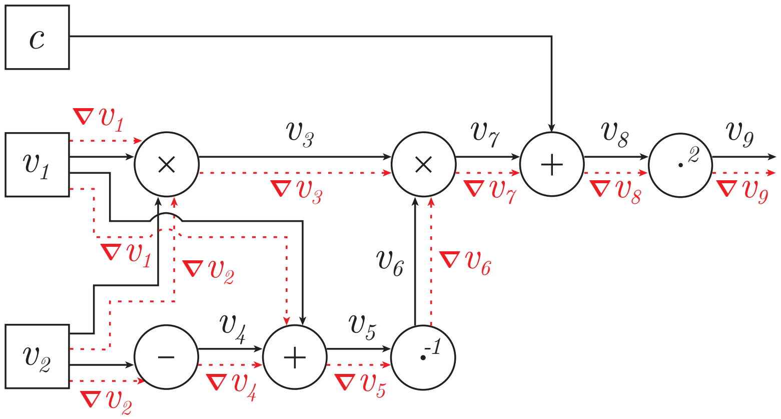


Figure 4.4: Illustration of forward pass of the computational graph of the example problem. The intermediate variables (black) and directional derivatives (red) are calculated simultaneously.

Table 4.2: Gradients of the example problem expressed in terms of the intermediate variables.

| Intermediate Variable | Gradient Expression | Intermediate Variable | Gradient Expression |
|---|---|---|---|
| $\nabla v_1$ | $\nabla v_1$ | $\nabla v_6$ | $-(v_5)^{-2}\nabla v_5$ |
| $\nabla v_2$ | $\nabla v_2$ | $\nabla v_7$ | $v_3\nabla v_6 + v_6\nabla v_3$ |
| $\nabla v_3$ | $v_1\nabla v_2 + v_2\nabla v_1$ | $\nabla v_8$ | $\nabla v_7$ |
| $\nabla v_4$ | $-\nabla v_2$ | $\nabla v_9$ | $2\nabla v_8$ |
| $\nabla v_5$ | $\nabla v_2 + \nabla v_4$ | | |

*adjoint variables*, at each node in the graph. The adjoint $\bar{v}_i$ is calculated as follows

$$\bar{v}_i = \sum_{j \in \mathscr{C}} \bar{v}_j \frac{\partial v_j}{\partial v_i}, \tag{4.28}$$

where $\mathscr{C}$ represents all the variables that directly depend on $v_i$ referred to as child variables. For instance, the adjoint $\bar{v}_2$ has two child variables $v_3$ and $v_4$. All adjoint variables are initially set to zero, except $\bar{v}_9$, which is set to 1. We can use Equation 4.28 to calculate $\bar{v}_2$

$$\bar{v}_2 = \bar{v}_3 \frac{\partial v_3}{\partial v_2} + \bar{v}_4 \frac{\partial v_4}{\partial v_2} \tag{4.29}$$

which equals:

$$\bar{v}_2 = \bar{v}_3 v_1 - \bar{v}_4. \tag{4.30}$$

All the adjoint variables of the computational graph are shown in Table 4.3. The computational graph, illustrating the dependency of parameters and the backward propagation adjoints to the independent variables, is shown in Figure 4.5. Reverse mode differentiation is more efficient when the number of parameters is significantly greater than the number of objective functions.

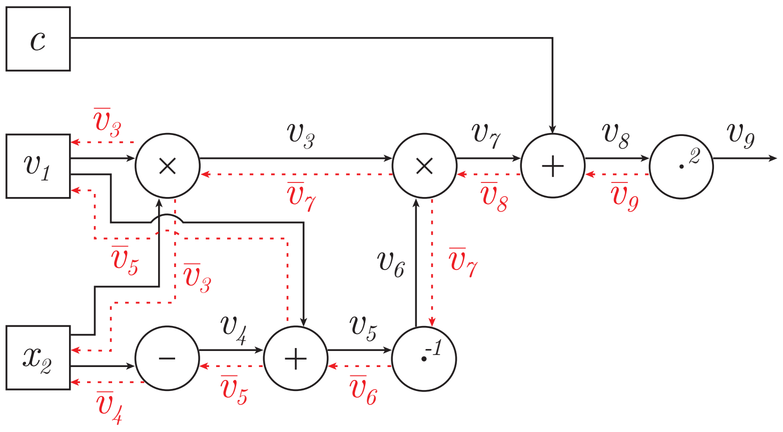


Figure 4.5: Illustration of forward pass and reverse sweep of the computational graph of the example problem. The intermediate variables (black) are calculated in the forward pass, and the adjoints (red) are calculated in the reverse sweep.

## 4.4. CONSTRAINED OPTIMIZATION

When optimizing an objective function, we often impose constraints that the parameters must satisfy. The functions $c_i$ are scalar functions of the variables $\boldsymbol{\pi}$, which define the constraint and can be equality or inequality constraints, given by:

$$\min_{\boldsymbol{\pi}} \mathscr{L}(\boldsymbol{\pi}) \qquad \text{subject to} \begin{cases} c_i(\boldsymbol{\pi}) = 0, & i \in \mathscr{E}, \\ c_i(\boldsymbol{\pi}) \geq 0, & i \in \mathscr{I}. \end{cases} \tag{4.31}$$

Table 4.3: Adjoint variables of the example problem expressed in terms of the intermediate variables.

| Adjoint Variable | Expression | Adjoint Variable | Expression |
|---|---|---|---|
| $\bar{v}_1$ | $v_2\bar{v}_3 + v_4\bar{v}_5$ | $\bar{v}_6$ | $v_3\bar{v}_7$ |
| $\bar{v}_2$ | $v_1\bar{v}_3 - \bar{v}_4$ | $\bar{v}_7$ | $\bar{v}_8$ |
| $\bar{v}_3$ | $v_6\bar{v}_7$ | $\bar{v}_8$ | $2v_8\bar{v}_9$ |
| $\bar{v}_4$ | $v_1\bar{v}_5$ | $\bar{v}_9$ | 1 |
| $\bar{v}_5$ | $-\bar{v}_6(v_5)^{-2}$ | | |

Here, $\mathscr{E}$ is the set of indices of the equality constraints, and $\mathscr{I}$ is the set of indices of the inequality constraints. We briefly discuss two approaches to constrained optimization, the first approach uses *weighted variables* while the second incorporates constraints through *regularization.*

### 4.4.1. Weighted variables

When a variable is constrained to a domain $-b_i \leq \pi_i \leq b_i$, we can reformulate the constrained optimization problem as an unconstrained optimization by applying a function that limits the range of the variable. This is for instance, withtanh or arctan, as shown in Figure 4.6. The function ensures parameter values do not exceed the given bounds by optimizing $\mathbf{w}$. Thus, Equation 7.3 becomes an unconstrained optimization problem

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \mathscr{L}(\mathbf{w}), \tag{4.32}$$

where

$$\pi_i^* = b_i \tanh\left(w_i^*\right). \tag{4.33}$$

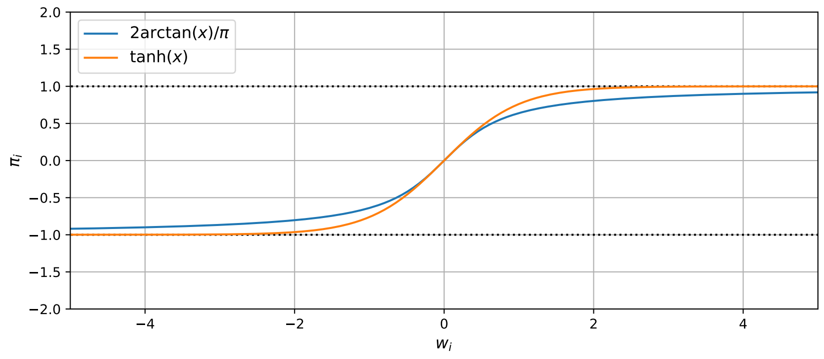


Figure 4.6: Ranges of the functions tanh and arctan

### 4.4.2. Regularized optimization

A constrained optimization problem can be reformulated as an unconstrained problem using regularization, which incorporates the constraints into the objective function. The new unconstrained optimization problem then becomes

$$\min_{\boldsymbol{\pi}} \mathscr{L}(\boldsymbol{\pi}) + \mu c(\boldsymbol{\pi}), \tag{4.34}$$

where $\mu$ is a parameter controlling the importance of the constraint. *Tikhonov regularization* [86], is a common regularization technique that sets the regularization term as:

$$c(\boldsymbol{\pi}) = \|\boldsymbol{\pi}\|_2^2, \tag{4.35}$$

which penalizes solutions with a large $L^2$-norm. Certain properties of $\boldsymbol{\pi}$ can be constrained. For instance, smoothness can be enforced by applying a discrete Laplace operator **L**, such that the constraint becomes:

$$c(\boldsymbol{\pi}) = \|\mathbf{L}\boldsymbol{\pi}\|_2^2. \tag{4.36}$$

For certain problems, it might be more beneficial to use a barrier function to impose a steep penalty when the parameter exceeds a certain value. One example of such a function is the logarithmic barrier function, which for a lower bound $b_i$ is defined as:

$$c(\pi_i) = -\log(\pi_i - b_i). \tag{4.37}$$

The penalty term increases to $\infty$ as $\pi_i$ approaches $b_i$. We can then rewrite the optimization problem as:

$$\min_{\boldsymbol{\pi}} \mathscr{L}(\boldsymbol{\pi}) + \sum_{i \in \mathscr{I}} -\log(b_i - \pi_i). \tag{4.38}$$

Another common regularization term to enforce positivity is the *maximum entropy regularization* [2, 80]. The regularization term is given by:

$$c(\boldsymbol{\pi}) = \sum_i \pi_i \log(\pi_i). \tag{4.39}$$

# 5

# DESIGNING FREEFORM OPTICS WITH ALGORITHMIC DIFFERENTIABLE RAY TRACING

With the theory presented in Chapters 2, 3 and 4 we start to describe one of the principal tools to design freeform optics, *algorithmic differentiable ray tracing*. Throughout this chapter two differentiable ray tracers are used. The first one being the ray tracer developed in Koning et al. (2023) used to generate the results of Section 5.5 and MITSUBA 3 [47] which has been used to generate the results of Section 5.1. We start by defining the optimization problem and how the simulations are set up in LightTools and MITSUBA 3 to make the results as comparable as possible. We show that the method of optimizing a freeform lens for a prescribed irradiance distribution works well for simple distributions only needing a low number of control points. For more complicated irradiance distributions, the optimizers can find a lens capable of projecting the desired distribution, however, these lenses are undesirable from a manufacturing perspective and suffer from stray light around the target distribution.

## 5.1. OPTIMIZING A SINGLE LENS

Our goal is to design a freeform lens with a refractive index of $n$, which redirects the light from a source such that a prescribed irradiance distribution is formed at a predetermined target plane as depicted in Figure 5.1(a). An overview of the parameters needed to define the system is shown in Figure 5.1(b). The source can be either a collimated bundle of light or an extended rectangular source with Lambertian emission with its centroid on the optical axis and a normal perpendicular to it. The distance from the source to the origin of the first lens surface ($O_{\text{front}}$) is given by $d_0$, the distance between the origins of the front and the rear surface ($O_{\text{rear}}$) by $d_1$, and the distance between the origin of the rear surface and the target plane by $d_2$. The surfaces of the lens are described in terms of

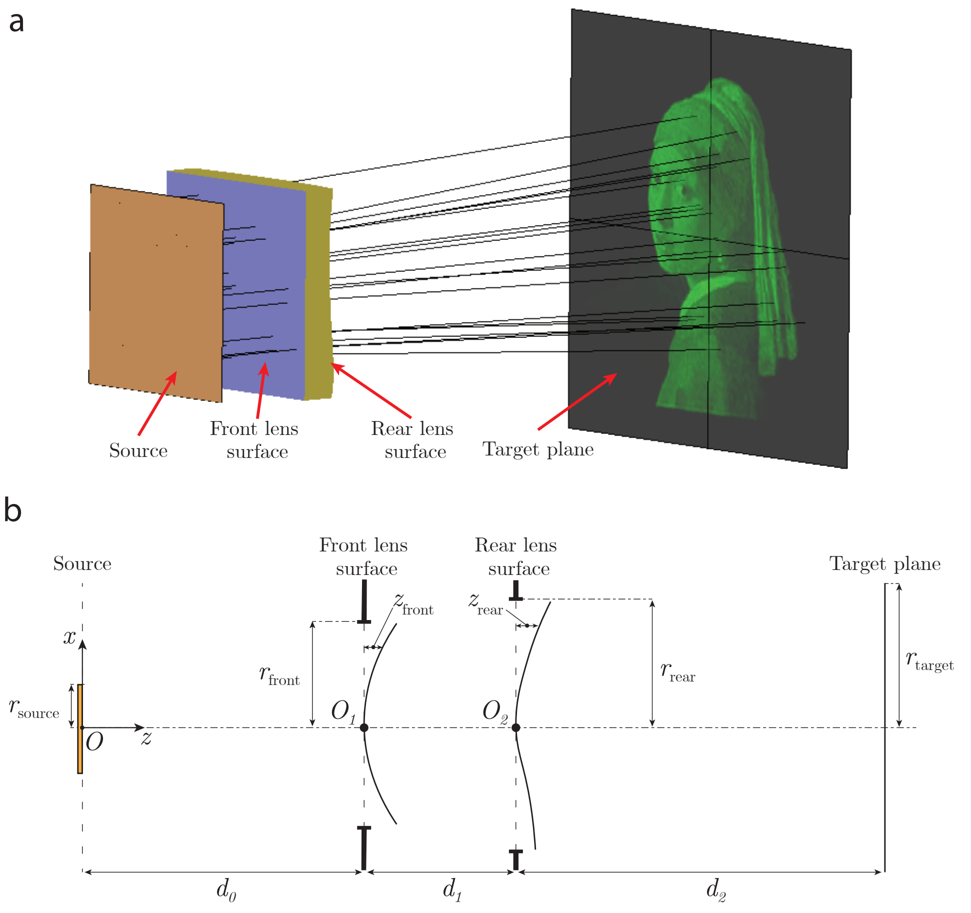


Figure 5.1: (a) 3D model of the system consisting of a source, freeform lens, and a target plane where the irradiance is measured; (b) Simplified, 2D view of the system depicting the different parameters used to define the system.

local coordinates $(x_s, y_s)$ and consist of a base conic and a sag given by a B-spline:

$$z_{\text{surface}}(x_s, y_s) = z_{\text{conic}}(R, K, x_s, y_s) + z_{\text{B-spline}}(\mathbf{C}, x_s, y_s). \tag{5.1}$$

The base conic $z_{\text{conic}}$ depends on its radius of curvature $R$ and conic constant $K$ and governed by Equation 3.2. The Cox-de Boor relations (Equation 3.5 and Equation 3.6) are used to calculate the B-spline basis functions which span the spline spaces $\mathscr{U}$ and $\mathscr{V}$.

To construct the B-spline surface, the spline control points $\mathbf{c}_{i,j,k} = \begin{bmatrix} c^x_{i,j,k} & c^y_{i,j,k} & c^z_{i,j,k} \end{bmatrix}^T$ have to be specified, with $i \in \{0, \dots, N_u - 1\}$ and $j \in \{0, \dots, N_v - 1\}$ and $k \in \{\text{front}, \text{rear}\}$.

The B-spline surface, as discussed in Chapter 3 is defined using B-spline basis functions $N_{i,p}(u)$ and $N_{j,q}(v)$ which require a specified degree $p$ and $q$ and knot vectors

$U = \{u_0, \ldots, u_m\}$ and $V = \{v_0, \ldots, v_m\}$. We also take both B-spline surfaces use degree $p$ in $u$ and $v$, therefore, we omit explicit $p, q$ from the notation from now on. Following the procedures outlined in Section 3.2.2 we can write the B-spline as a tensor product:

$$\mathbf{S}(u, v) = \mathbf{N}_u \mathbf{C} \mathbf{N}_v^T. \tag{5.2}$$

Using Section 3.2.3 we choose $c^x_{i,j,k}$ and $c^y_{i,j,k}$ so that $x(u) = (2u - 1) r_{k,x}$ and $y(v) = (2v - 1) r_{k,y}$, with inverse $u = \frac{1}{2}(x_s / r_{k,x} + 1)$ and $v = \frac{1}{2}(y_s / r_{k,y} + 1)$ (Equation 3.16), and can define the system parameters as:

$$\begin{aligned}\boldsymbol{\pi} = \{&r_{\text{source}}, r_{\text{front}}, r_{\text{rear}}, r_{\text{target}}, d_0, d_1, d_2, n, \ldots \\ &R_{\text{front}}, K_{\text{front}}, \mathbf{C}^z_{\text{front}}, p_{\text{front}}, R_{\text{rear}}, K_{\text{rear}}, \mathbf{C}^z_{\text{rear}}, p_{\text{rear}}\}.\end{aligned}$$

To limit the maximum deviation, we choose to optimize a weight, as discussed in Section 4.4.1, $w_{i,j,k} \in \mathbb{R}$, with the set of all weights denoted by $\mathscr{W} = \{w_{i,j,k}\}$. These weights are directly related to the $z$-position of the control points by:

$$c^z_{i,j,k} = \frac{2 z_{\text{freedom}}}{\pi} \arctan(w_{i,j,k}). \tag{5.3}$$

Using the arctangent, we can restrict the maximum deviation of $c^z_{i,j,k}$ to $[-z_{\text{freedom}}, z_{\text{freedom}}]$. These weights are then used to minimize the *Frobenius norm* of the difference between the irradiance measured at the target plane $\mathbf{E}^{\mathscr{W}}_{\text{target}}$ which is a variable of the weights $\mathscr{W}$ and the desired irradiance $\mathbf{E}_{\text{desired}}$:

$$\mathscr{L}(\mathscr{W}) = \sqrt{\sum_{n=1}^{N} \sum_{m=1}^{M} \left| \mathbf{E}^{\mathscr{W}}_{\text{target}}[n, m] - \mathbf{E}_{\text{desired}}[n, m] \right|^2}. \tag{5.4}$$

To minimize Equation (5.4), we utilize an algorithmically differentiable ray tracer to compute $\nabla_{\mathscr{W}} \mathscr{L}$ by backpropagating through the ray tracer.

This is then combined with a gradient-based optimization approach by making use of the PyTorch toolbox [69] and the ADAM optimizer [49]. The scheme we use to solve this problem is shown in Figure 5.2. At each iteration, we first ray trace to evaluate the irradiance distribution produced by the current lens. Then, we evaluate the loss function, and using algorithmic differentiation calculate the gradients of the control points with respect to the loss function. The amount by which the control points are changed, is calculated by the step size obtained using Adam. We then recompute the B-spline using the new control points and repeat this process until the loss falls below a threshold $\varepsilon$ or is manually stopped.

## 5.2. SYSTEM MODELING IN LIGHTTOOLS AND MITSUBA 3

We discuss how the systems are modeled in LightTools and MITSUBA. We cover setting up the light source, freeform lens, detector, and simulation settings. Additionally, we highlight the key differences between the two ray tracers.

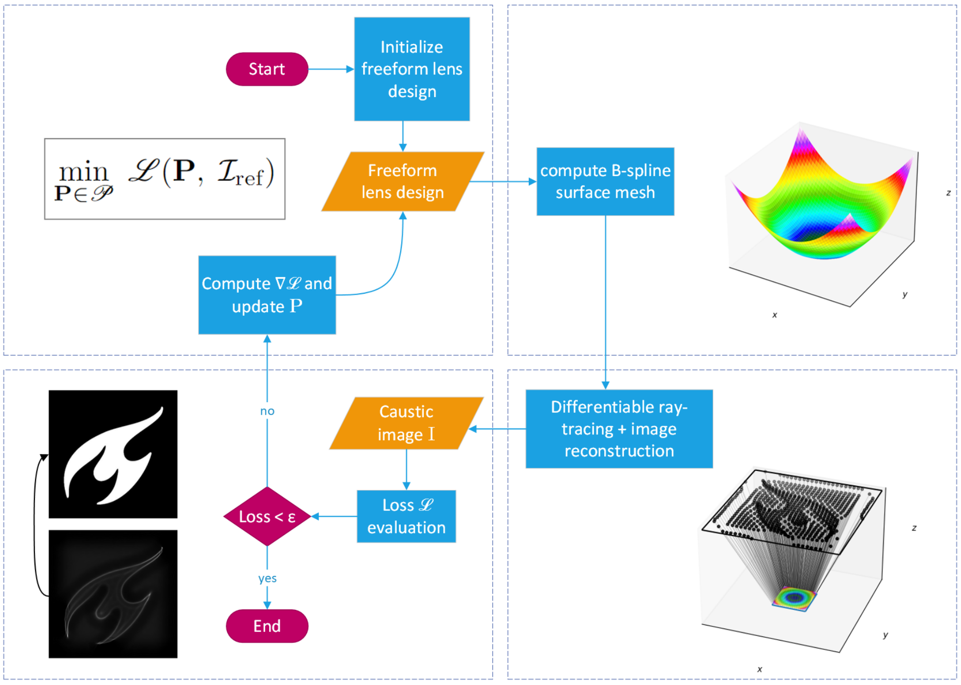

Start
Initialize freeform lens design
$\min_{\mathbf{P}\in\mathscr{P}} \mathscr{L}(\mathbf{P},\, \mathcal{I}_{\mathrm{ref}})$
Freeform lens design
compute B-spline surface mesh
Compute $\nabla\mathscr{L}$ and update P
no
Caustic image I
Differentiable ray-tracing + image reconstruction
Loss < ε
Loss $\mathscr{L}$ evaluation
yes
End
x
y
z

Figure 5.2: Overview of our learning-based freeform design pipeline.

### 5.2.1. LIGHTTOOLS

LightTools version 2022.03 SR1 is used and controlled using the Python API for all simulations.

**Source** The collimated and extended sources are initialized as rectangular sources with a width and height equal to the source radius. The emission is monochromatic at a wavelength of 550nm. In the case of the collimated source, the aim region is set to collimated, with a total flux of 1 Watt measured over the aim region. For the extended source, the aim region is set to a hemisphere facing the freeform lens, and the angular distribution is set to Lambertian with a total flux of 1 Watt, measured over the aim region.

**Lens** To model the lens, we start with a rectangular block with a height and width of the front and rear surfaces equal to the radii of the respective surface and depth of the lens thickness. The front surface is then set to a conic, and the rear surface to a freeform surface initiated with the number of vertices of the MITSUBA model and the coordinates of these points as the coordinates of the vertices. The material of the lens is a user-defined, homogeneous material with a refractive index of 1.5 at the wavelength of 550nm. The optical properties of all the lens surfaces (including the edge surfaces) are smooth optical surfaces that transmit and totally internally reflect rays with Fresnel losses.

**Detection** To measure irradiance, a dummy plane is created at the location of the target screen, with the appropriate dimensions onto which a receiver is attached with a mesh, equaling the detector's resolution and measuring the irradiance.

**Simulation** The simulation uses 10 million rays with polarization and coherence settings turned off. We chose pseudorandom ray sampling to match the ray sampling strategy from MITSUBA.

The intersection points of the rays with the target screen and the ray weights are imported into Python. To match the Gaussian reconstruction filter used by MITSUBA 3, we used the Gaussian filter Equation 2.36 with $\mu$, the average value, which determines the position of the Gaussian, and $\sigma$, the standard deviation, which determines the spread. Equation 2.36 is then used to define the reconstruction filter $W_{n,m}$, given by Equation 2.37. The standard deviation is chosen to equal the pixel separation $\sigma = 2/(N_t - 1) r_{\text{target}}$ and the filter extent $r = 4\sigma$. Finally, the pixel values are then calculated as

$$E_{n,m} = \sum_{k=1}^{N'_k} W_{n,m}(x'_k, y'_k) L(x'_k). \tag{5.5}$$

### 5.2.2. MITSUBA 3

In all simulations, MITSUBA 3.2.1 and drjit 0.4.1 are used. We largely follow the instructions of the Caustic optimization tutorial [62].

**Source** Both collimated and extended sources are initialized as rectangular objects. When a collimated light source is required, the object is given a directional area emission. In the case of an extended source, the object is given an area source emission, giving it a Lambertian emission distribution.

**Lens** We initiate the rear and front surfaces as a flat rectangle and turn them into a conic or freeform surface by displacing their vertices using a displacement texture. The height map of the THB-spline gives the amount of displacement. The front and rear surfaces are connected with a boundary, which is made by linearly interpolating the edge vertices of the two surfaces. Each lens surface is given a smooth dielectric BSDF with an interior refractive index of 1.5 and an exterior refractive index of 1. The material information is only used when a ray interacts with a lens surface, meaning that rays do not know in what material they are propagating. Therefore, it is essential to close the lens so that all the rays that enter the freeform lens need two refractions to exit the lens. Diffusion is not supported with the used version of MITSUBA. Thus, the refractive index is constant for all wavelengths.

**Detection** Unlike LightTools, MITSUBA cannot measure the irradiance directly in a plane for a grid of pixels. Therefore, the irradiance is measured by projecting the light onto a diffuse screen and imaged with a pinhole camera. The diffuse screen causes every incident ray to be evenly redistributed over the hemisphere above the target screen, thus allowing each ray to be captured by the pinhole camera. A field of view $\gamma$ is specified to ensure the entire target plane is captured. This determines the camera's position on the optical axis, a distance $d_3$ from the target screen, as illustrated in Figure 5.3. When choosing the field of view, we made sure that the pinhole camera does not fall inside or behind the freeform lens.

**Simulation** We utilize the *ptracer* integrator to simulate light propagation from source to target. The number of rays traced is determined by the pixel count of the detector and the *samples per pass* setting of the integrator, which specifies the number of rays sampled per pixel during a simulation.

## 5.3. FREEFORM OPTIMIZATION

To see how the algorithm performs we consider two target distributions: one simple target, a uniform top hat distribution, and a complex target, the painting "Girl with a Pearl Earring" depicted in Figure 5.4(a) and Figure 5.4(b) As for all examples a B-spline of degree 5 is used and the knot spans for the case of 100 control points having a width of approximately 1 mm the smallest detail on the lens will be around $5 \times 5$ mm$^2$ in size, which is much larger than the wavelength 0.55 $\mu$m, allowing us to disregard diffraction effects. We design freeform lenses for varying numbers of control points: 10, 25, 50, 75, and 100 in both $x$ and $y$. The simulation parameters used are shown in Table. 5.1 and the step size used during optimization is $10^{-3}$, unless stated otherwise,

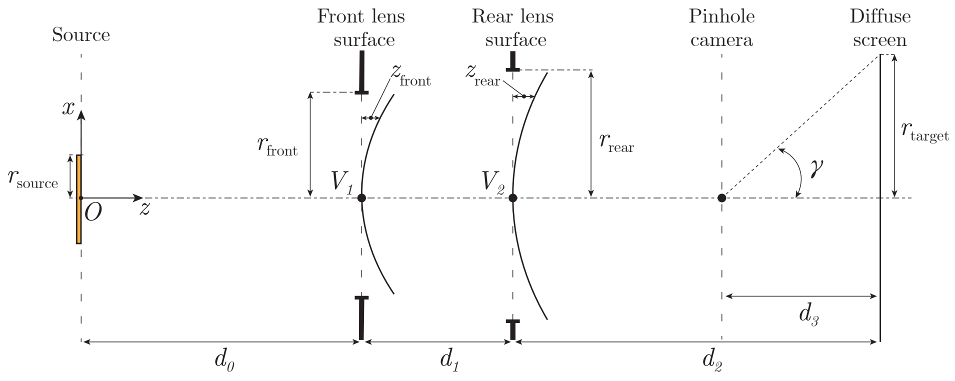


Figure 5.3: Simplified view of the optimized system consisting of a source, freeform lens, and a target plane with pinhole camera imaging the target plane.

Table 5.1: System parameters: Demonstration of directly optimizing B-spline with different amount of control points

| System Parameters | | | | Surface parameters | | | |
|---|---|---|---|---|---|---|---|
| $r_{\text{source}}$ | 47.5 mm | $d_0$ | N/A | $R_{\text{front}}$ | $\infty$ | $N_u, N_v$ | N/A |
| $r_{\text{front}}$ | 50 mm | $d_1$ | 20 mm | $K_{\text{front}}$ | 0 | $p, q$ | 5 |
| $r_{\text{rear}}$ | 50 mm | $d_2$ | 1000 mm | $R_{\text{rear}}$ | 666.7 mm | $N_g, M_g$ | 512 |
| $r_{\text{target}}$ | 75 mm | $N_t, M_t$ | 256 | $K_{\text{rear}}$ | 0 | $L$ | N/A |
| $W$ | Gaussian | $\sigma_W$ | 0.5 | $z_{\text{freedom}}$ | N/A | n | 1.5 |

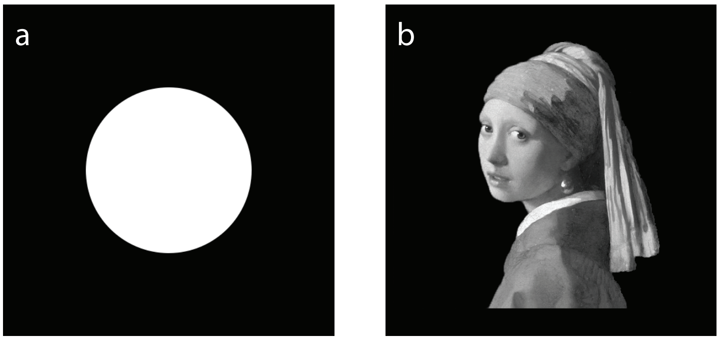


Figure 5.4: Target distributions used: (a) top hat distribution (b) painting of "Girl with a Pearl Earring."

## 5.3.1. Freeform design for uniform top-hat distribution

Optimizations for the top hat distribution were stopped after 250 iterations, and the results are shown in Figure 5.5. At first glance, the final irradiance distributions seem similar to the target distribution in both LightTools and MITSUBA, as seen in Figures 5.5(a1-e1) and Figures 5.5(a2-e2). However, the difference becomes apparent when we increase the contrast by raising each pixel to the power of 0.2, as shown in Figures 5.5(a3-e3). The light from freeform lenses optimized using 50, 75, and 100 control points falls outside the desired irradiance distribution due to the irregular surfaces obtained from optimization, as shown in Figures 5.5(c3-e3). This results in a small area on the lens contributing to a large area in the total irradiance distribution. In contrast, the surfaces obtained using 10 and 25 control points, shown in Figures 5.5(a3-b3), are smooth, and all the light falls within the irradiance distribution.

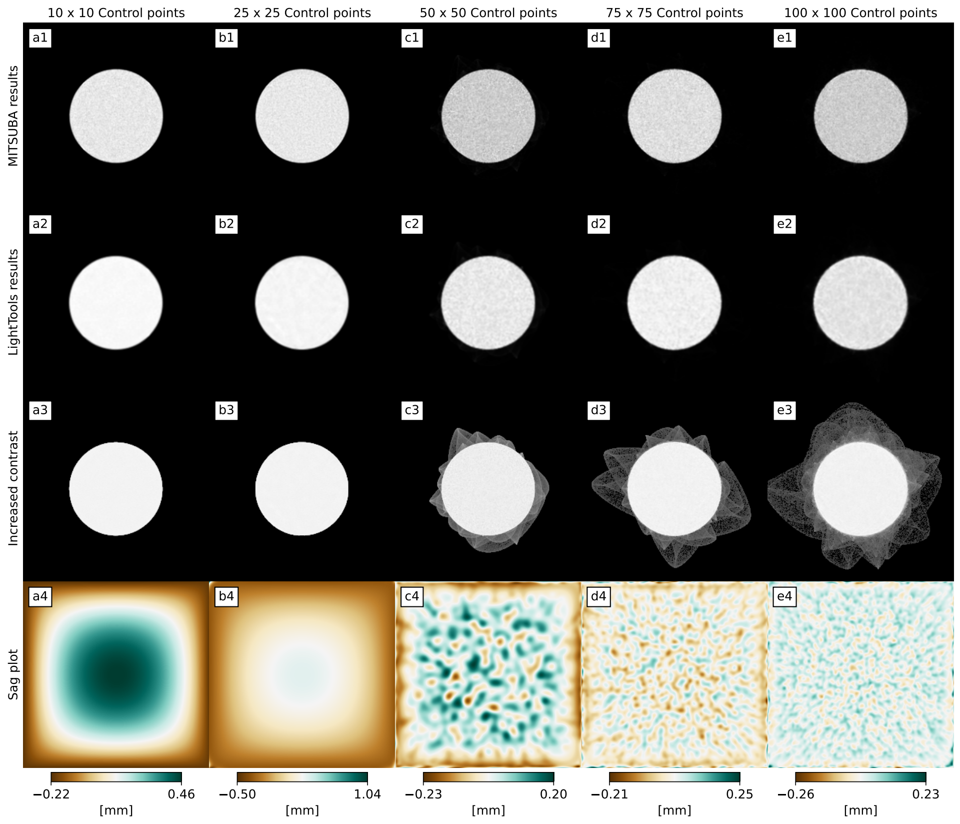


Figure 5.5: (a1 - e1) final results using MITSUBA; (a2 - e2) final results using LightTools; (a3 - e3) increased contrast by raising each pixel of the LightTools results to a power of 0.2; (a4 - e4) the final freeform surface obtained.

### 5.3.2. FREEFORM DESIGN FOR GIRL WITH A PEARL EARRING

We now consider the more complex target distribution, the picture of the "Girl with a Pearl Earring." The optimizations were stopped after 1000 iterations. To accurately reproduce the picture's fine details, we expect significantly more control points to be needed than the top hat to reproduce it. The results validate this assumption. Specifically, Figures 5.6(a1-b1) and Figures 5.6(a2-b2) show that $10^2$ and $25^2$ control points are not enough to accurately reproduce the target distribution. From $50^2$ or more control points, finer details can be included in the distribution, and only slight improvements are achieved by using $75^2$ and $100^2$ control points as seen in Figures 5.6(c1-e1) and Figures 5.6(c2-e2). By increasing the contrast of the irradiance and examining the light outside the target distribution (as shown in Figures 5.6(a3-e3)), we observe that a higher amount of control points leads to more light appearing outside the target. This is caused by increasingly irregular lenses, as seen in Figures 5.6(a4-e4).

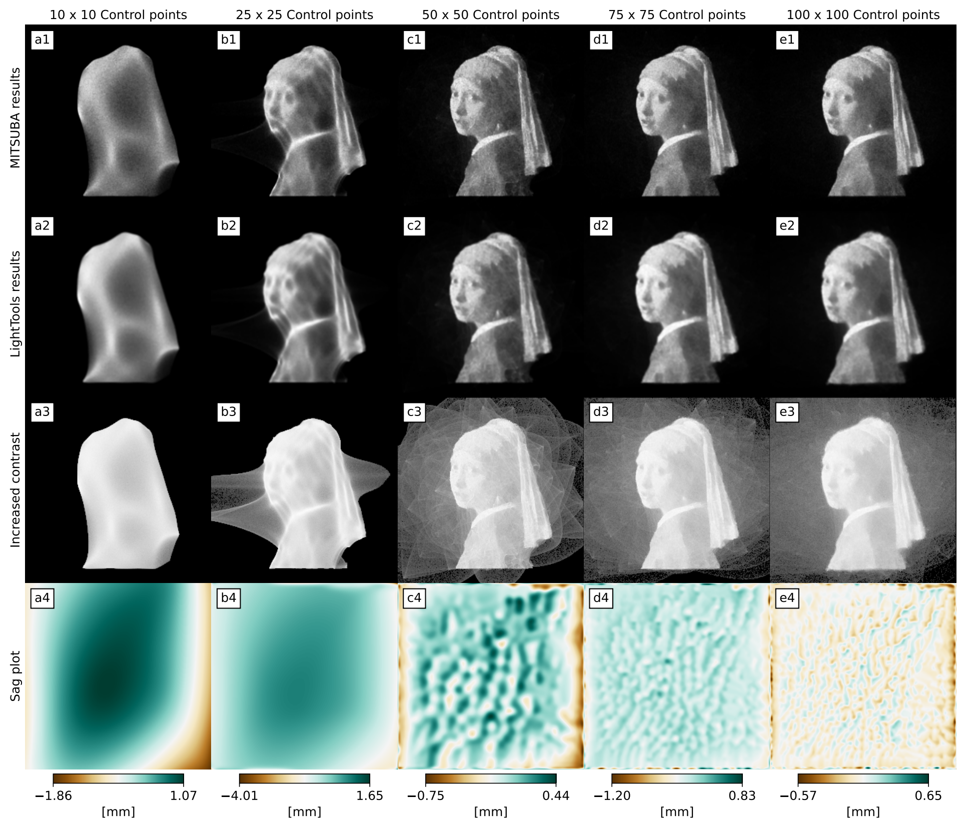


Figure 5.6: (a1 - e1) final results using MITSUBA; (a2 - e2) final results using LightTools; (a3 - e3) increased contrast by raising each pixel of the LightTools results to a power of 0.2; (a4 - e4) the final freeform surface obtained.

The reason irregular lenses are obtained is likely attributed to the rather large step size.

This causes the lens to locally want to contribute to the whole irradiance distribution which results in these diffuser like lenses. To see whether this is the case, we investigate the effect of decreasing step size on the final lens geometry and the amount of stray light.

## 5.4. Decreasing step size

To make lenses smoother and reduce light outside the target distribution, we test whether decreasing the step size of the optimizer has the desired effect. Figure 5.7 shows the results of using a step size of $10^{-4}$ for the top hat and the "Girl with a Pearl Earring" target distributions. The lenses obtained using $50^2$ and $75^2$ control points are much smoother and have significantly less light outside the target distribution as seen in Figures 5.7(a3,b3, d3, and e3) and Figures 5.7(a4,b4, d4, and e4). However, for $100^2$, rough patches persist on the lens surface, indicating the need for further step size reduction.

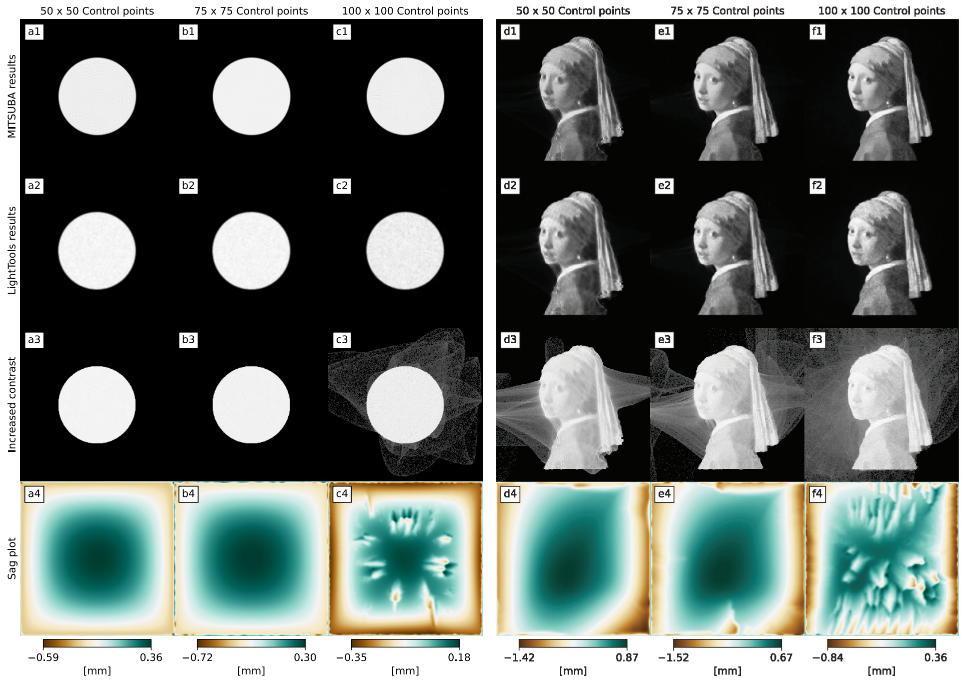


Figure 5.7: Results obtained using a step size of $10^{-4}$: (a1 - c1) final results using MITSUBA for the top hat distribution; (d1 - f1) final results using MITSUBA for Girl with a Pearl Earring; (a2 - c2) final results using LightTools for the top hat distribution; (d2 - f2) final results using LightTools for Girl with a Pearl Earring; (a3 - f3) increased contrast by raising each pixel of the LightTools results to a power of 0.2; (a4 - f4) the final obtained freeform surfaces.

While decreasing the step size is able to create better behaving freeform lenses, it leads to much longer optimization times, as can be seen in Figure 5.8 and Figure 5.9. The graphs show the progression of the optimization loss for two irradiance distributions for step sizes of $10^{-3}$ and $10^{-4}$. While this might not be an issue in certain situation, as we

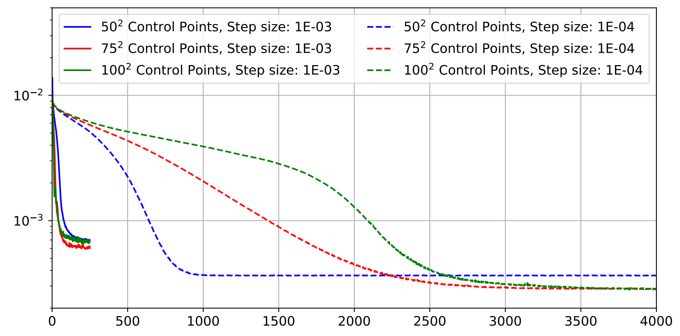


Figure 5.8: Loss progression of optimizations of the top hat distribution.

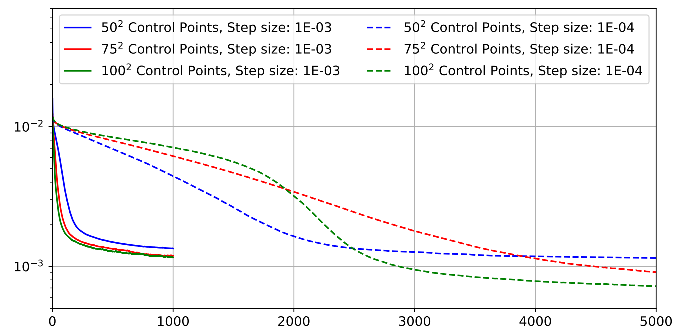


Figure 5.9: Loss progression of optimizations of the Girl with a Pearl Earring.

see the decrease in step size does not guarantee. One way to decrease step size, without having much longer optimization times is through the use of optimizing a multi-layer perceptron instead of the control points of the lens directly.

## 5.5. MULTI-LAYER PERCEPTRON AS AN OPTIMIZATION ACCELERATOR

The idea of employing multi-layer perceptrons (MLPs) originated from an exploratory attempt to train a neural network capable of directly generating freeform lenses with the input being a specific target irradiance distributions. The initial objective of fully replacing direct optimization was not achieved. However, during experimentation we discovered that the inclusion of a neural network could accelerate the optimization process significantly. Here, we investigate in more detail how different MLP architectures influence the convergence behavior.

We parameterize the control-point heights with several MLP architectures that take a constant scalar input of 1. In this configuration, instead of directly optimizing the control point $z$-coordinates, we optimize the trainable parameters of the neural network, which subsequently map onto the $z$-values of the control points. The hypothesis behind choosing different neural network architectures is that this indirect parameterization yields smoother lenses depending on the network connections as changing a neuron will affect multiple control points in which information from different control points can be influenced by a single parameter and potentially faster convergence than direct optimization. The networks we considered are standard feed-forward MLPs, consisting of multiple layers of neurons, illustrated schematically in Figure 5.10.

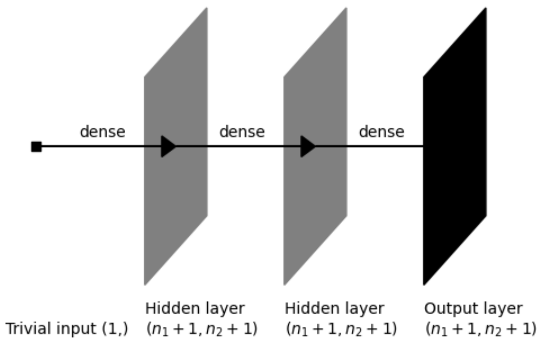

dense
dense
dense
Trivial input (1,)
Hidden layer $(n_1+1, n_2+1)$
Hidden layer $(n_1+1, n_2+1)$
Output layer $(n_1+1, n_2+1)$

Figure 5.10: The dense multi-layer perceptron architecture based on the size of the control net $(n_1+1)\times(n_2+1)$.

Specifically, we compare the following architectures:

1. *No network* (NN): direct optimization of the control points, serving as baseline.
2. *Random Sparse* (RS): network with randomly initialized weights and biases

3. *Uniform Sparse* (US): with sparsity informed by the support of overlapping B-spline basis functions. This approach enables controlled interactions between control points sharing influence over common knot spans. Thus, each neuron layer has connectivity analogous to a convolutional layer with kernel size $(2p+1, 2q+1)$. However, in contrast to standard convolution, each connection possesses a unique weight, and each kernel has a distinct bias.

4. *Uniform Dense* (UD): fully connected MLP architecture, with three layers sized according to the control net dimensions with weights initialized uniformly from a small interval $U([-10^{-4}, 10^{-4}])$. This architecture rapidly increases the number of trainable parameters ($n^4$ parameters for two consecutive layers, given a square control net with side length $n$).

The hyperbolic tangent is chosen as the activation function for all neurons.

We tested these different architectures by optimizing a freeform lens that projects a circular top-hat distribution (Figure 5.11) from a collimated source. Figure 5.12 shows the

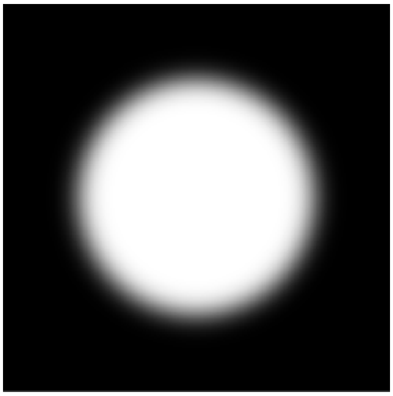


Figure 5.11: The circular top hat target illumination.

optimization loss progression over 1000 iterations for different step sizes ($10^{-3}$, $10^{-4}$, and $10^{-5}$). The resulting lens shapes and irradiance distributions obtained during optimization at a step size of $10^{-4}$ are shown in Figures 5.13, 5.14, 5.15, 5.16, and 5.17.

The first notable difference between randomly and uniformly initialized sparse neural networks is that the uniformly initialized converged loss values are much lower than the randomly initialized, regardless of the chosen step size. There are two possible explanations for why this is the case. First, due to the random initialization of the network the optimizer tends to converge to a different local minimum than the uniform initialized networks, as shown in Figure 5.13. Second, as the parameter values are randomly initiated, the influence of certain nodes on certain control points might be unbalanced, resulting in some control points having a much higher contribution throughout the optimization than other control points.

When comparing the different step sizes of the optimizations done using no network, US, and UD, we see that when using a step size of $10^{-3}$, all three cases converge to a loss of $5 \times 10^{-7}$ in roughly 600 iterations as shown in Figure 5.12. At a step size of $10^{-4}$, the uniform dense network initially decreases slower but catches up to the other networks using a step size of $10^{-3}$ at around 400 iterations, eventually converging to a similar loss value of $5 \times 10^{-7}$. While the uniform sparse network takes longer to reach the same loss value, it still manages to do so after 1000 iterations. When decreasing the step size to $10^{-5}$, none of the cases fully converge. However, we observe the same behavior of the uniform dense network, converging faster than the uniform sparse network, which again converges faster than the no networks case.

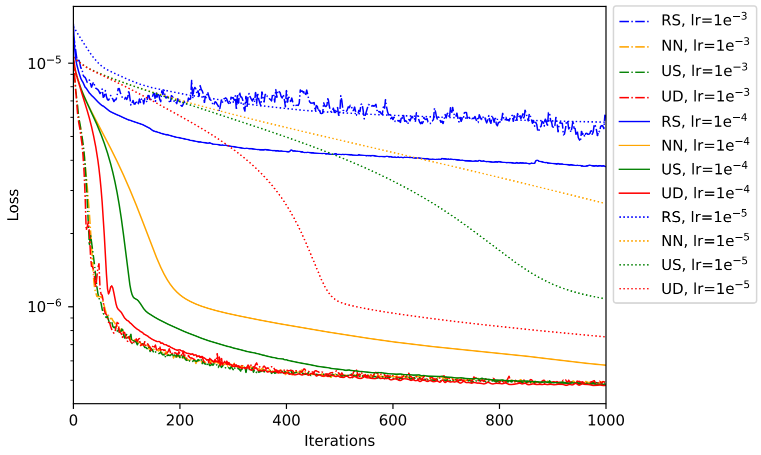


Figure 5.12: Loss progress over the iterations for the pipeline-setups: random sparsely connected (RS), no network (NN), uniform sparsely connected (US), and uniform densely connected (UD) using step sizes $10^{-3}$, $10^{-4}$, and $10^{-5}$ for forming a top hat distribution from a collimated beam.

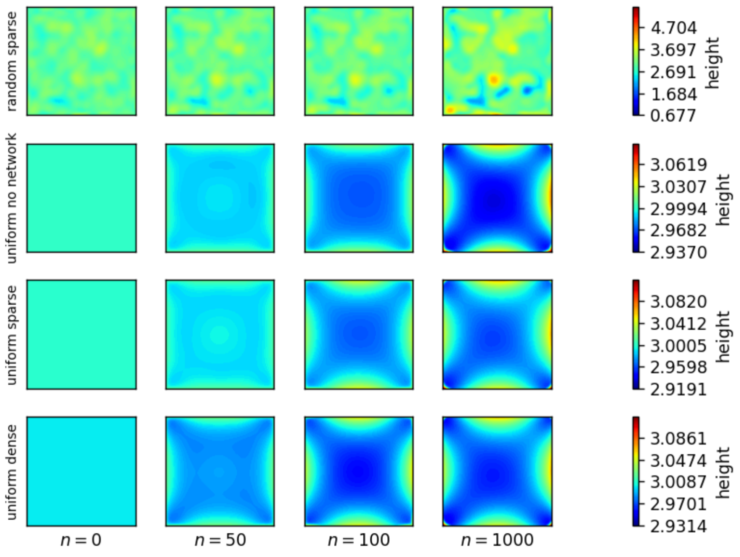


Figure 5.13: The lens height field after initialization ($n = 0$), and $n = 50, 100$ and 1000 iterations respectively, using a step size of $10^{-4}$ and different network architectures and network parameter initializations.

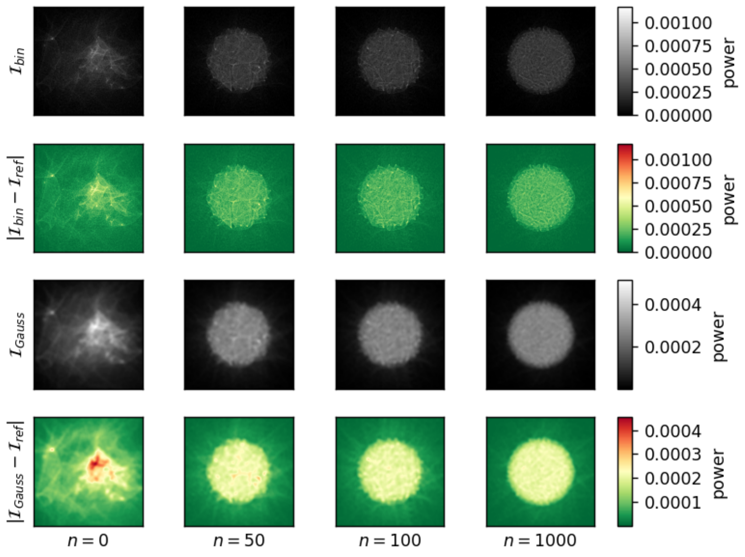


Figure 5.14: Irradiance distributions and pixel-wise errors in the optimization progress of a lens with a random sparse network and a step size of $10^{-4}$ towards a circular top hat illumination.

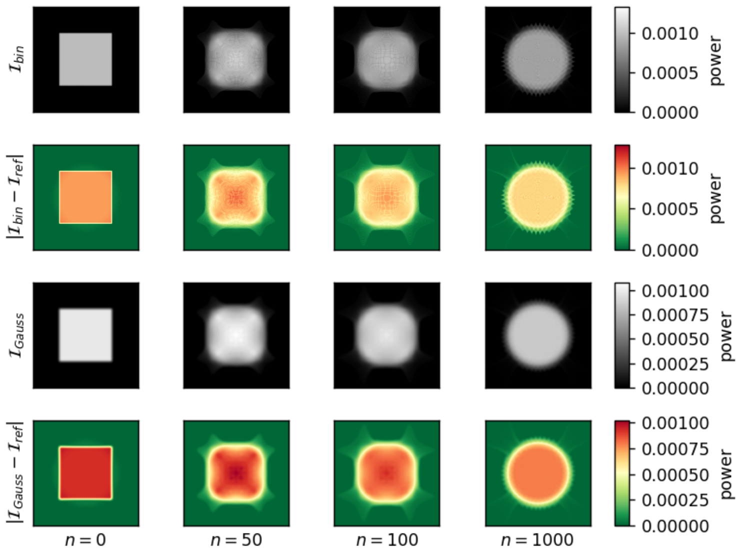


Figure 5.15: Irradiance distributions and pixel-wise errors in the optimization progress of a flat lens without a network and a step size of $10^{-4}$ towards a circular top hat illumination.

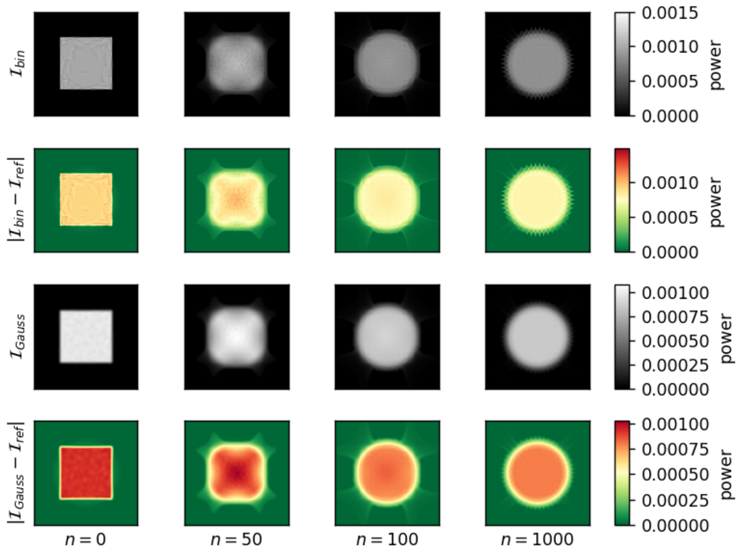


Figure 5.16: Irradiance distributions and pixel-wise errors in the optimization progress of a flat lens with a sparse network and a step size of $10^{-4}$ towards a circular top hat illumination.

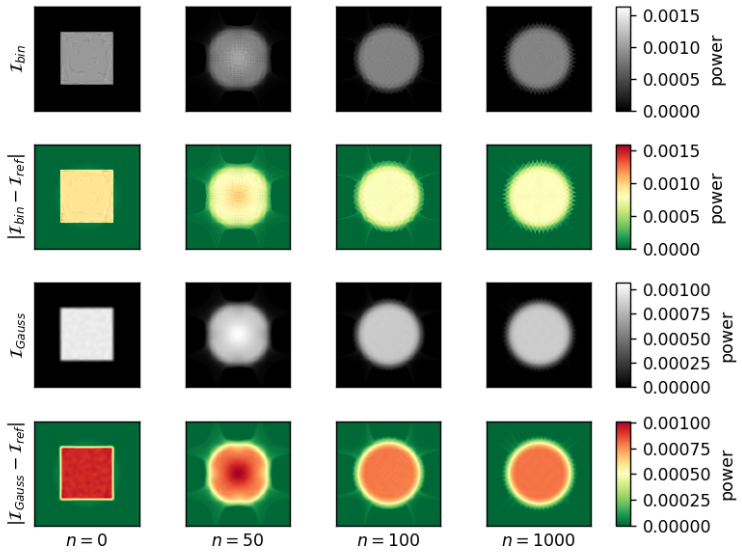


Figure 5.17: Irradiance distributions and pixel-wise errors in the optimization progress of a flat lens with a dense network and a step size of $10^{-4}$ towards a circular top hat illumination.

## 5.6. CONCLUSION

We demonstrated that non-sequential differentiable ray tracing is a viable tool for designing freeform lenses.

For simple irradiance distributions, freeform lenses made using a low number of degrees of freedom converge to smooth lenses. However, when the number of degrees of freedom increases, which is also required to project complex irradiance distributions, this is not guaranteed. With this approach the obtained lenses become irregular, almost diffuser like objects.

One simple way to resolve this is to decrease the step size used during optimization. This allows smaller changes during optimizations, giving the control points more time to adjust to their neighbors contributions.

Using a neural network to remap the optimization space provides an interesting way to increase the convergence speed of the optimization. However, further investigation is required to see whether this generally holds and what the effect is on other network architectures. Furthermore, using transfer learning knowledge from past optimizations can be transferred to new optimization scenarios, potentially reducing the time required for optimization.

Using only a few control points yields smooth, well-behaved lenses, but it cannot reproduce the fine detail of a complex target. It is able to produce only a coarse, yet recognizable, approximation. A progressive-refinement strategy can overcome this limitation: by gradually adding control points to increase the degrees of freedom of the B-spline. With the introduction of new degrees of freedom, more details can then be added to the obtained irradiance distribution. Such refinement can be realized with both knot insertion and the use of THB-spline refinement.

# 6

# THB-refinement and Optimization Procedure

In Chapter 5, we observed that designing freeform lenses with B-splines for a complex irradiance distribution can easily lead to irregular lens surfaces unless preventative measures are taken such as decreasing the step size. We also observed that for a smaller number of degrees of freedom the fine details of the irradiance distribution are missing, although the general shape is typically recognizable. Building on these insights, we propose a gradual refinement strategy. Initially, we optimize the lens using a small number of control points to establish its coarse shape. Subsequently, additional degrees of freedom are introduced to capture the finer details. To accomplish this, we use two methods to introduce new control points: knot insertion and Truncated Hierarchical B-splines (THB-splines). Both approaches allow for local refinement of the surface. However, as discussed in Section 3.3.2, there is ambiguity when inserting a knot using B-splines, and therefore we adopt uniform refinement when using ordinary B-splines. In contrast, THB-splines allow for local refinement. This allows us to develop a more rigorous algorithm to decide where new control points should be added. To achieve this we make use of algorithmic differentiation, which can give us both the gradient of the control points and the surface vertices. The vertex gradients quantify local sensitivity and let us identify regions that need refinement. This technique is introduced in Section 6.1 and its effectiveness is demonstrated by fitting a THB to a height map in Section 6.2. Finally, Section 6.3.1 compares THB-refinement with uniform knot insertion for the Girl with a Pearl Earring target distribution with a zero-étendue source. Section 6.3.2 extends the THB-spline approach to a finite étendue source which projects a curved target distribution.

## 6.1. INITIATING THE THB ALGORITHM

We start the optimization by initializing the surface with a low number of control points and perform an optimization for a fixed number of iterations. Once finished, we identify areas where the surface requires refinement by examining how changes in the vertex positions affect the irradiance distribution. The surface has $N_g$ vertices in the $x$-direction and $M_g$ vertices in the $y$-direction. These vertices are located on a grid $G$ within the domain $\mathbf{\Omega} = [0,1]^2$ with the $x$ and $y$ positions of the points:

$$G_{i,j} = \left(\frac{i}{N_g - 1}, \frac{j}{M_g - 1}\right) \quad \text{with} \quad i, j \in \mathbb{N},\ 0 \le i \le N_g - 1,\ 0 \le j \le M_g - 1. \tag{6.1}$$

Each vertex lies inside exactly one knot span product. These are domains between adjacent knots represented by $C^{\ell}_{r,s} = [u^{\ell}_r, u^{\ell}_{r+1}) \times [v^{\ell}_s, v^{\ell}_{s+1})$ which subdivide the surface. Each knot span product is identified by its index $rs$ and its refinement level $\ell$. To determine which knot spans need refinement, we calculate the gradients of the $z$-positions of the vertices, denoted as $Z_{i,j}$, within a knot span product and sum the absolute values within the knot span $C^{\ell}_{r,s}$, given by

$$\mathcal{Z}^{\ell}_{r,s} = \sum_{i,j \in C^{\ell}_{r,s}} \left| \frac{\partial \mathcal{L}(Z_{i,j})}{\partial Z_{i,j}} \right|. \tag{6.2}$$

6

If the absolute sum of the vertex gradients within a knot span exceeds the average absolute gradient by a threshold value $\alpha$:

$$\mathcal{Z}^{\ell}_{r,s} > \frac{\alpha}{2^{\ell} m_u m_v} \sum_{r',s'} \mathcal{Z}^{\ell}_{r',s'}, \tag{6.3}$$

the cell is refined, and a new optimization batch is started. The sum is weighted by the number of knots $m_u$ and $m_v$ in the $u$ and $v$ directions at level $\ell = 0$. We iterate optimization and adaptive refinement until the maximum hierarchy level $L$ is reached.

## 6.2. FITTING A THB-SPLINE TO A HEIGHT MAP VIA OPTIMIZATION

To demonstrate the effectiveness of the automated surface refinement strategy, we approximate a height map $\mathbf{Z}_{\text{target}} \in \mathbb{R}^{N \times M}$ depicted in Figure 6.1(a1). using a THB-spline $\mathbf{Z}^{\mathbf{C}}_{\text{THB}} \in \mathbb{R}^{N \times M}$ with control point matrix $\mathbf{C}$. The THB-spline is initialized as a B-spline of degree 3 with $10 \times 10$ control points, and we set the maximum allowable refinement level of the THB-spline to $L = 4$. We minimize the Frobenius norm of the difference between spline and target surface:

$$\min_{\mathbf{C}} \sqrt{\sum_{n=1}^{N} \sum_{m=1}^{M} \left|\mathbf{Z}^{\mathbf{C}}_{\text{THB}}[n,m] - \mathbf{Z}_{\text{target}}[n,m]\right|^2}. \tag{6.4}$$

After 150 optimization iterations at one refinement level, the mesh was refined. To reach the highest refinement level of 4, a total of 750 iterations are performed. In Figure 6.1

the optimization results are shown. Each column (a-e) corresponds to the successive refinement levels $\ell = 0\ldots 4$. The difference of the THB-spline surface before and after optimization is shown in Figures 6.1(a2-e2) with the THB-surface shown in Figures 6.1(a3-e3). Each iteration, higher resolution details are added, as seen in the hierarchical meshes shown in Figure 6.1(a4-e4). The vertex gradient maps Figure 6.1(a5–e5) confirm that high-gradient regions trigger local refinement.

## 6.3. RESULTS

We demonstrate the working of the proposed optimization method with two design examples. The first design is a freeform lens, illuminated by a collimated beam of light that projects the image of the "Girl with a Pearl Earring." The second design projects a smooth, curved irradiance distribution with an extended source close to the front surface. All simulations were done on a desktop computer with an Intel Xeon CPU E5-1620 v3 and NVIDIA RTX 2080 Ti GPU.

A Gaussian measurement function is used to ensure the stability of the gradient calculations of the surface control points. This is because the Gaussian measurement functions of neighboring pixels slightly overlap, preventing rays from suddenly jumping from one pixel to another. In addition, the standard deviation of the Gaussian was tuned to smooth out the vertex gradients. This was necessary as small standard deviations produced noisy gradients, making the refinement procedure unstable.

MITSUBA 3 does not provide a direct method of measuring irradiance values for a grid of pixels. Therefore, the light redirected by the freeform lens is projected onto the target screen and imaged by a pinhole camera, giving an unaberrated image of the irradiance distribution used in the optimization. More details are found in the Section 4 of the supplementary materials. It is also possible to design freeform lenses with B-splines with a static number of degrees of freedom. In Section 1 of the supplementary materials we compare results for a simple top-hat distribution and the image of the Girl with a Pearl Earring with B-splines with various static numbers of degrees of freedom. We learn from these examples that large degrees of freedom with large step sizes lead to highly irregular lenses. This can be mitigated to some extent by choosing a smaller step size, leading to a longer optimization time. Thus giving a trade-off between the number of degrees of freedom and the step size used. In addition, for the zero-etendue example, we compare THB-splines with regular B-spline refinement, where all knot spans are refined after a fixed number of iterations. All results are compared to LightTools [84], in which the system is recreated and simulated. A detailed description of how the systems are modeled in both MITSUBA and LightTools is presented in Section 4 of the supplementary materials, and the lens before and after optimization and the corresponding LightTools models are shown in Section 5 of the supplementary material. Finally, we look at the size of the knot spans as their size indicates whether or not diffraction due to small details on the lens could be problematic [75].

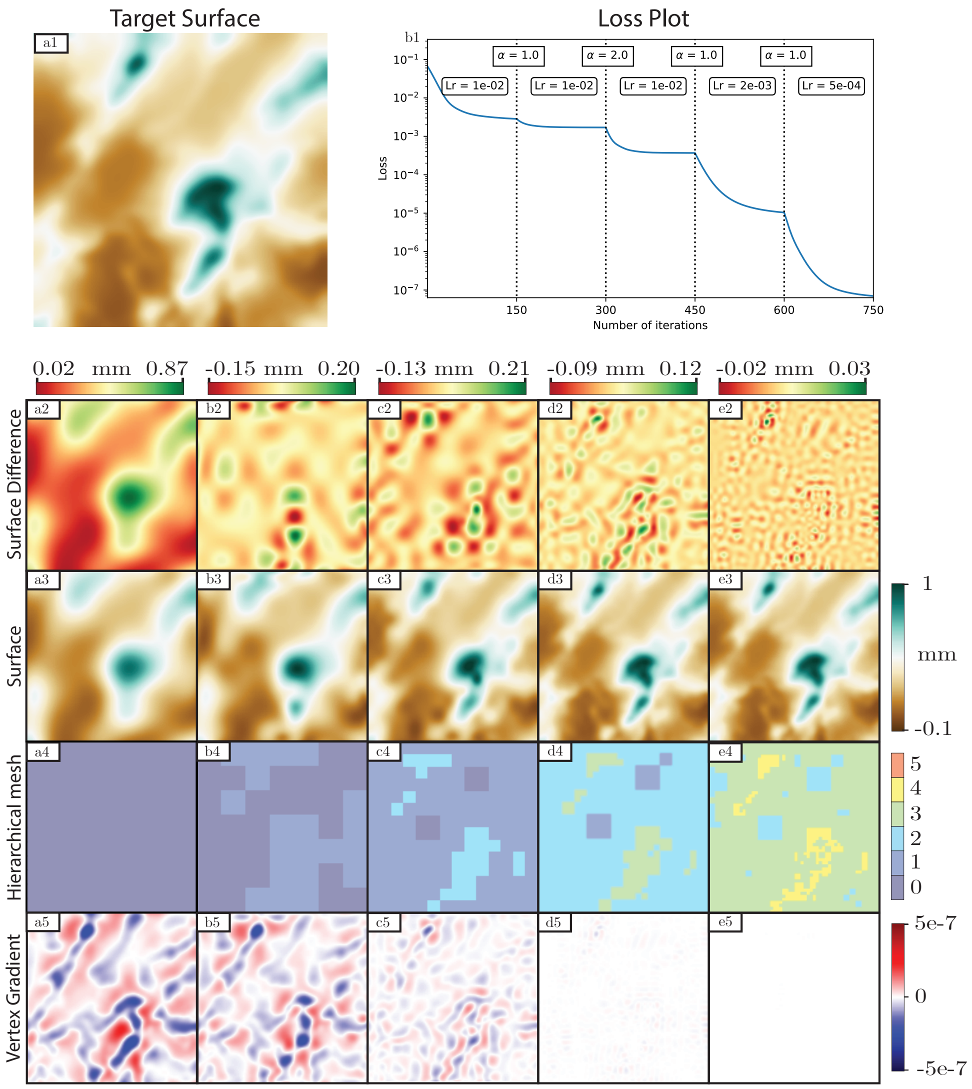


Figure 6.1: (a1) Desired surface; (b1) plot of loss through optimization with refinement parameters and step sizes; (a2-e2) difference in surface between subsequent optimization steps; (a3-e3) surface at the end of each optimization step; (a4-e4) hierarchical mesh of the THB-spline, showing which knot span products are refined; (a5-e5) gradients of the surfaces vertices;

### 6.3.1. THB-SPLINE FREEFORM LENS OPTIMIZATION ZERO-ÉTENDUE

We optimize the rear surface of the freeform lens, which is illuminated by a uniform collimated beam of light, with all rays having the same radiance, to generate the Girl with a Pearl Earring (Figure 6.2(a1)) at a distance of 300 mm of the lens. A radius of 100 mm and a conic constant of $-1$ were chosen for the base conic surface such that the initial irradiance distribution was slightly smaller than the desired target distribution. The spline surface was initialized as a B-spline of degree 3, and a control net of $10 \times 10$ and $z_{\text{freedom}}$ was chosen to be a quarter of the thickness of the lens. The values of all other system parameters can be found in Table. 6.1.

Table 6.1: **System parameters for a collimated beam of light with Girl with a Pearl Earring as target distribution**

| System Parameters | | | | Surface parameters | | | |
|---|---|---|---|---|---|---|---|
| $r_{\text{source}}$ | 47.5 mm | $d_0$ | N/A mm | $R_{\text{front}}$ | $\infty$ mm | $N_u, N_v$ | 14 |
| $r_{\text{front}}$ | 50 mm | $d_1$ | 20 mm | $K_{\text{front}}$ | 0 | $p, q$ | 3 |
| $r_{\text{rear}}$ | 50 mm | $d_2$ | 300 mm | $R_{\text{rear}}$ | 100 mm | $N_g, M_g$ | 449 |
| $r_{\text{target}}$ | 100 mm | $N_t, M_t$ | 256 | $K_{\text{rear}}$ | -1 | $L$ | 5 |
| $W$ | Gaussian | $\sigma_W$ | 1 | $z_{\text{freedom}}$ | 5 mm | n | 1.5 |

Figure 6.2(b1) shows the chosen step sizes and refinement parameters used during optimization batches. The step sizes were selected to minimize the number of iterations required for the optimization to converge and to be small enough to avoid instability. Furthermore, the refinement parameters were chosen first to refine the whole lens surface and then prioritize areas of high vertex gradients in later stages to correct the fine details in the desired irradiance distribution. It took 7 minutes and 31 seconds to optimize the lens. When analyzing the changes in the irradiance distribution after each optimization batch, as expected, the rough contour is shaped. With increasing refinement level of the THB-spline, finer details are filled in the irradiance. However, the first two refinement steps do not significantly reduce the total loss, as seen in the minor changes in the irradiance distributions in (Figures 6.2(b6 and c6)) compared to the initial optimization batch. Examining the vertex gradients, in which we can recognize a distorted version of the target distribution, reveals that vertex gradients remain mostly unchanged until the third refinement step Figures 6.2(a5-c5). This is because the high vertex gradients vary with a small area, requiring a high refinement level to eliminate them. As can be seen in the hierarchical meshes shown in Figures 6.2(a4-f4), it is precisely around these areas that the THB-spline has the highest refinement level.

One area where the optimizer has an issue is the dark area between the chin and the dress. Initially, it attempts to move the light off the target plane to prevent it from contributing to the irradiance distribution. However, upon close examination, we see that the size of this area is reduced, as in the distorted image which can be seen in the vertex gradients Figures 6.2(a4-e4), the chin and the dress move closer toward each other, causing the size of the area which discards light to decrease.

Figure 6.3 shows the results for the B-spline. No significant differences can be noticed either in the surface or irradiance distributions. Comparing the loss values shown in

Table 6.2: Numbers of degrees of freedom at the end of each optimization batch for both B- and THB-splines

| | 250 | 500 | 750 | 1000 | 1250 | 1500 |
|---|---|---|---|---|---|---|
| THB-spline | 100 | 161 | 387 | 1297 | 3818 | 11955 |
| B-spline | 100 | 289 | 961 | 3481 | 13225 | 51529 |

Figure 6.4, we see they both end up at comparable loss values. However, we observe that the THB-spline results were obtained using fewer degrees of freedom than the B-spline Table. 6.2.

We can see that the results obtained in MITSUBA 3 (Figure 6.2(f6)) and the verification in LightTools (Figure 6.2(c1)) are visually consistent with each other. The way the LightTools loss decreases compared with the MITSUBA loss is shown in Figure 6.2(b1) and Figure 6.3(b1). For the first three batches, the LightTools loss matches the MITSUBA loss. However, for the final three batches, the difference in the loss increases. As this difference increases during the final batches, the mismatch is likely caused by errors introduced by importing the THB surface into LightTools.

Of the total power emitted by the source (1 Watt), 0.92 Watts end up in the final distribution. At the highest refinement, a knot span has a size of $156 \times 156\ \mu\mathrm{m}^2$. As we work with degree-3 splines, the smallest change in the lens surface has a support of $468 \times 468\ \mu\mathrm{m}^2$, which is roughly one thousand times larger than the wavelength size. This allows us to disregard diffraction effects.

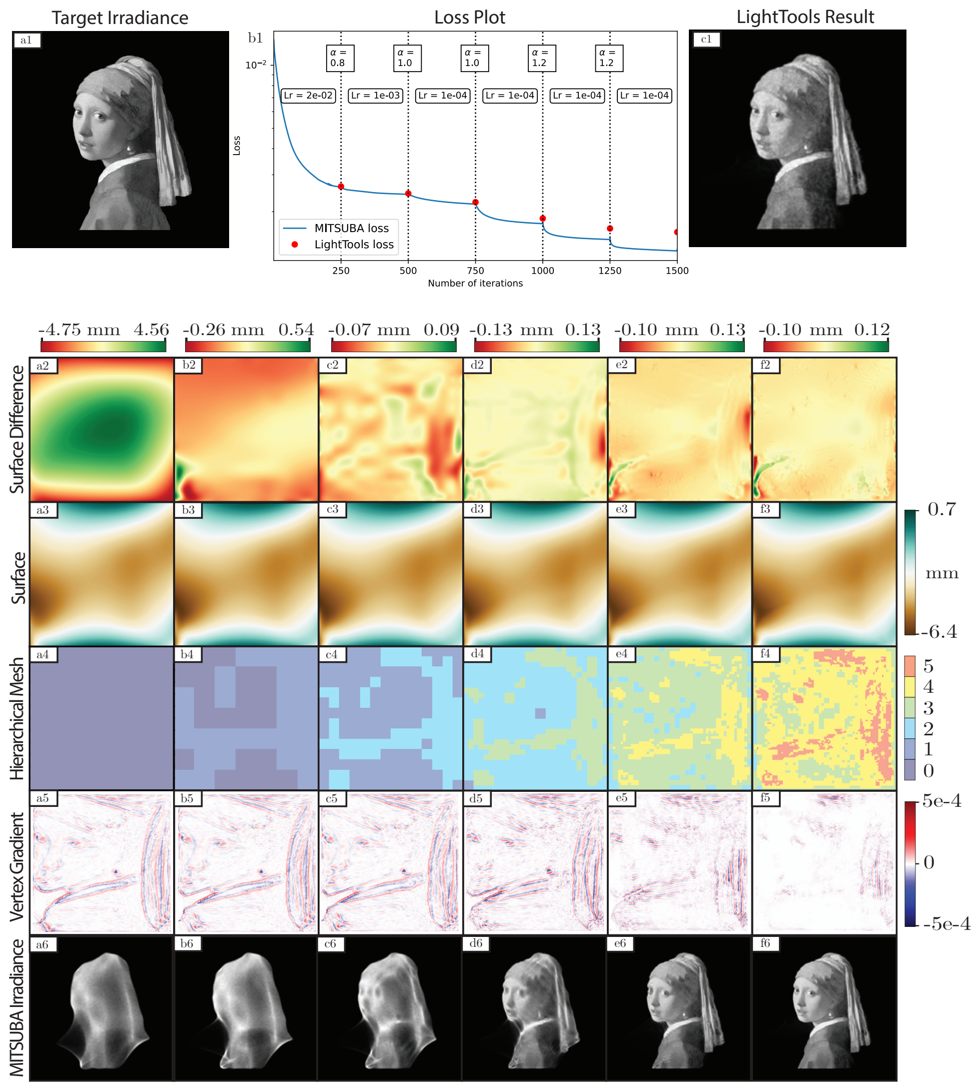


Figure 6.2: Results using THB-refinement strategy (a1) Desired irradiance distribution; (b1) Plot of loss through optimization with refinement parameters and step sizes and LightTools loss at the end of each batch; (c1) Irradiance obtained from LightTools; (a2-f2) The difference in the surface between subsequent optimization steps; (a3-f3) Surface at the end of each optimization step; (a4-f4) Hierarchical mesh of the THB-spline, showing which knot span products are refined; (a5-f5) Gradients of the vertices of the surface; (a6-f6) Obtained irradiance after every optimization batch.

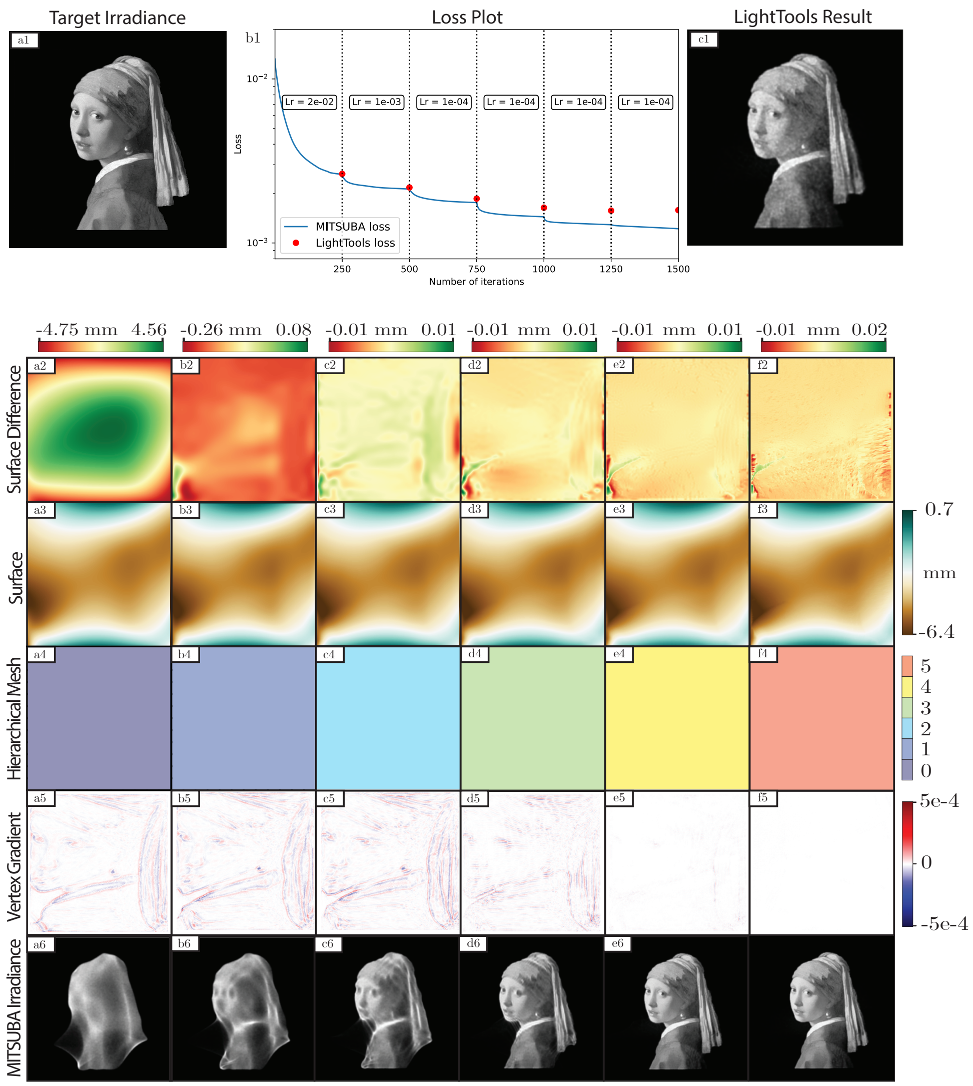


Figure 6.3: Results by using B-spline refinement strategy (a1) Desired irradiance distribution; (b1) Plot of loss through optimization with refinement parameters and step sizes and LightTools loss at the end of each batch; (c1) Irradiance obtained from LightTools; (a2-f2) The difference in the surface between subsequent optimization steps; (a3-f3) Surface at the end of each optimization step; (a4-f4) Hierarchical mesh of the THB-spline, showing which knot span products are refined; (a5-f5) Gradients of the vertices of the surface; (a6-f6) Obtained irradiance after every optimization batch.

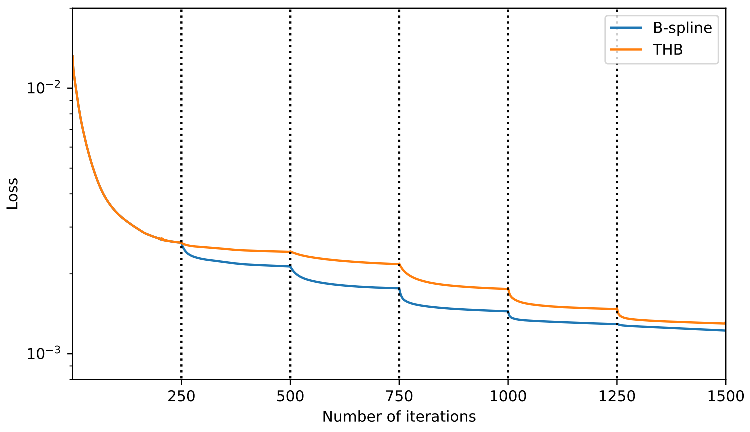


Figure 6.4: Loss comparison between THB and B-spline

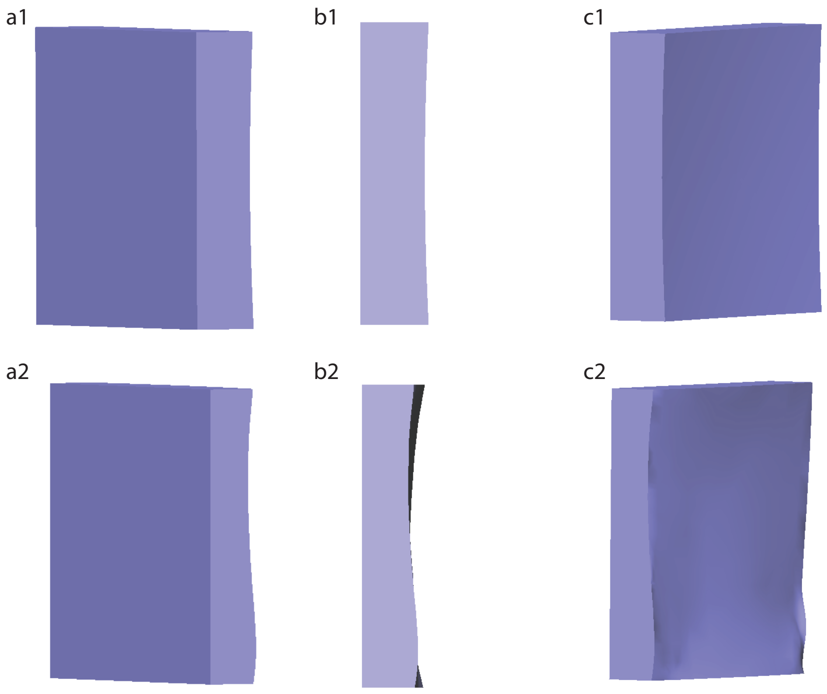


Figure 6.5: Images of the lens projecting the "Girl with a Pearl Earring" from left, center and right before and after optimization, (a1-c1) lens before optimization; (a2-c2) lens after optimization.

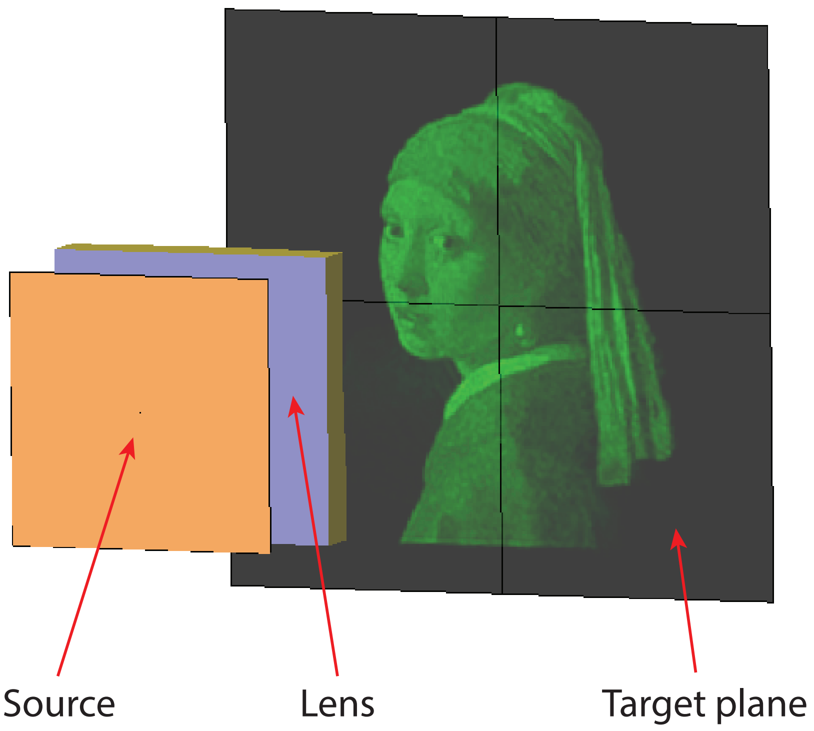


Figure 6.6: The LightTools model showing the source, lens and target plane with true color results projected onto it.

### 6.3.2. THB-SPLINE FREEFORM LENS OPTIMIZATION FINITE-ÉTENDUE

To demonstrate the effectiveness for an extended source, a smooth curved target distribution, depicted in Figure 6.7(a1). The reasoning behind the simpler target compared to the zero-etendue example is that the use of a finite étendue source limits the details that can be achieved [102, 15, 44]. The basic conics of the front and rear surfaces are set such that the initial irradiance distribution was of similar size to the desired distribution, and the B-spline of the rear surface was initialized with 5 × 5 control points while the front surface is left unchanged throughout the optimization and $z_{\text{freedom}}$ was chosen to be roughly a quarter of the thickness of the lens. An overview of all the system parameters can be found in Table 6.3.

Table 6.3: **System parameters: extended source with curved uniform target distribution**

| System Parameters | | | | Surface parameters | | | |
|---|---|---|---|---|---|---|---|
| $r_{\text{source}}$ | 0.5 mm | $d_0$ | 2 mm | $R_{\text{front}}$ | −5 mm | $N_u, N_v$ | 9 |
| $r_{\text{front}}$ | 5 mm | $d_1$ | 5 mm | $K_{\text{front}}$ | -1 | $p, q$ | 3 |
| $r_{\text{rear}}$ | 8 mm | $d_2$ | 1000 mm | $R_{\text{rear}}$ | -10 mm | $N_g, M_g$ | 450 |
| $r_{\text{target}}$ | 2000 mm | $N_t, M_t$ | 256 | $K_{\text{rear}}$ | -1 | $L$ | 5 |
| $W$ | Gaussian | $\sigma_W$ | 1 | $z_{\text{freedom}}$ | 1.5 mm | n | 1.5 |

The step sizes and refinement parameters are shown in Figure 6.7(b1). The step sizes were again chosen to balance the number of iterations to the stability of the optimization. The refinement parameters were chosen to ensure that the mirror symmetry of the hierarchical mesh was maintained as long as possible throughout the refinement process. The preservation of symmetry was critical in the earlier stages, as breaking it would result in different areas of the lens being optimized at different refinement levels, leading to worse-performing lenses. It took 29 minutes and 23 seconds to optimize the lens. At the end of each batch, the number of control points is 25, 49, 70, 253, 637, and 2180.

The graph depicting the loss shows that each refinement significantly reduces the loss. Why this is the case can be understood by analyzing the vertex gradients. In Figures 6.7(a5-d5), we can identify two distinct gradient areas: a large area in the center with a structured pattern and a ring surrounding it. It is important to note that although these areas seem disconnected, they are connected. This is due to the chosen standard deviation of the Gaussian measurement function, which causes the gradients in this intermediate area to be much smaller than in the ring or central parts of the vertex gradients when blurred. During the first four optimization batches, the structured details in the central area gradually decrease in size (Figures 6.7(a5-d5)), shaping the central, uniform part of the irradiance distribution (Figures 6.7(a6-d6)). However, after the fifth optimization batch, unstructured and low-valued gradients dominate the central domain, while the edges show large gradients, as seen in Figure 6.7(e4). In the final optimization batch, the area of the rear surface, which corresponds to the edges in the vertex gradients, is used to correct the thin lines of light protruding from the center of the distribution Figure 6.7(e6). These corrections can also be seen when looking at the changes made to the freeform surface during optimization, Figure 6.7(f1). The central area is largely left unchanged. Thus, the lower THB levels fill the uniform distribution, while the distribution

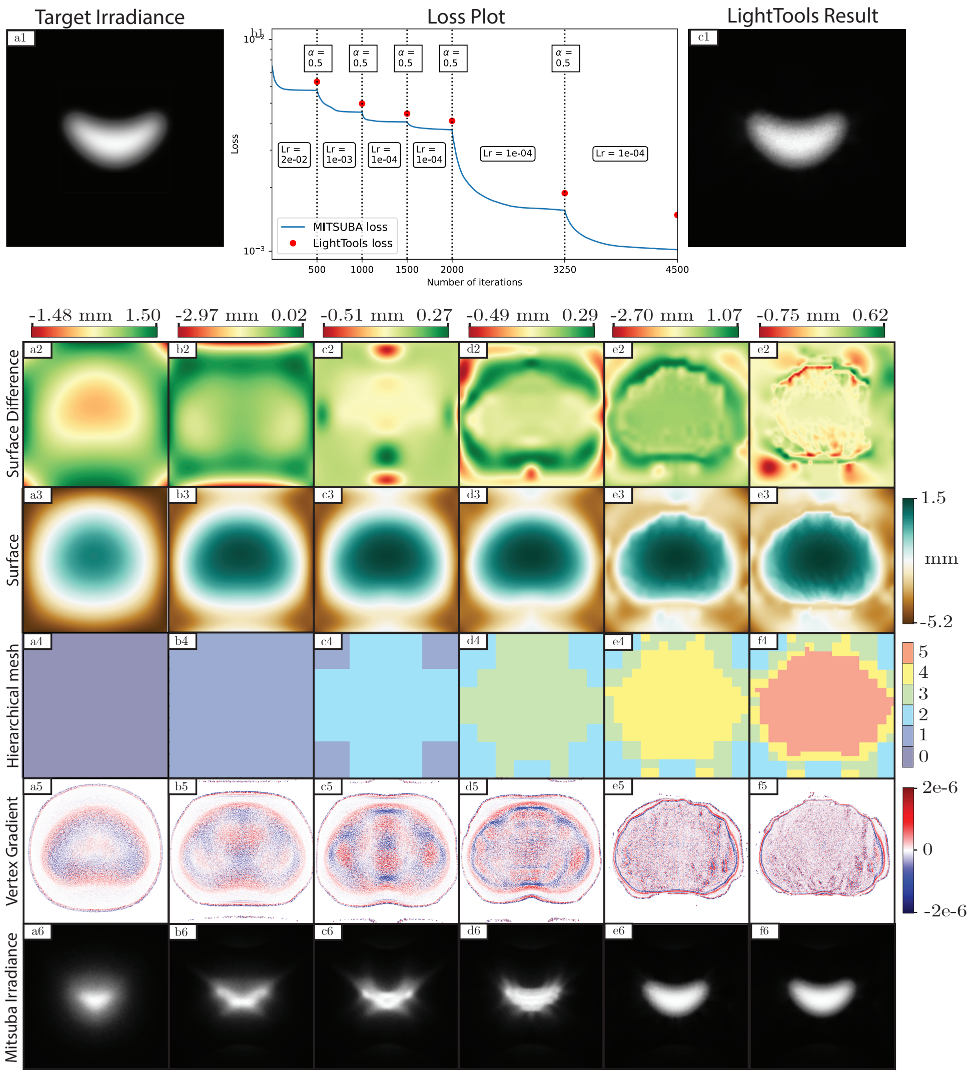


Figure 6.7: (a1) Desired irradiance distribution; (b1) Plot of loss through optimization with refinement parameters and step sizes; (c1) Irradiance obtained from LightTools; (a2-f2) The difference in the surface between subsequent optimization steps; (a3-f3) Surface at the end of each optimization step; (a4-f4) Hierarchical mesh of the THB-spline, showing which knot span products are refined; (a5-f5) Gradients of the vertices of the surface; (a6-f6) Obtained irradiance after every optimization batch.

edges are corrected at the higher level.

During later stages, large changes to the surface occur, as seen in Figures 6.7(e2,f2). However, these areas no longer affect the measured irradiance as they either do not receive any light or the light refracted by this part of the lens fails to reach the target screen.

The final results of MITSUBA (Figure 6.7(f6)) and LightTools (Figure 6.7(c1)), appear visually very similar. However, we see a structural offset in the LightTools loss shown in Figure 6.7(b1). The loss decreases similarly for both ray tracers, though their difference increases in the final batches, similar to the zero-étendue case.

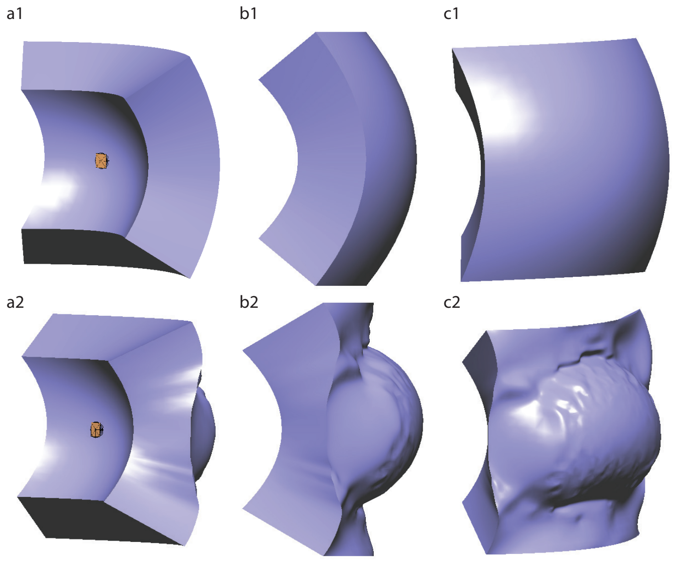


Figure 6.8: Images of the lens projecting the uniform curved distribution from left, center and right before and after optimization (a1-c1) lens before optimization; (a2-c2) lens after optimization.

Using an extended source emitting a total flux of 1 W, a total of 0.84 W gets through the lens, considering Fresnel losses of which 0.74 W ends up in the target distribution. The 0.16 W, which does not go through the lens, is reflected in the source. To investigate where the light goes that does not end up at the target screen, we follow the path of 8 non-sequential rays, using LightTools *non-sequential ray fans.* As is shown in Figure 6.10, the rays emitted by the source at the highest angles are reflected from the edge surface and totally internally reflect along the rear surface of the lens and refract back through the front surface into the source or miss it. Some rays that hit the lower or upper parts of the rear surface are refracted through the edge surface and, therefore, miss the target screen.

At the highest refinement level, the size of a knot span is $31.25 \times 31.25\ \mu\text{m}^2$. As we use splines of degree 3, the basis functions are three times the knot span. Therefore, the smallest change in the surface is approximately $95 \times 95\ \mu\text{m}^2$, which is 160 times larger than the wavelength of the light, which is $0.55\ \mu$m. Therefore, we can assume that diffrac-

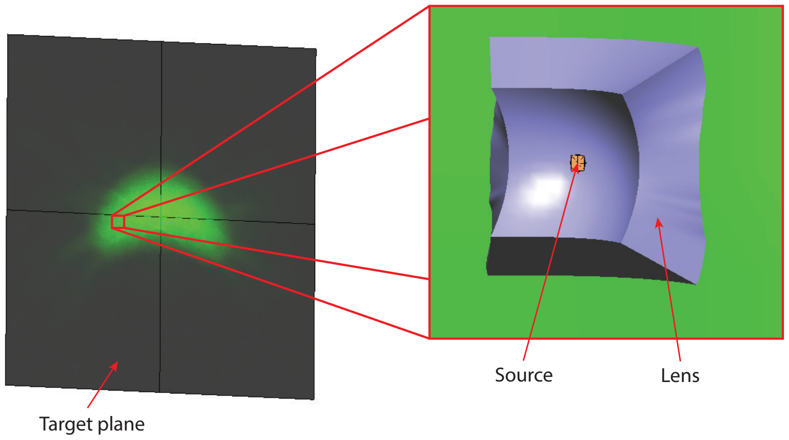


Figure 6.9: The LightTools model showing the source, lens and target plane with true color results projected onto it. As the size of the target plane is much larger than the freeform lens a zoomed in view of the lens and source is depicted.

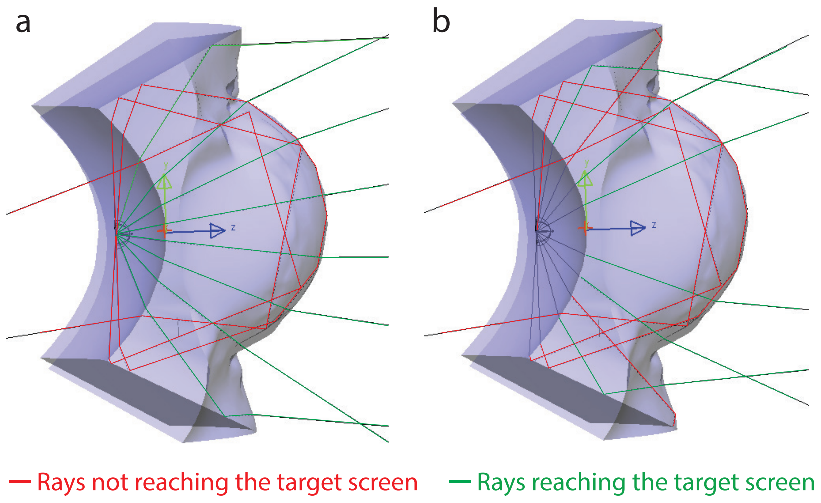

a
b
z
z
— Rays not reaching the target screen
— Rays reaching the target screen

Figure 6.10: Overview of the trajectory of different non-sequential rays, indicating which rays end up at the target screen (green) and those that do not (red) using two different types of ray sampling (a) an equi-angle sampling of 8 rays; (b) a linear sampling of 8 rays.

tion effects will be negligible.

## 6.4. CONCLUSIONS

In this chapter we demonstrated a technique to optimize freeform lenses for both zero-étendue sources and finite étendue sources. We accomplished this by utilizing the algorithmic differentiable non-sequential ray tracer MITSUBA 3, giving us access to gradients of the parameters of the freeform surface. This, combined with THB-splines, allows us to gradually increase the degrees of freedom to avoid getting trapped in undesirable local minima. The $L_1$ norm of the gradient of the vertices within a knot span was used to determine where refinement was necessary.

The method can find a freeform lens to accurately generate a prescribed target distribution for zero- and finite étendue sources. However, it is not as effective in finding solutions for zero-étendue sources as dedicated zero-etendue solves such as those which solve the Monge-Ampére equation [77]. These are capable of finding similar solutions in a much shorter time. The method excels in its adaptability, as it does not require significant changes to the algorithm to optimize for extended sources.

It is shown that a refinement strategy can significantly improve the design stability compared to simply initiating a freeform surface with many control points. In addition, using THB-splines allows the designer to keep control over where, on the freeform surface, new degrees of freedom can be added and obtain similar results to classical B-spline refinement with fewer degrees of freedom.

During optimization, the gradient vertices show interesting information, allowing us to understand how the optimization changes the surface and identify sensitive areas of the freeform surface.

Future work will focus on further developing the refinement strategy, such as investigating other metrics to determine which knot spans should be refined and a more efficient way to determine the refinement parameters. More freedom in modeling the freeform lens should be added so that it is not limited to a rectangular optimization domain. Having a more accurate way to transfer the THB-description between programs by using standardized file formats supported by LightTools, such as STEP or IGES.

# 7

# SHIFT-INVARIANT ÉTENDUE SYSTEMS

A common assumption made while designing freeform optics for illumination applications is to work with zero-étendue sources, such as point sources or highly collimated lasers. For most practical applications this assumption is too strict, as the source size or divergence is not negligible, and we must design for a finite étendue source [16, 81, 63, 36]. However, when using finite étendue sources there is no guarantee that a desired irradiance distribution is realizable with the given source. For instance, reproducing high-spatial-frequency detail can be difficult and be formulated in terms of resolution limits of freeform elements [102, 15].

Here, we further investigate when a specific finite étendue source can realize a given irradiance distribution. We model the optics as a shift-invariant *black-box* system: a shift in source position translates the output without changing its shape. Under this assumption, the irradiance from a finite étendue source equals the convolution of the source with the system's zero-étendue response. Prior work has proposed recasting extended-source design as a zero-étendue problem by deconvolving the desired irradiance with the source blur [58, 17, 1, 92]. However, the impact of source extent on the quality of achievable irradiance has received limited attention. We study this relationship by taking inspiration from deblurring images in astronomy and microscopy [20] where prior information of the system response, such as non-negativity and finite support, are used in an attempt to find a physically feasible blur kernel [21]. We use the definitions of positive definiteness and Bochner's Theorem to analyze the problem and highlight the issues in realizing the desired irradiance distribution for simple sources. Furthermore, we show that restricting the point sources to be located in a grid with equal intensities can simplify the problem and help find analytic solutions. To conclude, we propose a method of approximating the irradiance with a basis of non-negative functions and show that the approximation's quality of the desired irradiance distribution heavily depends on the chosen source for optimization. These results are then compared to regularization methods commonly used in the deblurring of images.

## 7.1. SHIFT INVARIANT RESPONSE

To analyze the problem of which irradiance distributions can be realized with a finite étendue source, we consider an optical black box system that redirects the light emitted by a source on the optical axis into an irradiance distribution $E_p$, as seen in Figure 7.1, which we call the optical impulse response of the system. We assume that the system

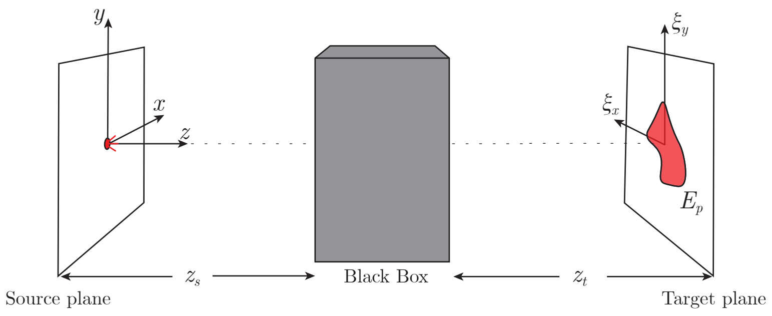


Figure 7.1: The optical black box system redirects the light emitted by a point source on the optical axis located in the source plane to generate an irradiance distribution $E_p$, called the impulse response.

is shift-invariant, meaning that a source at a position $\mathbf{r}^s$ in the source plane illuminating the optical black box system will shift the impulse response by an amount $\boldsymbol{\xi}^s$ in the target plane directly proportional to $\mathbf{r}^s$, as depicted in Figure 7.2. Now consider a set of $N_s$ mu-

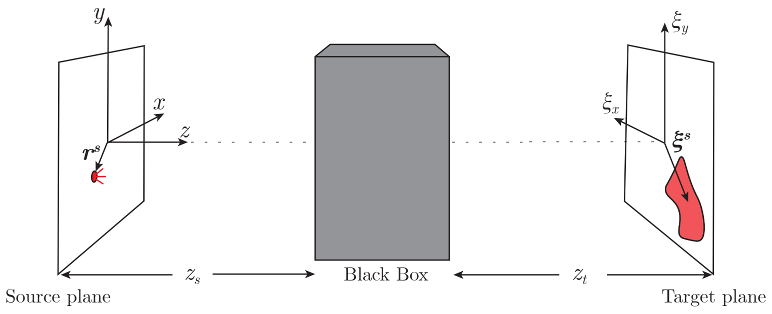


Figure 7.2: Light emitted by a point source located at $\mathbf{r}^s$ in the source plane is redirected by the optical black box system to generate $E_p$, which is shifted by an amount $\boldsymbol{\xi}^s$ with $\boldsymbol{\xi}^s \propto \mathbf{r}^s$.

tually incoherent monochromatic sources emitting the same wavelength in the source plane at $\mathbf{r}_n^s$ each with a different intensity $a_n$. Then the total irradiance at the target plane $E_{\text{tot}}$ is the incoherent sum of the contribution of each point source:

$$E_{\text{tot}}(\boldsymbol{\xi}) = \iint E_p(\boldsymbol{\xi} - \boldsymbol{\xi}^s) G(\boldsymbol{\xi}^s) \mathrm{d}\boldsymbol{\xi}^s, \tag{7.1}$$

where $G$, is the blurring caused by the source, and is defined as:

$$G(\boldsymbol{\xi}) = \sum_{n=1}^{N_s} a_n \delta(\boldsymbol{\xi}^s - \boldsymbol{\xi}_n^s). \tag{7.2}$$

Under this assumption, we can analyze the problem as a deconvolution problem where we want to find the impulse response $E_p$ using a predefined irradiance distribution $E_{\text{tot}}$ and source blurring $G$.

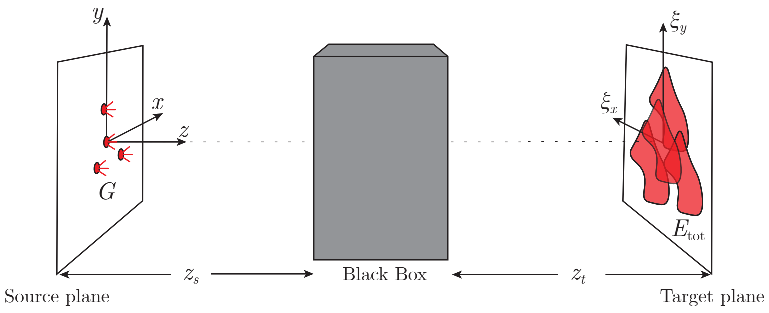


Figure 7.3: Multiple point sources in the source plane give an irradiance distribution $E_{\text{tot}}$.

## 7.2. REGULARIZED DECONVOLUTION OF THE TARGET DISTRIBUTION

Given a desired irradiance distribution $E_{\text{tot}}$ and a source blur $G$, we want to find the impulse response of the optical black box system such that when illuminated with the given source, the desired irradiance distribution is obtained. All these functions are measures of radiometric energy. Hence they are non-negative:

$$E_{\text{tot}}(\boldsymbol{\xi}), E_p(\boldsymbol{\xi}), G(\boldsymbol{\xi}) \geq 0 \text{ for all } \boldsymbol{\xi} \in \mathbb{R}^2.$$

To find an $E_p$ for a given source blur the following minimizing problem has to be solved:

$$\min_{E_p} \left\| E_{\text{tot}}(\boldsymbol{\xi}) - E_p(\boldsymbol{\xi}) * G(\boldsymbol{\xi}) \right\|_2^2, \tag{7.3}$$

where $\|\cdot\|_2$ is the $L^2$-norm. Using the Fourier transform, which we define as:

$$\mathscr{F}\{f\}(\hat{\boldsymbol{\xi}}) = \iint f(\boldsymbol{\xi}) e^{-2\pi i \boldsymbol{\xi}\cdot\hat{\boldsymbol{\xi}}} d\boldsymbol{\xi}, \tag{7.4}$$

where $\hat{\boldsymbol{\xi}}$ is the reciprocal coordinate of $\boldsymbol{\xi}$, and $\hat{f}(\boldsymbol{\xi}) = \mathscr{F}\{f\}(\boldsymbol{\xi})$. The inverse Fourier transform is defined as

$$\mathscr{F}^{-1}\{f\}(\boldsymbol{\xi}) = \iint f(\hat{\boldsymbol{\xi}}) e^{2\pi i \boldsymbol{\xi}\cdot\hat{\boldsymbol{\xi}}} d\hat{\boldsymbol{\xi}}. \tag{7.5}$$

A solution for $E_p$ can then be found in Fourier space [20]:

$$E_p(\boldsymbol{\xi}) = \mathscr{F}^{-1}\left\{\frac{\widehat{G}(\widehat{\boldsymbol{\xi}})\widehat{E}_{\text{tot}}(\widehat{\boldsymbol{\xi}})}{|\widehat{G}(\widehat{\boldsymbol{\xi}})|^2+\varepsilon}\right\}(\boldsymbol{\xi}), \tag{7.6}$$

where $\varepsilon$ prevents division by zero. Using Equation7.6 let us look at the solution obtained when using the 1D rectangle function as the desired irradiance distribution is:

$$E_{\text{tot}}(\xi) = \text{rect}\left(\frac{\xi_x}{\alpha}\right) \quad \text{with} \quad \text{rect}\left(\frac{\xi_x}{\alpha}\right) \equiv \begin{cases} 1 & \text{if} \quad |\xi_x| \le \alpha, \\ 0 & \text{if} \quad |\xi_x| > \alpha. \end{cases}, \tag{7.7}$$

with two sources of equal strength. Fig.7.4.a0 shows the desired irradiance distribution $E_{\text{tot}}(\xi_x) = \text{rect}(\xi_x/0.5)$ with a source blur $G(\xi_x) = \delta(\xi_x+0.25)+\delta(\xi_x-0.25)$. The $E_p(\xi_x)$ obtained from Equation7.6 with $\varepsilon = 10^{-14}$ is shown in Fig.7.4.b0. It is a non-negative function with bounded support. However, when we slightly change the width of $E_{\text{tot}}$ to $\alpha = 0.51$ while keeping $G$ unchanged, $E_p$ as given by Equation7.6, the obtained impulse response as seen in Fig.7.4.b1, has negative values and does not have bounded support anymore. Something similar happens when leaving $E_{\text{tot}}$ unchanged, and the source positions are slightly changed. As seen in Fig.7.4.b2, resulting in $E_p$ oscillating rapidly, has no finite support and negative values.

From these results, it is clear that this problem is ill-posed and is very sensitive to perturbations of the desired irradiance distribution and the a source blur. To better understand when a non-negative $E_p$ is obtained and what the requirements should be imposed on $E_{\text{tot}}$ and $G$ to assure this, we can make use of *positive-definite functions* (Definition 1) and *Bochner's Theorem* (Theorem 2).

**Definition 1 (Positive definite functions)** *A continuous function $\Phi : \mathbb{R}^n \to \mathbb{C}$ is positive definite on $\mathbb{R}^n$ if for every $N \ge 1$ and every $\mathbf{x}_1, \ldots, \mathbf{x}_N \in \mathbb{R}^n$, there holds:*

$$\sum_{j=1}^{N}\sum_{k=1}^{N} c_j \bar{c_k} \Phi(\mathbf{x}_j - \mathbf{x}_k) \ge 0$$

*for all complex numbers $[c_1, \cdots, c_N]^T \in \mathbb{C}$. Hence the matrix with elements $\Phi(x_j - x_k)$ is hermitian and non-negative.*

**Theorem 1 (Properties of positive-definite functions [31])**

a) *Any non-negative finite linear combination of positive definite function is positive definite., i.e., if $\Phi_1, \cdots \Phi_m$ are positive definite on $\mathbb{R}^n$ and $w_j \ge 0$ for all $j = 1, \cdots, m$, then:*

$$\Phi(\mathbf{x}) = \sum_{j=1}^{m} w_j \Phi_j(\mathbf{x}), \quad \mathbf{x} \in \mathbb{R}^n$$

*is also positive definite on $\mathbb{R}^n$.*

b) *For any positive definite function: $\Phi(\mathbf{0}) \ge 0$.*

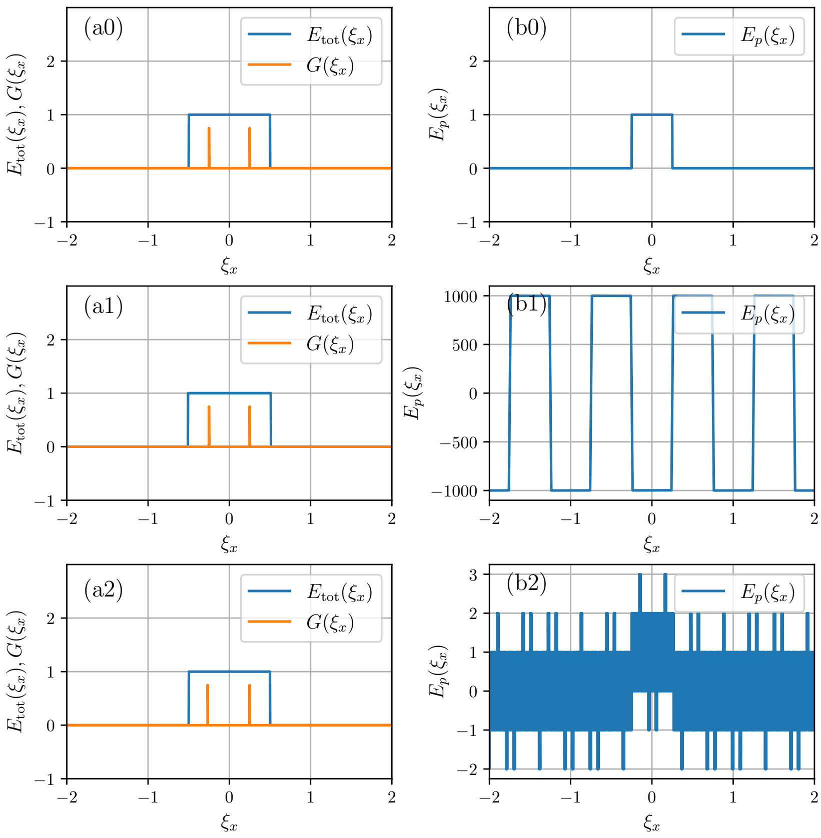


Figure 7.4: Results from applying regularized deconvolution to a rectangular function when $\varepsilon = 10^{-14}$: (a0) $E_{\text{tot}}(\xi_x) = \text{rect}(\xi_x/0.5)$ and $G(\xi_x) = \delta(\xi_x + 0.25) + \delta(\xi_x - 0.25)$; (b0) $E_p(\xi_x)$; (a1) $E_{\text{tot}}(\xi_x) = \text{rect}(1.02\xi_x/0.51)$ and $G(\xi_x) = \delta(\xi_x + 0.25) + \delta(\xi_x - 0.25)$ ; (b1) $E_p(\xi_x)$ ; (a2) $E_{\text{tot}}(\xi_x) = \text{rect}(\xi_x/0.5)$ and $G(\xi_x) = \delta(\xi_x + 0.25) + \delta(\xi_x - 0.2501)$; (b2) $E_p(\xi_x)$ obtained for ; Results are obtained using linear sampling of $\xi_x$ on the domain $[-5, 5]$ with $N = 1000001$ points.

c) *For any positive definite function:* $\Phi(-\mathbf{x}) = \overline{\Phi(\mathbf{x})}$.

d) *Any positive definite function is bounded. In fact,*

$$|\Phi(\mathbf{x})| \leq \Phi(\mathbf{0}) \quad \textit{for all}\, \mathbf{x} \in \mathbb{R}^n.$$

e) *If* $\Phi$ *is positive definite with* $\Phi(\mathbf{0}) = 0$ *then* $\Phi = 0$.

*f) The product of positive definite functions is positive definite.*

**Theorem 2 (Bochner's Theorem [31])** *A (complex-valued) function $\Phi \in C(\mathbb{R}^n)$ is positive definite on $\mathbb{R}^n$ if and only if it is the Fourier transform of a bounded Borel measure $\mu$ on $\mathbb{R}^n$, i.e.*

$$\Phi(\mathbf{x}) = \widehat{\mu}(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^n}} \int_{\mathbb{R}^n} e^{-\mathrm{i}\mathbf{x}\cdot\mathbf{y}} d\mu(\mathbf{y}), \quad \mathbf{x} \in \mathbb{R}^n.$$

Since the functions $E_{\text{tot}}, K$, and $E_p$ are non-negative, Bochner's theorem implies that the Fourier transforms $\widehat{E}_{\text{tot}}, \widehat{G}$ and $\widehat{E}_p$ are all positive definite. When applying regularized deconvolution of Equation7.6 we can use Property.1f: positive definiteness is preserved under multiplication. This property guarantees that $\widehat{E}_p$ is positive definite if $1/\widehat{G}$ is also positive definite but is not a necessary condition. It is simple to show that this cannot be the case because: $\widehat{G}(\mathbf{0}) \geq |\widehat{G}(\widehat{\boldsymbol{\xi}})|$ for all $\widehat{\boldsymbol{\xi}} \in \mathbb{R}^2$ implies that $1/\widehat{G}(\mathbf{0}) \leq |1/\widehat{G}(\widehat{\boldsymbol{\xi}})|$ for all $\widehat{\boldsymbol{\xi}} \in \mathbb{R}^2$. Hence, if $1/\widehat{G}$ were positive definite: $1/\widehat{G}(\mathbf{0}) \leq |1/\widehat{G}(\widehat{\boldsymbol{\xi}})| \leq 1/\widehat{G}(\mathbf{0})$ requiring $|\widehat{G}|$ to be constant which is only possible when only a single source is used. Therefore, if multiple sources are used, it is not possible for $|\widehat{G}(\widehat{\boldsymbol{\xi}})|$ to be constant. Thus, $1/\widehat{G}$ is in general not positive definite, and hence $\widehat{E}_{\text{tot}}/\widehat{G}$ is in general also not positive definite, and hence $\widehat{E}_p$ is not positive definite. Besides the single source solution, a second trivial case exists where $G(\boldsymbol{\xi}) = E_{\text{tot}}(\boldsymbol{\xi})$ with $E_p(\boldsymbol{\xi}) = \delta(\boldsymbol{\xi})$. This case is realized by choosing the distribution $E_{\text{tot}}$ as the source and projecting it to the desired target plane using an imaging system.

One can view these two trivial solutions as two extreme solutions to the problem of realizing the desired irradiance using a combination of source and optical systems. The first solution corresponds to putting all the information into the optical system and only using a single source. In contrast, the second solution corresponds to shaping the source distribution and imaging it. The challenge is to find useful solutions between these two extremes. Therefore, we could reformulate the minimization described in Equation7.3 by treating both the blur and impulse response as variables leading to a blind deconvolution problem given by:

$$\min_{E_p, G} \left\| E_{\text{tot}}(\boldsymbol{\xi}) - E_p(\boldsymbol{\xi}) * G(\boldsymbol{\xi}) \right\|_2^2. \tag{7.8}$$

Algorithms such as iterative blind-deconvolution [5] and Richardson-Lucy deconvolution [56, 74, 33] can then be used to find both the source and impulse response of the system. However, despite our efforts in applying these methods, they have yet to produce significant results. Therefore, we limit our analysis to cases where the source is given.

## 7.2.1. ANALYTIC EXAMPLE

To be able to analyze the problem with an analytic example, we impose a couple of restrictions on the source blur, given by Equation7.2 of which the Fourier transform is:

$$\widehat{G}(\widehat{\boldsymbol{\xi}}) = \sum_{i=0}^{N_s} |a_i|^2 \exp(\mathrm{i}\widehat{\boldsymbol{\xi}} \cdot \boldsymbol{\xi}_i). \tag{7.9}$$

The sum of complex exponentials can be rewritten as a complex function:

$$\widehat{G}(\widehat{\boldsymbol{\xi}}) = |\widehat{G}(\widehat{\boldsymbol{\xi}})| \exp(\mathrm{i}\Phi_{\widehat{G}}(\widehat{\boldsymbol{\xi}})), \tag{7.10}$$

where $\Phi_{\widehat{G}}$ is the phase of the complex function and $|\widehat{G}|$ is the modulus, which can be written as:

$$|\widehat{G}(\widehat{\boldsymbol{\xi}})| = \sqrt{\sum_{n=0}^{N_s} a_n^2 + \sum_{n\neq m} 2a_n a_m \cos(\widehat{\boldsymbol{\xi}} \cdot (\boldsymbol{\xi}_n - \boldsymbol{\xi}_m))}, \tag{7.11}$$

$$\Phi_{\widehat{G}}(\widehat{\boldsymbol{\xi}}) = \arctan\left(\frac{\sum_{n=0}^{N_s} a_n \sin(\widehat{\boldsymbol{\xi}} \cdot \boldsymbol{\xi}_n)}{\sum_{n=0}^{N_s} a_n \cos(\widehat{\boldsymbol{\xi}} \cdot \boldsymbol{\xi}_n)}\right). \tag{7.12}$$

To obtain a valid solution, $\widehat{E}_{\text{tot}}/\widehat{G}$ should, according to Property.4.d, be bounded, which is the case when all the zeros of $|\widehat{G}|$ are also zeros of $\widehat{E}_{\text{tot}}$. However, $|\widehat{G}|$ is a cosine polynomial of which only a lower bound can be given for the number of zeros [10, 78] making it extremely challenging to ensure that all the zeros are found. Two assumptions can be made to simplify Equation7.11 and Equation7.12. The first assumes that all the sources have the same intensity $a_n = a$ for all $n = 1, \ldots, N_s$. The second restricts the source positions to be equidistant with some separation $\Delta\boldsymbol{\xi} = [\Delta\xi_x, \Delta\xi_y]^T$, such that $\Delta\boldsymbol{\xi}_n = n\Delta\boldsymbol{\xi}$. Combining these assumptions gives the following expression for Equation7.9

$$\widehat{G}(\widehat{\boldsymbol{\xi}}) = a \sum_{n=0}^{N_s} \exp(\mathrm{i}\widehat{\boldsymbol{\xi}} \cdot n\Delta\boldsymbol{\xi}), \tag{7.13}$$

which can simplified to:

$$\widehat{G}(\widehat{\boldsymbol{\xi}}) = a\frac{\sin\left(N_s\widehat{\boldsymbol{\xi}} \cdot \Delta\boldsymbol{\xi}/2\right)}{\sin\left(\widehat{\boldsymbol{\xi}} \cdot \Delta\boldsymbol{\xi}/2\right)} \exp(\mathrm{i}N_s\widehat{\boldsymbol{\xi}} \cdot \Delta\boldsymbol{\xi}/2). \tag{7.14}$$

This expression is closely linked to the Dirichlet kernel [7], and it enables the analytical analysis of specific desired irradiance distributions.

Again consider the 1D rectangle function of Equation7.7 as the desired irradiance distribution then its Fourier transform is

$$\widehat{E}_{\text{tot}}(\widehat{\xi}_x) = \frac{1}{a}\operatorname{sinc}\left(\frac{\pi\widehat{\xi}_x}{a}\right) = \frac{\sin(\pi\widehat{\xi}_x/a)}{\pi\widehat{\xi}_x}. \tag{7.15}$$

We can calculate $E_p$ using the simplified kernel Equation7.14:

$$\widehat{E}_p(\widehat{\xi}_x) = \frac{\sin(\pi\widehat{\xi}_x/a)}{\pi\widehat{\xi}_x}\frac{\sin(\widehat{\xi}_x\Delta\xi/2)}{\sin(N_s\widehat{\xi}_x\Delta\xi/2)} \exp(-\mathrm{i}\widehat{\xi}_x\Delta\xi(N_s-1)/2). \tag{7.16}$$

By setting $\Delta\xi = 2\pi/N_s a$ the sine in the denominator is canceled by the fist sinus in the numerator, leaving us with:

$$\widehat{E}_p(\widehat{\xi}_x) = \frac{1}{\pi\widehat{\xi}_x}\sin\left(\frac{\pi\widehat{\xi}_x}{N_s a}\right)\exp(-\mathrm{i}\widehat{\xi}_x\widetilde{\Delta\xi}), \tag{7.17}$$

where $\widetilde{\Delta\xi} = \Delta\xi(N_s-1)/2$ is used to simplify the expression. We can rewrite this expression as a sinc function:

$$\widehat{E}_p(\widehat{\xi}_x) = \frac{1}{N_s a}\operatorname{sinc}\left(\frac{\pi\widehat{\xi}_x}{N_s a}\right)\exp(-\mathrm{i}\widehat{\xi}_x\widetilde{\Delta\xi}). \tag{7.18}$$

The inverse Fourier transformed of Equation7.18 then gives the solution for the impulse response

$$E_p(\xi_x) = \mathrm{rect}\Big(aN_s\xi_x - \widetilde{\Delta\xi}\Big), \tag{7.19}$$

which is depicted in Figure 7.5.b0. This expression shows that as the number of sources increases, an equal amount of rectangles can be placed next to each other to get back the original rectangle.

It should be noted that Equation7.19 is one of many solutions we can obtain. By rewriting Equation7.16 using the sine double angle formula, we can find:

$$\widehat{E}_p(\widehat{\xi}_x) = \frac{2}{\pi\widehat{\xi}_x}\sin\left(\frac{\pi\widehat{\xi}_x}{2a}\right)\cos\left(\frac{\pi\widehat{\xi}_x}{2a}\right)\frac{\sin(\widehat{\xi}_x\Delta\xi/2)}{\sin(N_s\widehat{\xi}_x\Delta\xi/2)}\exp(-\mathrm{i}\widehat{\xi}_x\Delta\xi(N_s-1)/2). \tag{7.20}$$

By choosing $\Delta\xi = \pi/(N_s a)$ the sinus in the denominator is canceled by the first sinus on the right-hand side of Equation7.20, an alternative solution for the impulse response:

$$E_p(\xi_x) = \mathrm{rect}\Big(2N_s a\xi_x - \widetilde{\Delta\xi}\Big) * \left[\delta\Big(\xi_x + \frac{1}{4a}\Big) + \delta\Big(\xi_x - \frac{1}{4a}\Big)\right]. \tag{7.21}$$

This solution can be understood as dividing the rectangle into $2N_s$ rectangles. The impulse response equals the combination of the first and the $(N_s+1)^{\mathrm{th}}$ rectangle, as seen in Figure 7.5.a1.

The double angle formula can be applied an arbitrary number of times, and gives the general expression for when it is applied $M$ times

$$\widehat{E}_p(\widehat{\xi}_x) = \frac{2^M}{\pi\widehat{\xi}_x}\sin\left(\frac{\pi\widehat{\xi}_x}{2^M a}\right)\frac{\sin(\widehat{\xi}_x\Delta\xi/2)}{\sin(N_s\widehat{\xi}_x\Delta\xi/2)}\exp(-\mathrm{i}\widehat{\xi}_x\Delta\xi(N_s-1)/2)\prod_{m=1}^{M}\cos\left(\frac{\pi\widehat{\xi}_x}{2^m}\right). \tag{7.22}$$

7

By choosing $\Delta\xi = \pi/2^M N_s a$, the sinus in the denominator is again canceled. By Fourier Transformation of the resulting expression, we get:

$$E_p(\xi_x) = \mathrm{rect}\Big(2^M a\xi_x - \widetilde{\Delta\xi}\Big) * \underset{m=1}{\overset{M}{\circledast}}\left[\delta\Big(\xi_x + \frac{1}{2^{m+1}a}\Big) + \delta\Big(\xi_x - \frac{1}{2^{m+1}a}\Big)\right]. \tag{7.23}$$

The $\circledast_{m=1}^{M}$ is used to denote the $M$ times repeated convolution:

$$f_1 * f_2 * \cdots * f_M = \underset{m=1}{\overset{M}{\circledast}} f_m. \tag{7.24}$$

This solution can be understood as dividing the rectangle into $MN_s$ rectangles. The impulse response equals the combination of the first and every $(N_s+1)^{\mathrm{th}}$ rectangle after that, in Figure 7.5.b2 the result is shown where the double sine angle is applied three times.

Based on this example, we can see the importance of correctly choosing the number of sources and the distance between them because, otherwise, the sines in Equation7.16 do not cancel. However, even with the assumptions used, finding an analytic expression for the impulse response $E_p$ is only possible for a limited amount of cases. Therefore, estimating the desired irradiance distribution using a basis guaranteed to have a solution provides a more general approach.

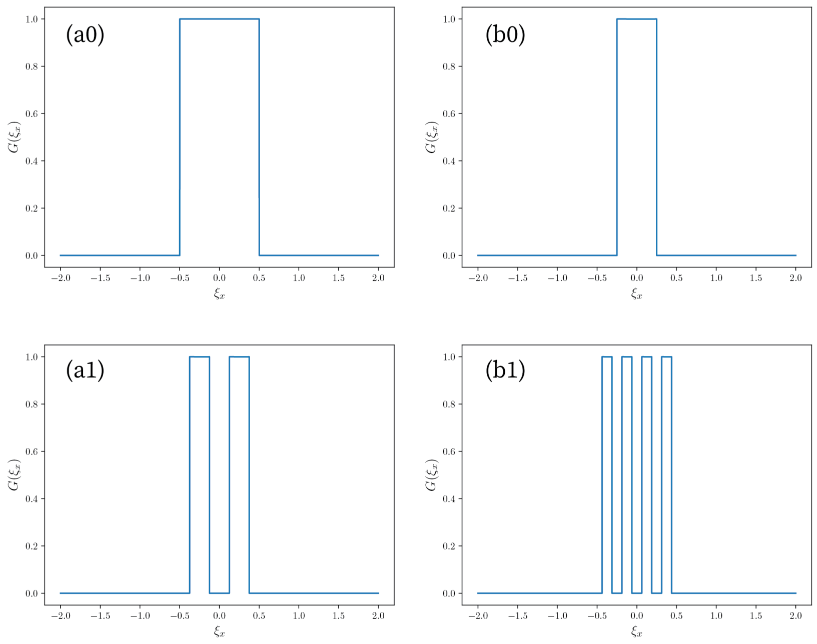


Figure 7.5: Visualization of results obtained by applying Equation7.19, Equation7.21 and Equation7.23 with $N_s = 2$ (a0) Desired irradiance distribution; (b0) Impulse response obtained using Equation7.19; (a1) Impulse response obtained using Equation7.21; (b1) Impulse response obtained using Equation7.23 with $M = 3$;

## 7.3. APPROXIMATING THE IRRADIANCE DISTRIBUTION THROUGH OPTIMIZATION

As shown in Section.7.2.1 finding an analytic expression for the impulse response is only possible for a limited set of desired irradiance distributions. In addition, most irradiance distributions do not have an analytic expression, and we must turn to optimization to find a suitable $E_p$ given an irradiance distribution $E_{\text{tot}}$ and source blurring $G$.

To implement the minimization algorithm to solve Eq.7.3, we formulate the discretized problem. The matrices $\mathbf{E}_{\text{tot}}$, $\mathbf{E}_p$, $\mathbf{G}$ are the discrete counterparts of $E_{\text{tot}}$, $E_p$, $G$ and are matrices of dimension $N \times N$. Furthermore, to write Eq.7.3 in terms of matrix and vector multiplications, we use the vectorization vec operator, which for a matrix

$$\mathbf{A} = \begin{bmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,n} \\ a_{2,1} & a_{2,2} & \dots & a_{2,n} \\ \vdots & \ddots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \dots & a_{m,n} \end{bmatrix} \tag{7.25}$$

is defined as

$$\text{vec}(\mathbf{A}) = \begin{bmatrix} a_{1,1}, & \dots, & a_{1,n}, & a_{2,1}, & \dots, & a_{2,n}, & a_{m,1}, & \dots & a_{m,n} \end{bmatrix}^T. \tag{7.26}$$

In addition, we define the toeplitz operator for a general vector:

$$\mathbf{a} = \begin{bmatrix} a_1, & a_2, & a_3, & \dots, & a_{n-2}, & a_{n-1}, & a_n \end{bmatrix} \tag{7.27}$$

as:

$$\text{toepl}(\mathbf{a}) = \begin{bmatrix} a_1 & 0 & \dots & 0 & 0 \\ a_2 & a_1 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n-1} & a_{n-2} & \ddots & a_1 & 0 \\ a_n & a_{n-1} & \ddots & a_2 & a_1 \end{bmatrix}. \tag{7.28}$$

and the reshaping operator $\text{resh}_{n\times m}$:

$$\text{resh}_{n\times m}(\mathbf{a}) = \begin{bmatrix} a_1 & a_2 & \dots, & a_m \\ a_{m+1} & a_{m+2} & \dots, & a_{2m} \\ \vdots & \ddots & \ddots & \vdots \\ a_{(n-1)m+1} & a_{(n-1)m+2} & \dots & a_{nm} \end{bmatrix}, \tag{7.29}$$

which takes a vector of size $NM$ and reshapes it into a matrix of size $N \times M$ such that for a square matrix $\mathbf{M} \in \mathbb{R}^{N\times N}$ we have $\mathbf{M} = \text{resh}_{N\times N}\big(\text{vec}(\mathbf{M})\big)$.
Using these operators the convolution of two matrices can be written as a matrix vector multiplication:

$$\text{vec}(\mathbf{G} * \mathbf{E}_p) = \text{toepl}(\text{vec}(\mathbf{G}))\text{vec}(\mathbf{E}_p). \tag{7.30}$$

We can then formulate the discrete minimization problem as:

$$\min_{\mathbf{E}_p} \left\| \text{vec}(\mathbf{E}_{\text{tot}}) - \text{toepl}\,[\text{vec}(\mathbf{G})]\,\text{vec}(\mathbf{E}_p) \right\|_2^2. \tag{7.31}$$

While solving Equation7.31, it is crucial to include prior information such as non-negativity and finite support of the solution. To accomplish this, we describe two approaches: one involves approximating the desired irradiance distribution using non-negative basis functions, while the other utilizes regularization techniques.

### 7.3.1. APPROXIMATION USING NON-NEGATIVE BASIS FUNCTIONS

We define a set of non-negative functions $\{P_1, P_2, \cdots, P_n\}$ with coefficient $\omega_i > 0$ to approximate the distribution

$$E_{\text{tot}}(\boldsymbol{\xi}) \approx \sum_i \omega_i P_i(\boldsymbol{\xi}) * G(\boldsymbol{\xi}), \quad \omega_i \geq 0. \tag{7.32}$$

Using this basis we can then reformulate Equation7.3 as finding the optimal coefficients, such that the following expression is minimized

$$\min_{\omega_1,\omega_2,\dots,\omega_n} \left\| E_{\text{tot}}(\boldsymbol{\xi}) - \sum_i G(\boldsymbol{\xi}) * \omega_i P_i(\boldsymbol{\xi}) \right\|_2^2. \tag{7.33}$$

The distribution $E_p$ is then obtained by summing the weighted basis functions:

$$E_p(\boldsymbol{\xi}) = \sum_i \omega_i P_i(\boldsymbol{\xi}) \tag{7.34}$$

A suitable choice for $P$ is any probability distribution with finite support such as Beta distributions, Bates distributions, Irwin distributions, or Kronecker delta distributions [48]. There are two ways of forming a basis for a chosen distribution. First, probability density functions defined by shape parameters, such as the beta distribution with shape parameters $\alpha$ and $\beta$:

$$P(x) = x^{\alpha-1}(1-x)^{\beta-1}, \quad \alpha, \beta \geq 0 \quad \text{and} \quad 0 \geq x \geq 1, \tag{7.35}$$

allow for creating a non-orthogonal basis by choosing a range over which to define $\alpha$ and $\beta$ and discretize it. For instance, select the range $\alpha \in [0, A]$ and $\beta \in [0, B]$ and the amount of functions in the set using $N_\alpha$ and $N_\beta$. Then the following set of basis functions is obtained:

$$P_{i,j}(x) = x^{iA/N_\alpha - 1}(1-x)^{jB/N_\beta - 1}, \quad \text{with} \quad i = 0,1,\ldots,N_\alpha \quad \text{and} \quad j = 0,1,\ldots,N_\beta. \tag{7.36}$$

Secondly, a probability density function $P(x)$ which does not have shape parameters, such as the Irwan-Hall distributions, can give a basis by spatially shifting $P(x)$ over a distance $x_i$ by which the basis function becomes:

$$P_i(x) = P(x) * \delta(x - x_i). \tag{7.37}$$

To obtain the discrete optimization problem we define matrix $\mathbf{P} \in \mathbb{R}^{N^2 \times M}$ as the concatenation of vectorized basis functions

$$\mathbf{P} = \left[\mathrm{vec}(\mathbf{P}_1), \quad \mathrm{vec}(\mathbf{P}_2), \quad \ldots, \quad \mathrm{vec}(\mathbf{P}_M)\right], \tag{7.38}$$

then $\mathrm{vec}(\mathbf{E}_p)$ can be calculated as:

$$\mathrm{vec}(\mathbf{E}_p) = \mathbf{P}\boldsymbol{\omega}, \tag{7.39}$$

where $\boldsymbol{\omega} \in \mathbb{R}^{M \times 1}$ is a vector containing the weights of the basis functions. Combining Equation7.31 and Equation7.39 the discrete minimization problem becomes:

$$\min_{\boldsymbol{\omega}} \left\| \mathrm{vec}(\mathbf{E}_{\mathrm{tot}}) - \mathrm{toepl}\left[\mathrm{vec}(\mathbf{G})\right] \mathbf{P}\boldsymbol{\omega} \right\|_2^2 \quad \text{subject to} \quad \boldsymbol{\omega} \geq 0, \tag{7.40}$$

which can be solved using non-negative least squares [67, 35]. Once a $\boldsymbol{\omega}$ is found which minimizes Eq.7.40 or a maximum number of iterations is reached, $\mathbf{E}_p$ can be calculated as:

$$\mathbf{E}_p = \mathrm{resh}_{N \times N}\left(\mathbf{P}\,\boldsymbol{\omega}\right). \tag{7.41}$$

## 7.4. RESULTS

We consider the case of 2 sources with equal intensity, yielding the following source blur:

$$G(\boldsymbol{\xi}) = \delta(\boldsymbol{\xi}) + \delta(\boldsymbol{\xi} + \boldsymbol{\Delta\xi}). \tag{7.42}$$

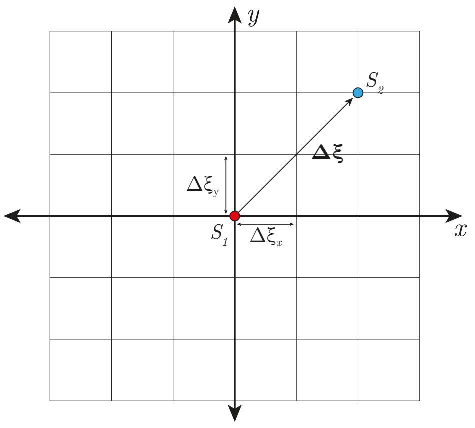


Figure 7.6: Schematic representation of how the two sources are defined: the first source is located at the origin (on the optical axis), the second has position vector $\boldsymbol{\Delta\xi} = (m\Delta\xi_x, n\Delta\xi_y)$.

One source is fixed at the center of the source plane such that $\boldsymbol{\xi}^s = 0$ while the other can move freely with a position $\boldsymbol{\Delta\xi}$. A schematic representation is shown in Figure 7.6. For all cases, the target irradiance distribution is chosen to be of size $256 \times 256$, and the basis chosen for optimization consists of shifted Kronecker delta functions,

$$\delta(\boldsymbol{\xi}, \boldsymbol{\xi}_0) = \begin{cases} 0, & \text{if } \boldsymbol{\xi} \neq \boldsymbol{\xi}_0 \\ 1, & \text{if } \boldsymbol{\xi} = \boldsymbol{\xi}_0 \end{cases} . \tag{7.43}$$

The position of the second source is incrementally changed for source positions $\boldsymbol{\Delta\xi} = (m\Delta\xi_x, n\Delta\xi_y)$ with $m, n \in [0, 20]$ and $\Delta\xi_x, \Delta\xi_y = 1/128$. The size of $\mathbf{G}$ becomes very large. However, due to the choice of source blur, the matrix toepl(vec($\mathbf{G}$)) only has $2N^2$ non-zero elements allowing the use of sparsity. At each position, Equation7.3 is solved using non-negative least squares, which is stopped once a maximum amount of iterations has exceeded or has not decreased for several iterations, yielding a solution for $E_p(\boldsymbol{\xi})$. The final $L^2$ norm value is stored in a matrix of size $20 \times 20$, called the loss matrix, when plotted, shows a grid of the loss values, called the loss landscape. The loss landscape visualizes how well the optimization converges for the different source configurations.

The optimization was done for two desired irradiance distributions: a uniform square and a uniform circle, of which the results can be seen in Figure 7.7 and Figure 7.8 respectively.

The loss landscape of the uniform square distribution, Figure 7.7.b0, shows several positions where good estimation is achieved, which are situated along the $x$ and $y$-axis. The impulse responses obtained for two cases are shown in Figure 7.7.b1 and Figure 7.7.b3. These solutions are equivalent to the analytical solutions found by applying the double sine formula in Equation7.23 in the $x$ or $y$-direction. Moving the source away from the $x$ or $y$ axis causes a degradation of the quality of the obtained irradiation distribution. The obtained distribution is shown in Figure 7.7.a2, and in Figure 7.8.b2, the respective impulse response.

For the uniform circle, we see that the desired distribution can be accurately estimated when the distance between the two sources is small compared to the distribution size, as seen in Figure 7.8.a1. As the distance between the sources increases, the estimation quality further degrades.

In all results, we see that if the shift induced by moving the source is small with respect to the size of the target irradiation distribution, an impulse response can be found, which can be used to approximate the desired distribution accurately.

### 7.4.1. COMPARISON WITH REGULARIZATION

We compare the results of the non-negative basis function approximation with three types of regularization: maximum entropy, Tikhonov regularization with $\mathbf{L}$ the identity operator, and Tikhonov regularization with $\mathbf{L}$ the discrete Laplace operator, which is commonly used in edge detection [13] and is chosen to enforce smoothness of the solution and dampen out the wild oscillation observed in Figure 7.4.b1 and Figure 7.4.b2. We solved the regularized problems using *Regularization Tools* [42] employing different solvers for the regularized problems. The *maxent* solver was used to solve for the maximum entropy regularization, the conjugate gradient algorithm (*cgls*) for the Thikonov with identity regularization, and the preconditioned conjugate gradient algorithm (*pcgls*) for the discrete Laplace operator regularization. We compare the results of two sets of source positions for both the square and circular distributions. For the square distribution, we compare the results for the source positions: $\boldsymbol{\Delta\xi} = (8\Delta\xi_x, 0)$ (Figure 7.9) and $\boldsymbol{\Delta\xi} = (6\Delta\xi_x, 6\Delta\xi_y)$ (Figure 7.10). For the circular distribution, we compare the results of source positions: $\boldsymbol{\Delta\xi} = (0\Delta\xi_x, 4\Delta\xi_y)$ (Figure 7.11) and $\boldsymbol{\Delta\xi} = (13\Delta\xi_x, 4\Delta\xi_y)$ (Figure 7.12). For both cases of the square distribution, the regularization parameter used to solve the maximum entropy was set to $\mu = 0.4642$. Both the preconditioned and regular conjugate gradient algorithms converged in 50 iterations. For the circular distribution, $\mu = 0.315$ was chosen for the maximum entropy algorithm, and the preconditioned and regular conjugate gradient algorithm converged in 150 iterations.

In all test cases, the maximum entropy regularization produced non-negative impulse responses and total irradiances, which, upon visual inspection, closely resembled the outcomes obtained through approximation by non-negative basis functions. The results obtained using the conjugate gradient method converge to the known non-negative solution as seen in Figure 7.9.a1 and Figure 7.9.b1. However, in all other scenarios, the obtained impulse response oscillates rapidly between positive and negative values and lacks finite support, as depicted in Figure 7.10.b2, Figure 7.11.b2 and Figure 7.12.b2. Furthermore, the impulse responses obtained using the preconditioned conjugate gradient method with the discrete Laplace operator are much smoother than the other results, as shown in Figure 7.10.b2, Figure 7.11.b2 and Figure 7.12.b2. Although both impulse response and total irradiance become negative, the amount is much less than the results obtained using the regular conjugate gradient algorithm. Moreover, while the preconditioned conjugated gradient result extends beyond the desired irradiance domain, the oscillation appears to be damped towards the edges.

## 7.5. CONCLUSION

We have presented a mathematical study of the problem of generating a desired irradiance distribution under the assumption that the irradiance distributions generated by different point sources are the same except for a translation. Under this assumption can be analyzed as a deconvolution problem where the desired irradiance distribution, illumination, and impulse response should all be non-negative. Using positive definite functions and Bochner's theorem, we have shown two trivial solutions: one uses a single, zero-étendue source; the other shapes the source to be the desired irradiance distribution and designs an imaging system that projects it to the desired plane. When restricted to equidistantly spaced sources with equal strength, an analytic solution for $E_p$ can be found in specific cases. However, a more general approach is obtained through optimization using a set of non-negative basis functions. Analysis of the results showed, for the case of two sources, that if the shift induced by moving the source is small compared to the size of the irradiance distribution, a good estimation can be obtained. However, once this shift becomes too large, the quality by which the desired irradiance distribution can be estimated decreases with the distance between the sources.

We compared these results with various types of regularization. The maximum entropy regularization can come close to the solutions obtained by our proposed method. However, this approach requires careful selection of the regularization parameter to achieve optimal results. The conjugate gradient was able to converge to the known non-negative solution. However, for all other cases, it converges to a solution that is neither non-negative nor has finite support. Finally, the preconditioned gradient method with a discrete Laplace operator always converges to a solution that has negative values but due to the smoothness constraint imposed by the Laplace operator, it has finite support, but it extends beyond the domain on which the desired irradiance distribution is defined.

Future work should address two aspects. First, the issue of large matrices required to solve the problem will be tackled, enabling the analysis of higher-resolution irradiance and complex source distributions. Second, the theory should be extended to accommodate shift-variant impulse responses, which provide a more realistic representation of what is observed when moving a source in illumination systems.

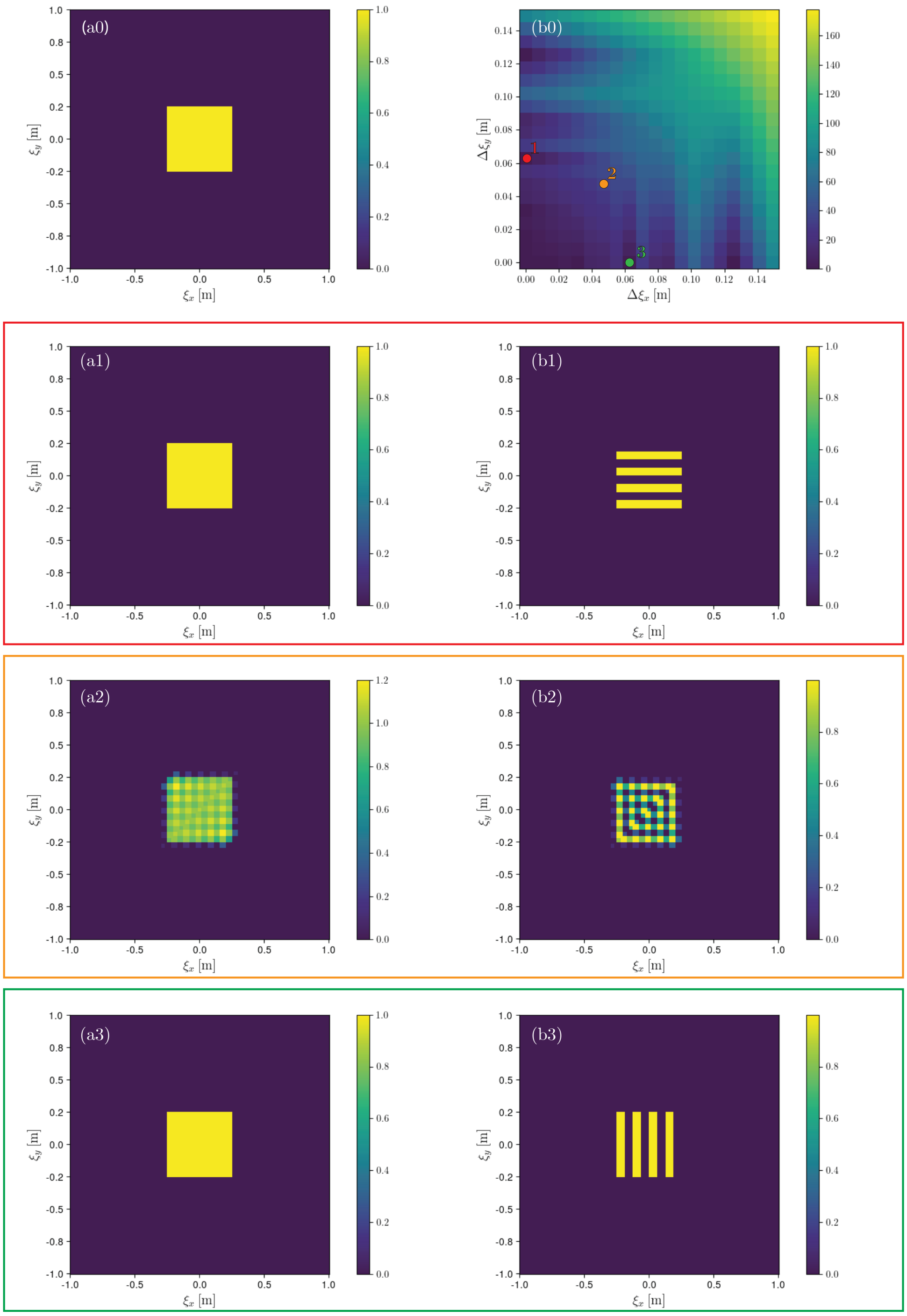


Figure 7.7: (a0) Desired irradiance distribution: square distribution of width 0.5; (b0) Loss landscape; (a1-b1) Total irradiance and impulse response obtained for $\boldsymbol{\Delta\xi} = (0, 8\Delta\xi_y)$ (red dot); (a2-b2) Total irradiance and impulse response obtained for $\boldsymbol{\Delta\xi} = (6\Delta\xi_x, 6\Delta\xi_y)$ (orange dot); (a3-b3) Total irradiance and impulse response obtained for $\boldsymbol{\Delta\xi} = (8\Delta\xi_x, 0)$ (green dot);

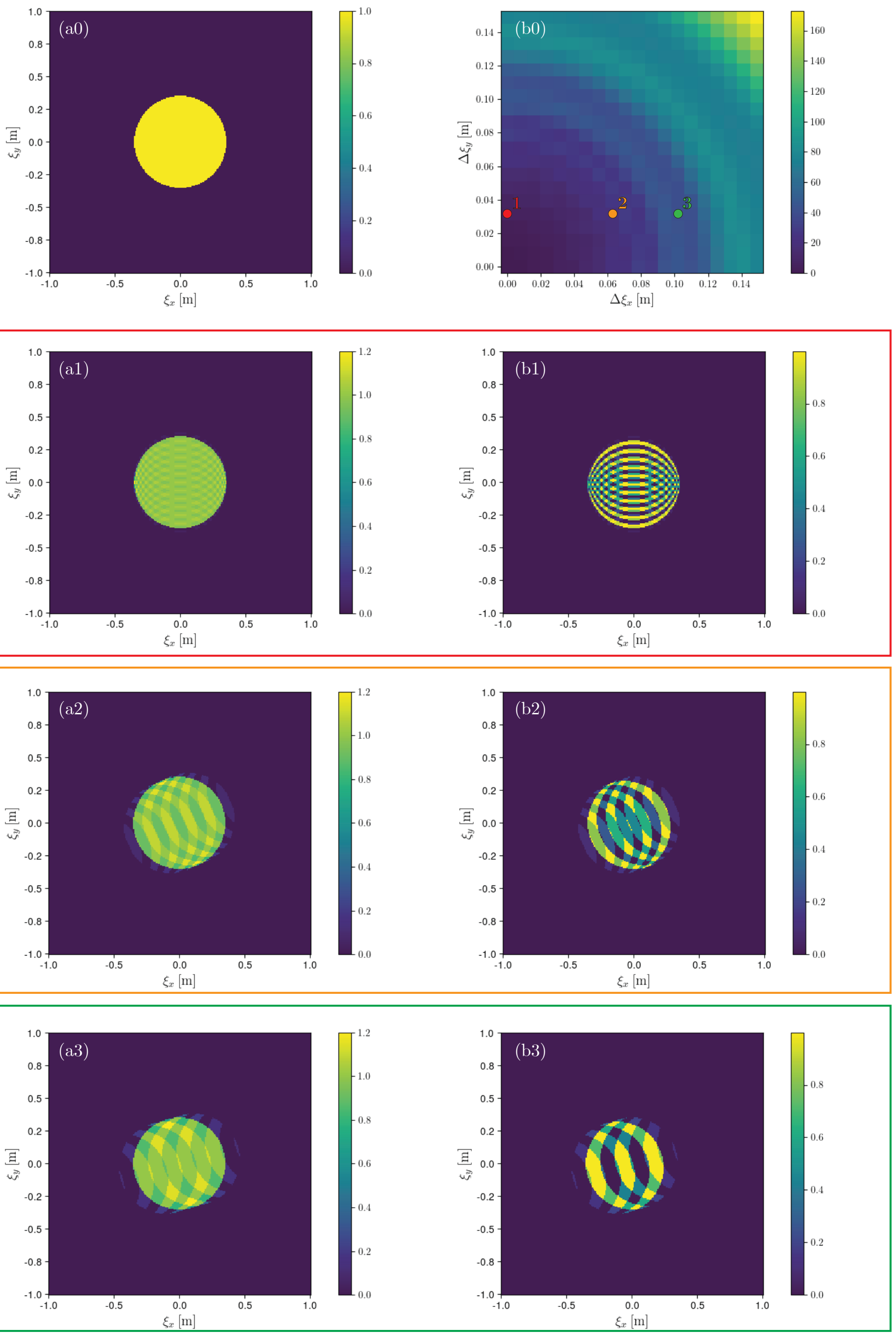


Figure 7.8: (a0) Desired irradiance distribution: uniform circle distribution with radius 0.33; (b0) Loss landscape; (a1-b1) Total irradiance and impulse response obtained for $\boldsymbol{\Delta\xi} = (0, 4\Delta\xi_y)$ (red dot); (a2-b2) Total irradiance and impulse response obtained for $\boldsymbol{\Delta\xi} = (8\Delta\xi_x, 4\Delta\xi_y)$ (orange dot); (a3-b3) Total irradiance and impulse response obtained for $\boldsymbol{\Delta\xi} = (13\Delta\xi_x, 4\Delta\xi_y)$ (green dot);

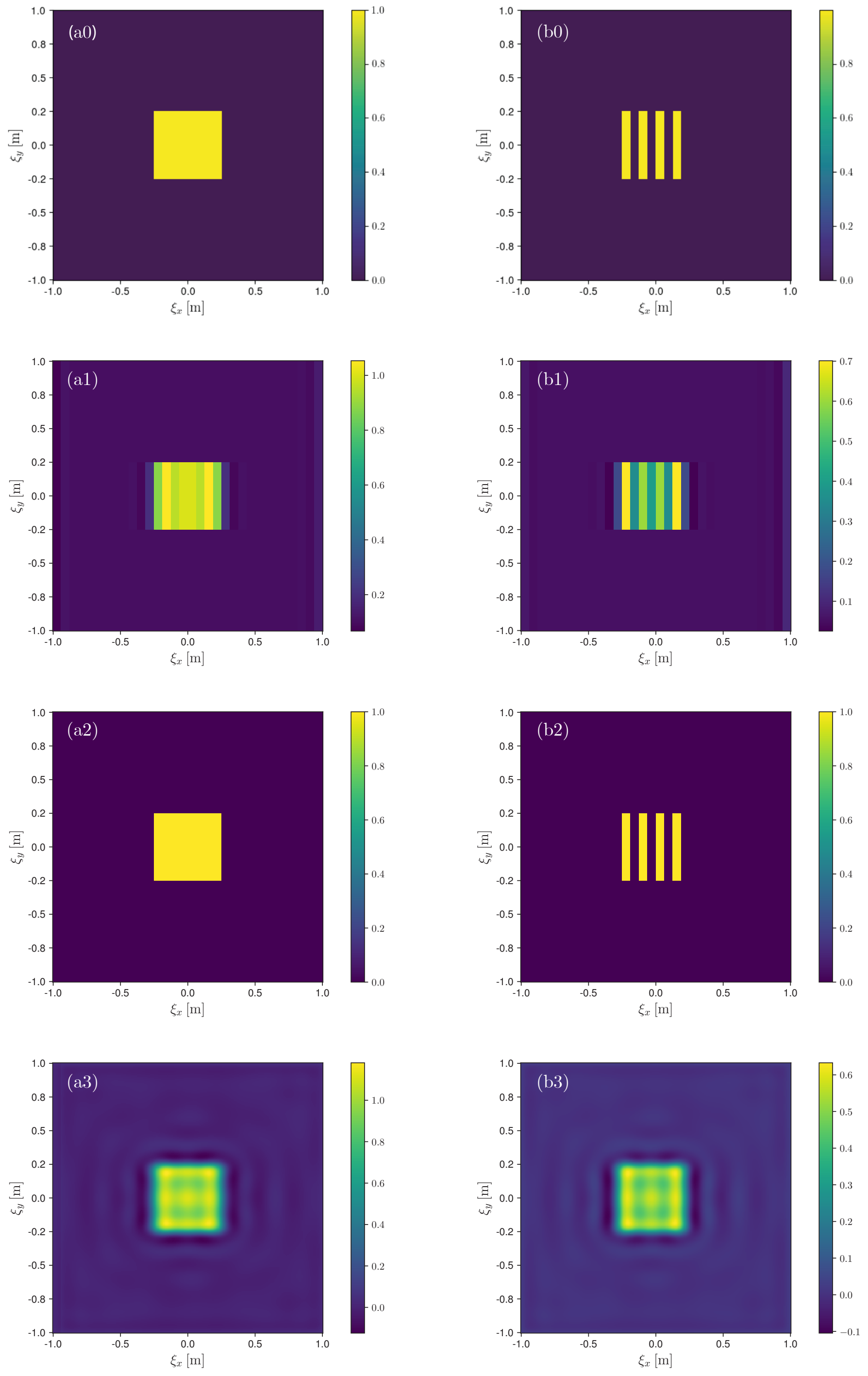


Figure 7.9: Comparison of regularized results and approximation using non-negative basis functions for the square distribution and for $\mathbf{\Delta\xi} = (8\Delta\xi_x, 0)$. Total irradiances (left) and impulse responses (right) obtained using: (a0-b0) Approximation using non-negative basis functions; (a1-b1) Maximum entropy regularization; (a2-b2) Conjugate gradient algorithm; (a3-b3) Preconditioned conjugate gradient algorithm and $\mathbf{L}$ the discrete Laplace operator;

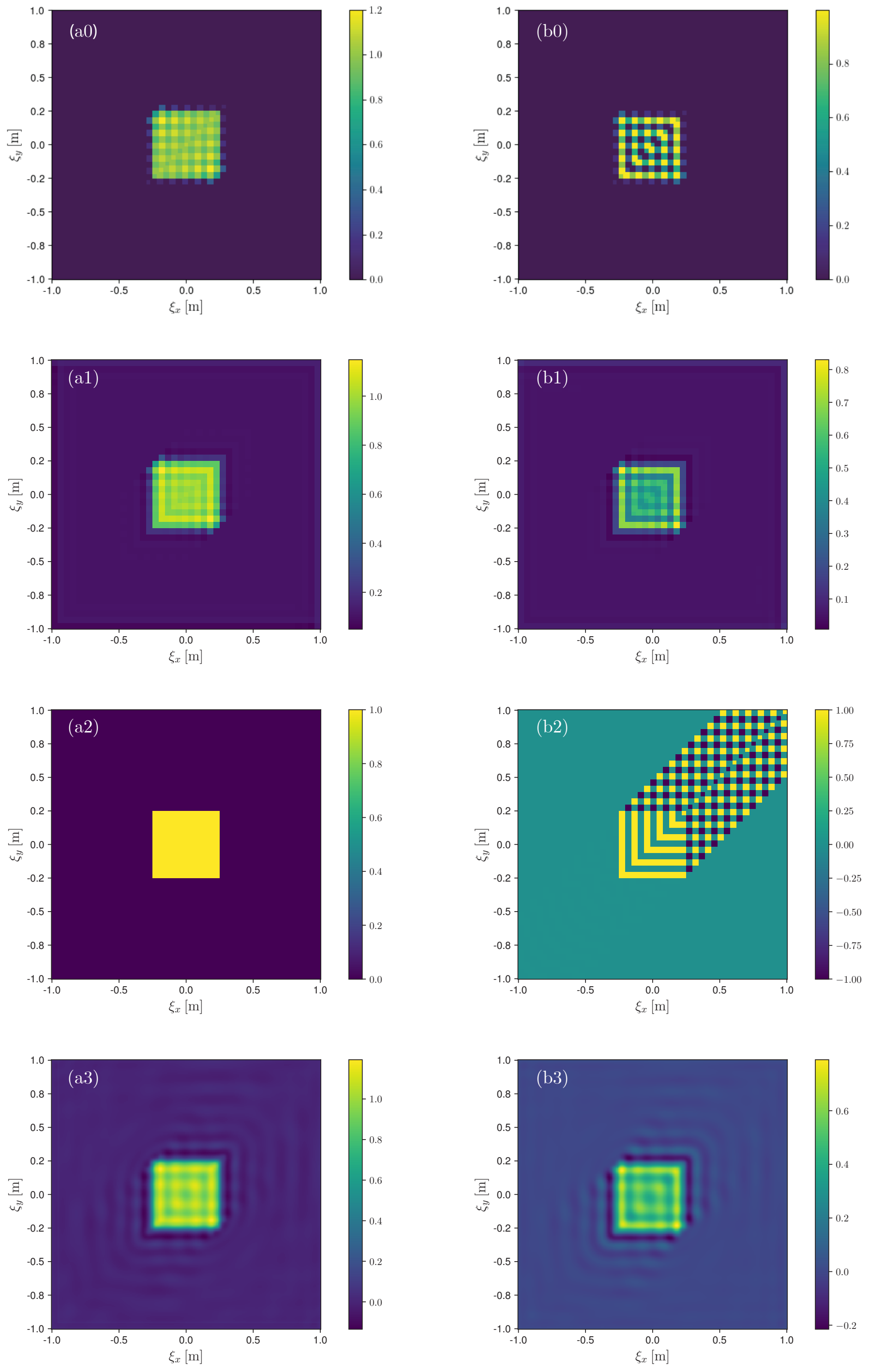


Figure 7.10: Comparison of regularized results and approximation using non-negative basis functions for the square distribution and for $\mathbf{\Delta\xi} = (6\Delta\xi_x, 6\Delta\xi_y)$. Total irradiances (left) and impulse responses (right) obtained using: (a0-b0) Approximation using non-negative basis functions; (a1-b1) Maximum entropy regularization; (a2-b2) Conjugate gradient algorithm; (a3-b3) Preconditioned conjugate gradient algorithm and $\mathbf{L}$ the discrete Laplace operator;

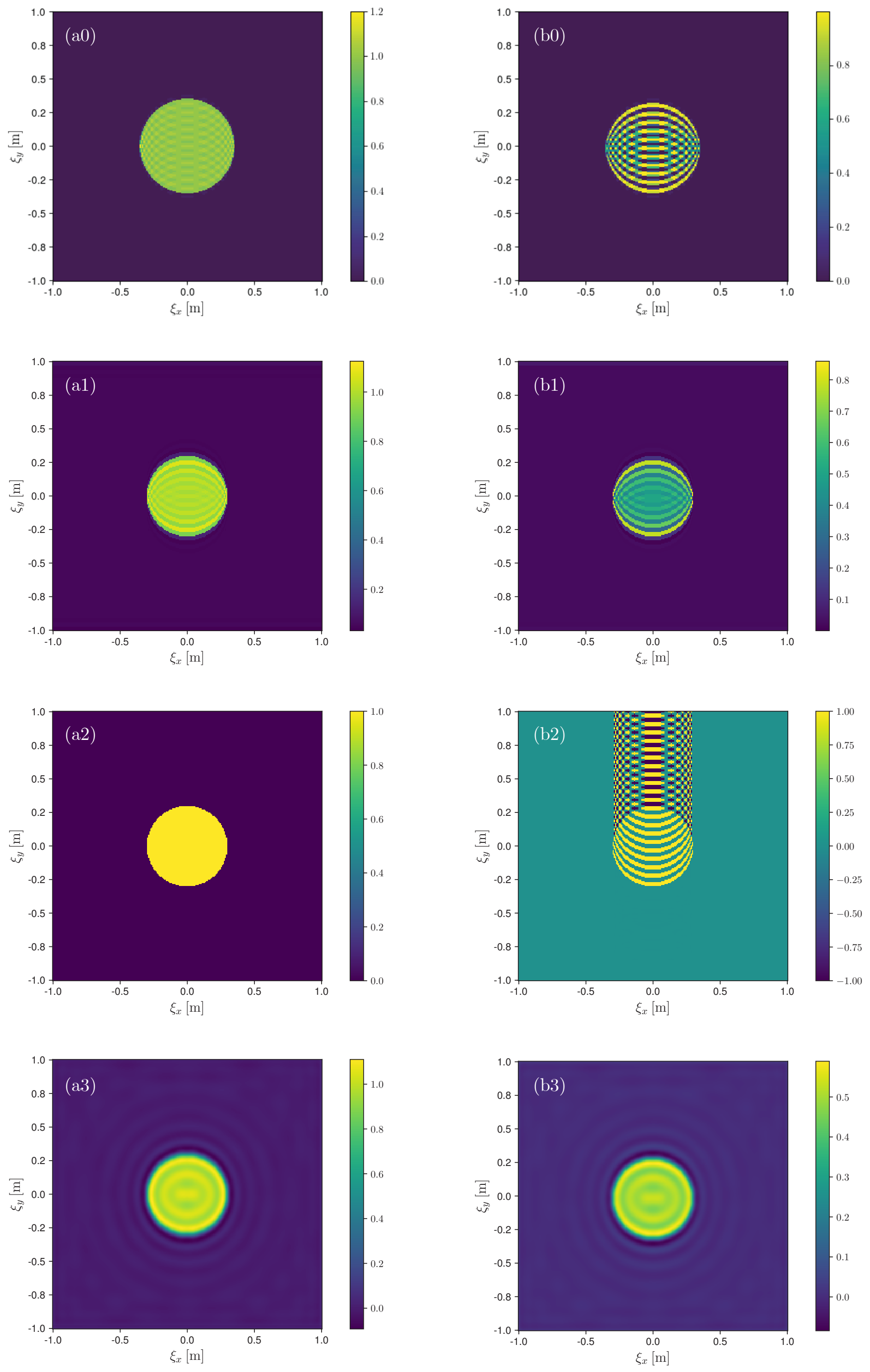


Figure 7.11: Comparison of regularized results and approximation using non-negative basis functions for the circular distribution and for $\boldsymbol{\Delta\xi} = (0\Delta\xi_x, 4\Delta\xi_y)$. Total irradiances (left) and impulse responses (right) obtained using: (a0-b0) Approximation using non-negative basis functions; (a1-b1) Maximum entropy regularization; (a2-b2) Conjugate gradient algorithm; (a3-b3) Preconditioned conjugate gradient algorithm and **L** the discrete Laplace operator;

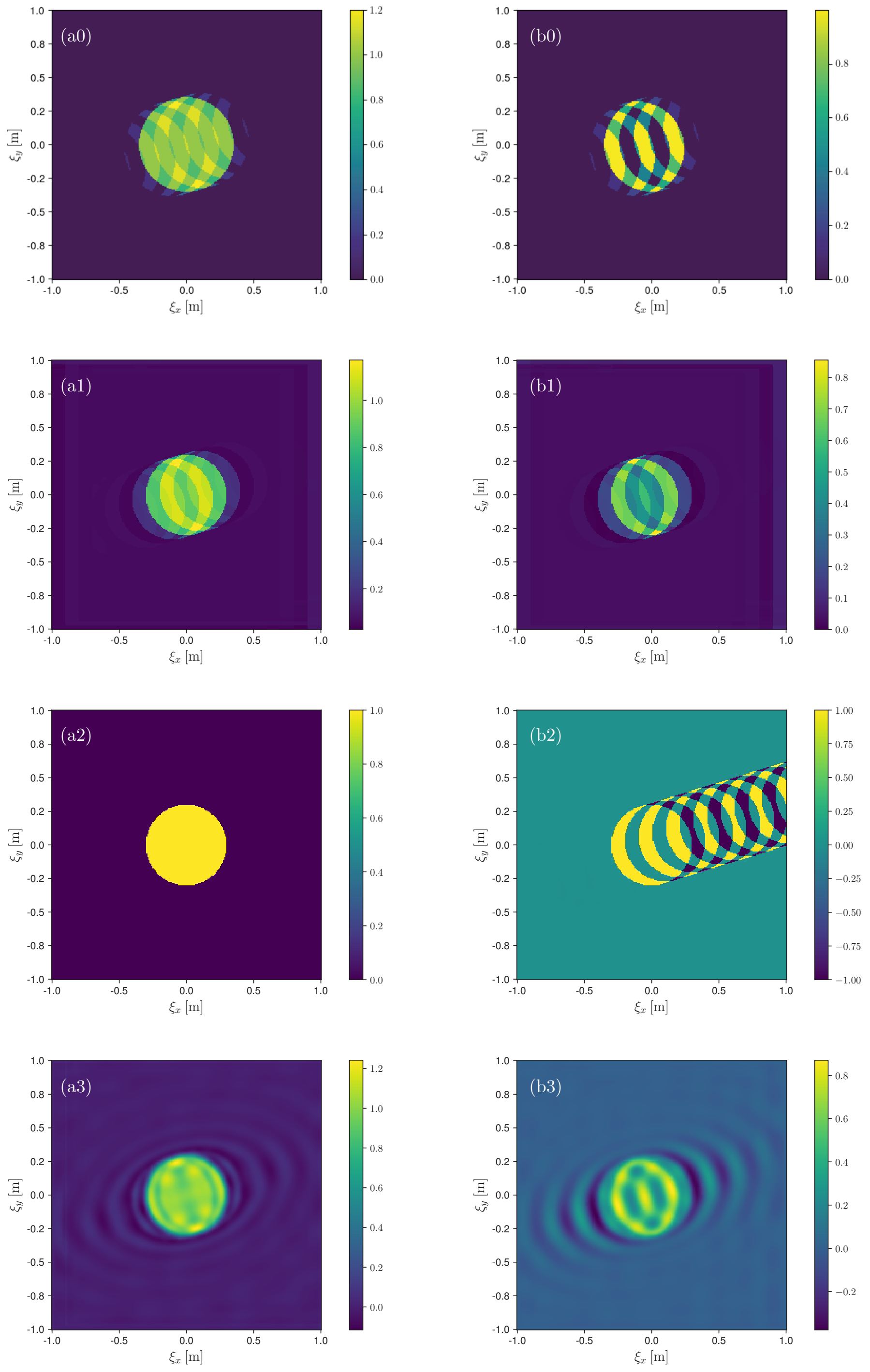


Figure 7.12: Comparison of regularized results and approximation using non-negative basis functions for the circular distribution and for $\boldsymbol{\Delta\xi} = (13\Delta\xi_x, 4\Delta\xi_y)$. Total irradiances (left) and impulse responses (right) obtained using: (a0-b0) Approximation using non-negative basis functions; (a1-b1) Maximum entropy regularization; (a2-b2) Conjugate gradient algorithm; (a3-b3) Preconditioned conjugate gradient algorithm and $\mathbf{L}$ the discrete Laplace operator;

# 8 CONCLUSION AND OUTLOOK

## 8.1. CONCLUSION

We began by showing that combining algorithmic differentiable ray tracing with B-splines and Truncated Hierarchical B-splines (THB-splines) provides a practical framework for the design of freeform lenses for illumination applications for both zero-étendue and finite étendue sources, and concluded by exploring the fundamental limits of which irradiance distributions are achievable for a given finite étendue source.

In chapter 5 we demonstrated that combining algorithmic differentiable ray tracing with B-splines works well for simple irradiance distributions which can be produced by freeform surfaces described by only a few control points. For more complex targets, more degrees of freedom are required and without properly tuning the optimization parameters, irregularities start to appear in the resulting freeform surfaces. These irregularities, in turn, increase stray light. Reducing the gradient descent step size can reduce the amount of irregularities. However, this comes at the cost of longer optimization times. Introducing a neural network to remap the optimization space can accelerate convergence, although its broader impact still needs investigation.

To address the trade-off between smoothness and convergence time, Chapter 6 introduced the use of knot insertion and THB-splines to gradually increase the number of degrees of freedom in the freeform surface. To leverage the THB-splines capability to locally refine the surface, we used a gradient-based criterion to pinpoint where the surface should be refined. This strategy shows that gradually increasing the degrees of freedom yields well-behaved freeform lenses without excessive optimization overhead.

Finally, in Chapter 7, we discuss the more fundamental question: can we generate any irradiance distribution using an extended source? By approximation an extended source by multiple spatially shifted sources and assume a shift invariant response of the system such that the irradiance distribution does not change with source position. We uncovered the underlying complexities which arise when trying to convert a finite-etendue problem into a zero-etendue problem. We identify two seemingly trivial cases for this problem. The first is to use a single source and to design a system to create the target distribution. The second is to shape the source and image the source onto the target plane. The complexity arises when not all information can be embedded exclusively in the system or exclusively in the source. Unless the source is specifically designed for the target distribution, the only practical path is approximation, subject to a non-negativity constraint.

## 8.2. OUTLOOK

Plenty remains to be explored. So far, the results are confined to single-surface freeform lenses for an extended source.

First, the neural network is currently used only to accelerate optimization. With sufficient training data, it could generate an initial freeform lens for a given target distribution, and transfer learning could leverage prior optimizations to speed up future runs. Second, beyond the finite-étendue problem, existing approaches are being extended to include multiple optical surfaces [4], Fresnel losses [87], surface scattering [51], and spatially varying refractive index [54]. Combining several of these effects remains challenging. The method developed here could incorporate them with only moderate changes, but the added degrees of freedom and complexity will require smarter optimization strategies, perhaps using higher-dimensional splines for GRIN media. Finally, to better understand which target distributions are achievable with a finite-étendue source, the shift-invariance analysis should be generalized to shift-variant source responses.

# BIBLIOGRAPHY

[1] Aydan Aksoylar. “Modeling and model-aware signal processing methods for enhancement of optical systems”. PhD thesis. Boston University, 2016.

[2] U Amato and W Hughes. “Maximum entropy regularization of Fredholm integral equations of the first kind”. In: *Inverse problems* 7.6 (1991), p. 793.

[3] Ansys. *Zemax*. https://www.zemax.com/. 2023.

[4] Martijn JH Anthonissen et al. “Unified mathematical framework for a class of fundamental freeform optical systems”. In: *Optics Express* 29.20 (2021), pp. 31650–31664.

[5] GR Ayers and J Christopher Dainty. “Iterative blind deconvolution method and its applications”. In: *Optics letters* 13.7 (1988), pp. 547–549.

[6] Edward Bailey and Sebastien Carayon. “Beyond NURBS: enhancement of local refinement through T-splines”. In: *Nonimaging Optics and Efficient Illumination Systems IV*. Vol. 6670. SPIE. 2007, pp. 151–160.

[7] Agamirza Bashirov. “Mathematical analysis fundamentals”. In: Academic Press, 2014.

[8] Pablo Benitez et al. “SMS design method in 3D geometry: examples and applications”. In: *Nonimaging Optics: Maximum Efficiency Light Transfer VII*. Vol. 5185. SPIE. 2004, pp. 18–29.

[9] Max Born and Emil Wolf. *Principles of optics: electromagnetic theory of propagation, interference and diffraction of light*. Elsevier, 2013.

[10] Peter Borwein et al. “On the zeros of cosine polynomials: solution to a problem of Littlewood”. In: *Annals of mathematics* (2008), pp. 1109–1117.

[11] Christoph Bösel and Herbert Gross. “Double freeform illumination design for prescribed wavefronts and irradiances”. In: *JOSA A* 35.2 (2018), pp. 236–243.

[12] Christoph Bösel and Herbert Gross. “Ray mapping approach for the efficient design of continuous freeform surfaces”. In: *Optics express* 24.13 (2016), pp. 14271–14282.

[13] Alan C Bovik. “The essential guide to image processing”. In: Academic Press, 2009.

[14] James Bradbury et al. *JAX: composable transformations of Python+NumPy programs*. Version 0.3.13. 2018. URL: `http://github.com/google/jax`.

[15] Matt Brand. “Minimum spot size and maximum detail in extended-source freeform illumination”. In: *Flat Optics: Components to Systems*. Optica Publishing Group. 2021, JTh1A–1.

[16] Matt Brand and Daniel A Birch. “Freeform irradiance tailoring for light fields”. In: *Optics express* 27.12 (2019), A611–A619.

[17] Matthew Brand and Aydan Aksoylar. “Sharp images from freeform optics and extended light sources”. In: *Frontiers in Optics 2016*. Optica Publishing Group, 2016, FW5H.2.

[18] A Brömel et al. “Performance comparison of polynomial representations for optimizing optical freeform systems”. In: *Optical Systems Design 2015: Optical Design and Engineering VI*. Vol. 9626. SPIE. 2015, pp. 217–228.

[19] Egor V Byzov et al. “Optimization method for designing double-surface refractive optical elements for an extended light source”. In: *Optics Express* 28.17 (2020), pp. 24431–24443.

[20] Patrizio Campisi and Karen Egiazarian. *Blind image deconvolution: theory and applications*. CRC press, 2017.

[21] Alfred S Carasso. “False characteristic functions and other pathologies in variational blind deconvolution. A method of recovery”. In: *SIAM Journal on Applied Mathematics* 70.4 (2009), pp. 1097–1119.

[22] Samji Isaac Chandra and Annie Shalom. *Intelligent Freeform Deformation for LED Illumination Optics*. Vol. 16. KIT Scientific Publishing, 2018.

[23] Julio Chaves. *Introduction to nonimaging optics*. CRC press, 2008.

[24] E Cohen et al. “Analysis-aware modeling: Understanding quality considerations in modeling for isogeometric analysis”. In: *Computer Methods in Applied Mechanics and Engineering* 199.5-8 (2010), pp. 334–356.

[25] Karel Desnijder, Peter Hanselaer, and Youri Meuret. “Ray mapping method for off-axis and non-paraxial freeform illumination lens design”. In: *Optics letters* 44.4 (2019), pp. 771–774.

[26] Tor Dokken, Tom Lyche, and Kjell Fredrik Pettersen. “Polynomial splines over locally refined box-partitions”. In: *Computer Aided Geometric Design* 30.3 (2013), pp. 331–356.

[27] Leonid L Doskolovich et al. “Designing double freeform surfaces for collimated beam shaping with optimal mass transportation and linear assignment problems”. In: *Optics Express* 26.19 (2018), pp. 24602–24613.

[28] Oliver Dross et al. “Review of SMS design methods and real world applications”. In: *Nonimaging optics and efficient illumination systems*. Vol. 5529. SPIE. 2004, pp. 35–47.

[29] John Duchi, Elad Hazan, and Yoram Singer. “Adaptive subgradient methods for online learning and stochastic optimization.” In: *Journal of machine learning research* 12.7 (2011).

[30] Gerald Farin. *Curves and Surfaces for CAGD*. 5th. Burlington, Massachusetts: Morgan Kaufmann Publishers, 2002.

[31] Gregory E Fasshauer. *Meshfree approximation methods with Matlab*. Vol. 6. World Scientific Publishing Company, 2007.

[32] Zexin Feng et al. “Designing double freeform optical surfaces for controlling both irradiance and wavefront”. In: *Optics express* 21.23 (2013), pp. 28693–28701.

[33] DA Fish et al. “Blind deconvolution by means of the Richardson–Lucy algorithm”. In: *Journal of the Optical Society of America A* 12.1 (1995), pp. 58–65.

[34] David R Forsey and Richard H Bartels. “Hierarchical B-spline refinement”. In: *Proceedings of the 15th annual conference on Computer graphics and interactive techniques*. 1988, pp. 205–212.

[35] Wilhelm Forst and Dieter Hoffmann. *Optimization—theory and practice*. Springer Science & Business Media, 2010.

[36] Florian R Fournier, William J Cassarly, and Jannick P Rolland. “Designing freeform reflectors for extended sources”. In: *Nonimaging optics: efficient design for illumination and solar concentration VI*. Vol. 7423. SPIE. 2009, pp. 11–22.

[37] Florian R Fournier, William J Cassarly, and Jannick P Rolland. “Fast freeform reflector generation using source-target maps”. In: *Optics Express* 18.5 (2010), pp. 5295–5304.

[38] Carlotta Giannelli, Bert Jüttler, and Hendrik Speleers. “THB-splines: The truncated basis for hierarchical splines”. In: *Computer Aided Geometric Design* 29.7 (2012), pp. 485–498.

[39] Carlotta Giannelli et al. “THB-splines: An effective mathematical technology for adaptive refinement in geometric design and isogeometric analysis”. In: *Computer Methods in Applied Mechanics and Engineering* 299 (2016), pp. 337–365.

[40] Barbara G. Grant. *Field guide to radiometry*. Bellingham, Wash: SPIE, 2011, p. 110.

[41] Andreas Griewank and Andrea Walther. *Evaluating derivatives: principles and techniques of algorithmic differentiation*. SIAM, 2008.

[42] Per Christian Hansen. “Regularization tools: A Matlab package for analysis and solution of discrete ill-posed problems”. In: *Numerical algorithms* 6.1 (1994), pp. 1–35.

[43] Eugene Hecht. *Optics*. Pearson Education India, 2012.

[44] Alexander NM Heemels, Aurèle JL Adam, and H Paul Urbach. “Limits of realizing irradiance distributions with shift-invariant illumination systems and finite étendue sources”. In: *JOSA A* 40.7 (2023), pp. 1289–1302.

[45] Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. *Overview of mini-batch gradient descent*. 2012.

[46] Shixiong Hu et al. “Ultra-compact LED lens with double freeform surfaces for uniform illumination”. In: *Optics express* 23.16 (2015), pp. 20350–20355.

[47] Wenzel Jakob et al. “DR. JIT: a just-in-time compiler for differentiable rendering”. In: *ACM Transactions on Graphics (TOG)* 41.4 (2022), pp. 1–19.

[48] Norman L Johnson, Samuel Kotz, and Narayanaswamy Balakrishnan. *Continuous univariate distributions, volume 2*. Vol. 2. John wiley & sons, 1995.

[49] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).

[50] Bart de Koning et al. “Gradient descent-based freeform optics design for illumination using algorithmic differentiable non-sequential ray tracing”. In: *Optimization and Engineering* (2023), pp. 1–33.

[51] Vı CE Kronberg et al. “Two-dimensional freeform reflector design with a scattering surface”. In: *JOSA A* 40.4 (2023), pp. 661–675.

[52] Linpei Li and Xiang Hao. “Optimizing triangle mesh lenses for non-uniform illumination with an extended source”. In: *Optics Letters* 48.7 (2023), pp. 1726–1729.

[53] Zongtao Li et al. “Energy feedback freeform lenses for uniform illumination of extended light source LEDs”. In: *Applied Optics* 55.36 (2016), pp. 10375–10381.

[54] David H. Lippman and Greg R. Schmidt. “Prescribed irradiance distributions with freeform gradient-index optics”. In: *Opt. Express* 28 (2020), pp. 29132–29147.

[55] ltioptics. *Photopia*. https://www.ltioptics.com/en/. 2023.

[56] Leon B Lucy. “An iterative technique for the rectification of observed distributions”. In: *Astronomical Journal, Vol. 79, p. 745 (1974)* 79 (1974), p. 745.

[57] Yi Luo et al. “Design of compact and smooth free-form optical system with uniform illuminance for LED source”. In: *Optics express* 18.9 (2010), pp. 9055–9063.

[58] Donglin Ma, Zexin Feng, and Rongguang Liang. “Deconvolution method in designing freeform lens array for structured light illumination”. In: *Applied optics* 54.5 (2015), pp. 1114–1117.

[59] Alejandro Madrid-Sánchez et al. “Freeform optics design method for illumination and laser beam shaping enabled by least squares and surface optimization”. In: *Optik* 269 (2022), p. 169941.

[60] Maplesoft. *Maple*. 2023. URL: `https://www.maplesoft.com/products/Maple/`.

[61] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: `https://www.tensorflow.org/`.

[62] *MITSUBA 3: Caustics optimization tutorial*. `https://mitsuba.readthedocs.io/en/stable/src/inverse_rendering/caustics_optimization.html`. Accessed: 2023-08-18.

[63] Julius Muschaweck. “Tailored freeform surfaces for illumination with extended sources”. In: *Nonimaging Optics: Efficient Design for Illumination and Solar Concentration XVIII*. Vol. 12220. SPIE. 2022, pp. 8–16.

[64] Julius Muschaweck and Henning Rehn. *Designing Illumination Optics*. SPIE Press, 2022.

[65] Yunfeng Nie et al. “Freeform optical system design with differentiable three-dimensional ray tracing and unsupervised learning”. In: *Optics Express* 31.5 (2023), pp. 7450–7465.

[66] Merlin Nimier-David et al. “Mitsuba 2: A retargetable forward and inverse renderer”. In: *ACM Transactions on Graphics (ToG)* 38.6 (2019), pp. 1–17.

[67] Jorge Nocedal and Stephen J Wright. *Numerical optimization*. Springer, 2006.

[68] Vladimir Oliker, Jacob Rubinstein, and Gershon Wolansky. “Supporting quadric method in optical design of freeform lenses for illumination control of a collimated light”. In: *Advances in Applied Mathematics* 62 (2015), pp. 160–183.

[69] Adam Paszke et al. “PyTorch: An Imperative Style, High-Performance Deep Learning Library”. In: *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035.

[70] Matt Pharr, Wenzel Jakob, and Greg Humphreys. *Physically based rendering: From theory to implementation*. Burlington, Massachusetts: Morgan Kaufmann Publishers, 2016.

[71] Matt Pharr, Wenzel Jakob, and Greg Humphreys. *Physically based rendering: From theory to implementation*. MIT Press, 2023.

[72] Les Piegl and Wayne Tiller. *The NURBS book*. Springer Science & Business Media, 1996.

[73] CR Prins et al. “A least-squares method for optimal transport using the Monge–Ampère equation”. In: *SIAM Journal on Scientific Computing* 37.6 (2015), B937–B961.

[74] William Hadley Richardson. “Bayesian-based iterative method of image restoration”. In: *JoSA* 62.1 (1972), pp. 55–59.

[75] Melissa N Ricketts, Roland Winston, and Vladimir Oliker. “Diffraction effects in freeform optics”. In: *Nonimaging Optics: Efficient Design for Illumination and Solar Concentration XII*. Vol. 9572. SPIE. 2015, pp. 126–131.

[76] Harald Ries and Julius Muschaweck. “Tailored freeform optical surfaces”. In: *Journal of the Optical Society of America A* 19.3 (2002), pp. 590–595.

[77] Lotte B Romijn, Jan HM ten Thije Boonkkamp, and Wilbert L IJzerman. “Freeform lens design for a point source and far-field target”. In: *JOSA A* 36.11 (2019), pp. 1926–1939.

[78] Julian Sahasrabudhe. “Counting zeros of cosine polynomials: on a problem of Littlewood”. In: *Advances in Mathematics* 343 (2019), pp. 495–521.

[79] Thomas W Sederberg et al. “T-splines and T-NURCCs”. In: *ACM transactions on graphics (TOG)* 22.3 (2003), pp. 477–484.

[80] C Ray Smith and Walter T Grandy Jr. *Maximum-Entropy and bayesian methods in inverse problems*. Vol. 14. Springer Science & Business Media, 2013.

[81] Simone Sorgato et al. “Design of illumination optics with extended sources based on wavefront tailoring”. In: *Optica* 6.8 (2019), pp. 966–971.

[82] Qilin Sun et al. “End-to-end complex lens design with differentiable ray tracing”. In: *ACM Trans. Graph* 40.4 (2021), pp. 1–13.

[83] Synopsys. *Code V*. https://www.synopsys.com/optical-solutions/codev.html. 2023.

[84] Synopsys. *LightTools*. https://www.synopsys.com/optical-solutions/lighttools.html. 2023.

[85] Derek C Thomas et al. “U-splines: Splines over unstructured meshes”. In: *Computer Methods in Applied Mechanics and Engineering* 401 (2022), p. 115515.

[86] Andrey Nikolayevich Tikhonov. *Solutions of ill posed problems*. John Wiley & Sons, 1977.

[87] AH Van Roosmalen et al. “Fresnel reflections in inverse freeform lens design”. In: *JOSA A* 39.6 (2022), pp. 1045–1052.

[88] Jean-Baptiste Volatier, Álvaro Menduiña-Fernández, and Markus Erhard. “Generalization of differential ray tracing by automatic differentiation of computational graphs”. In: *Journal of the Optical Society of America A* 34.7 (2017), pp. 1146–1151.

[89] Congli Wang, Ni Chen, and Wolfgang Heidrich. “dO: A differentiable engine for Deep Lens design of computational imaging systems”. In: *IEEE Transactions on Computational Imaging* 8 (2022), pp. 905–916.

[90] Haoqiang Wang et al. “Extended ray-mapping method based on differentiable ray-tracing for non-paraxial and off-axis freeform illumination lens design”. In: *Optics Express* 31.19 (2023), pp. 30066–30078.

[91] Shili Wei, Zhengbo Zhu, and Donglin Ma. “Efficient and compact freeform optics design for customized LED lighting”. In: *Optics & Laser Technology* 167 (2023), p. 109775.

[92] Shili Wei et al. “Compact freeform illumination optics design by deblurring the response of extended sources”. In: *Optics Letters* 46.11 (2021), pp. 2770–2773.

[93] Rolf Wester et al. “Designing optical free-form surfaces for extended sources”. In: *Optics express* 22.102 (2014), A552–A560.

[94] Roland Winston, Juan C Miñano, Pablo G Benitez, et al. *Nonimaging optics*. Elsevier, 2005.

[95] Wolfram. *Mathematica*. 2023. URL: `https://www.wolfram.com/mathematica/`.

[96] Rengmao Wu, José Sasián, and Rongguang Liang. “Algorithm for designing freeform imaging optics with nonrational B-spline surfaces”. In: *Applied Optics* 56.9 (2017), pp. 2517–2522.

[97] Rengmao Wu et al. “A mathematical model of the single freeform surface design for collimated beam shaping”. In: *Optics express* 21.18 (2013), pp. 20974–20989.

[98] Rengmao Wu et al. “Design of freeform illumination optics”. In: *Laser & Photonics Reviews* 12.7 (2018), p. 1700310.

[99] Rengmao Wu et al. “Freeform illumination design: a nonlinear boundary problem for the elliptic Monge–Ampére equation”. In: *Optics letters* 38.2 (2013), pp. 229–231.

[100] Nitin K Yadav, JHM ten Thije Boonkkamp, and WL IJzerman. “Computation of double freeform optical surfaces using a Monge–Ampère solver: Application to beam shaping”. In: *Optics Communications* 439 (2019), pp. 251–259.

[101] Zhengbo Zhu et al. “Freeform illumination optics design for extended LED sources through a localized surface control method”. In: *Optics Express* 30.7 (2022), pp. 11524–11535.

[102] Susanne Zwick et al. “Resolution limitations for tailored picture-generating freeform surfaces”. In: *Optics Express* 20.4 (2012), pp. 3642–3653.

# ACKNOWLEDGMENTS

When I started my bachelor's in Industrial Design Engineering in Delft back in 2013, I never expected to pursue a PhD in optics. But somehow, during my time in Delft I made a combination of choices that led me here. This feat wouldn't have been possible without the help of a lot of people, whom I want to thank.

First, I want to thank the people who introduced me to the field of optics. I initially chose to do a master's in Systems and Control because of my fascination with synthesizers and signal processing. Along the way, I learned that optics included many of the topics that drew me in in the first place. These insights were made possible by courses taught by Gleb Vdovine, Oleg Soloviev, Michel Verhagen, Paul Urbach, and Tope Agbana. Thank you for introducing me to the subject.

I want to thank Aurèle for his supervision. It was a fascinating journey, where we both started out knowing very little about illumination optics and which questions to pursue. But together we found our way. This was in part thanks to the freedom you gave me to explore different research questions, and to your amazing ability to recruit students so we could explore topics we otherwise would not have had time for.

The years within the optics group have been memorable. I always enjoyed the coffee breaks with Thomas van der Sijs, Thomas Kotte, Sven Weerdenburg, Dmytro Kolenov, Thim Zuidwijk, and Roland Horsten. Those coffee breaks helped break up the long days of programming, writing, and reading. Your company and conversations made a real difference, especially during the pandemic.

The research would not have been the same without Matthias Möller, with whom we started exploring the use of B-splines and THB-splines in optical design problems. The initial idea of solving everything with machine learning did not succeed, but it showed us how useful algorithmic differentiation (AD) is in these problems. This work came to a strong conclusion with the help of Bart de Koning, who was essential in combining the AD and THB-spline frameworks and had a major impact on the success of this thesis.

A special thanks to Stefan Baumer, who helped me focus the research on more relevant topics. I am grateful for your honesty and that you took the time to help me see that I was getting lost in irrelevant questions. Your feedback was difficult to process at the time, but I am happy with how everything turned out, and it is thanks to you that I can now say I am proud of what I have achieved.

There are also a lot of people who have supported me outside this world of optics. I owe thanks to my father Jan-Pieter and mother Sandra, and my siblings Tristan, Robin, and Niki, for their support and encouragement. The same goes for Lili and Arjan. The vacations in Spain were very necessary breaks from the research environment.

An essential way to decompress was playing music. It helped me forget about the stress that comes with a PhD. To all the people with whom I make music: Stijn, Naomi, Maud, Guy, Liset, Janneke, Matthias, Emilie, Daniel, Alo, Gijs, and Annelot. Thank you for all the fun during rehearsals and performances, and for being an amazing group of friends.

To my roommate, Tim Boot: I look back fondly on the time we commuted to work together, had dinner almost every day, and were able to support each other with PhD-related problems.

Finally, I want to thank my love, Amanda, for her support and calming words all those times when I was stressed, and for always motivating me to be the best version of myself.

To everyone mentioned, and the many I have missed, thank you. This achievement would not have been possible without you.

# CURRICULUM VITÆ

## Alexander Nicholas Michael HEEMELS

| | |
|---|---|
| 03-09-1994 | Born in Robbinsdale, Minnesota, United States of America. |

## WORK

| | |
|---|---|
| 2025–Present | Optical Engineer, Demcon |
| 2023–2025 | Optical Engineer, Delta Life Science |

## EDUCATION

| | | |
|---|---|---|
| 2019–2023 | PhD. Applied Physics<br>Technische Universiteit Delft<br>Faculty of Applied Sciences | |
| | *Thesis:* | Composing Light |
| | *Promotor:* | Prof. dr. H. Urbach |
| | *Promotor:* | Dr. M. Möller |
| | *Co-Promotor:* | Dr. A. Adam |
| 2017–2019 | Masters Systems and Control<br>Technische Universiteit Delft<br>Mechanical, Maritime and Materials Engineering | |
| 2016–2017 | Bridging Program Systems and Control<br>Technische Universiteit Delft<br>Mechanical, Maritime and Materials Engineering | |
| 2013–2016 | Bachelors Industrial Design Engineering<br>Technische Universiteit Delft<br>Faculty of Industrial Design Engineering | |

# List of Publications

## Journal Publications

3. **A. Heemels**, B. de Koning, M. Möller, A. Adam, *Optimizing freeform lenses for extended sources with algorithmic differentiable ray tracing and truncated hierarchical B-splines*, Opt. Express 32, 9730-9746 (2024).

2. B. de Koning, **A. Heemels**, A. Adam, M. Möller *Gradient descent-based freeform optics design for illumination using algorithmic differentiable non-sequential ray tracing*, Optim Eng, (2023).

1. **A. Heemels**, A. Adam, H. Urbach, *Limits of realizing irradiance distributions with shift-invariant illumination systems and finite étendue sources*, J. Opt. Soc. Am. A **40**, 1289-1302 (2023).

## Conference Contributions

8. **A. Heemels**, B. de Koning, M. Möller, A. Adam, *A novel tools for designing freeform optics*, Photonics Day (2023).

7. **A. Heemels**, B. de Koning, M. Möller, A. Adam, *Unsupervised design of illumination optics using algorithmic differentiable non-sequential raytracing*, International Optical Design Conference (2023).

6. **A. Heemels**, A. Adam, H. Urbach, *Limitations of generating prescribed irradiance distribution with finite étendue sources*, International Optical Design Conference (2023).

5. **A. Heemels**, A. Adam, H. Urbach, *Irradiance Tailoring with multiple sources using B-spline refinement*, European Optical Society Annual Meeting (2022).

4. **A. Heemels**, L. Pels, A. Adam, H. Urbach, *Influence of source translation on caustics generated by freeform surfaces*, NWO Physics (2022).

3. **A. Heemels**, A. Adam, H. Urbach, *Multi-source light shaping with diffractive optical elements*, European Optical Society Annual Meeting (2021).

2. **A. Heemels**, T. Agbana, S. Pereira, J. Diehl, M. Verhagen, G. Vdovine, *Effect of partial coherent illumination on Fourier ptychography*, Label-free Biomedical Imaging and Sensing (LBIS) **11251**, 112-118 (2020).

1. **A. Heemels**, T. Agbana, S. Pereira, J. Diehl, M. Verhagen, G. Vdovine, *Effect of partial coherent illumination on Fourier ptychography*, Face2Phase (2019).