End-to-end Surface Optimization for Light Control

YUOU SUN, University of Science and Technology of China, China
BAILIN DENG, Cardiff University, United Kingdom
JUYONG ZHANG*, University of Science and Technology of China, China

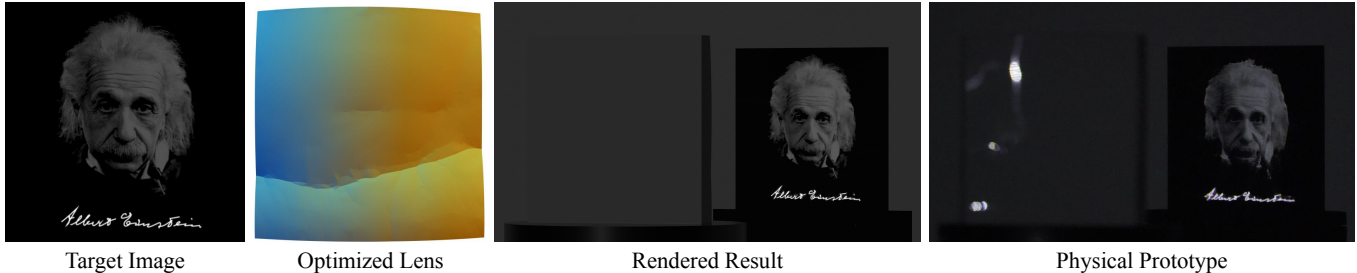| Target Image | Optimized Lens | Rendered Result | Physical Prototype |

Fig. 1. A lens shape designed by our method according to a target image. The light pattern produced by the lens closely resembles the target image, as shown by both the simulated rendering and the physical prototype. Einstein portrait © Magnum/IC photo.

Designing a freeform surface to reflect or refract light to achieve a target distribution is a challenging inverse problem. In this paper, we propose an end-to-end optimization strategy for an optical surface mesh. Our formulation leverages a novel differentiable rendering model, and is directly driven by the difference between the resulting light distribution and the target distribution. We also enforce geometric constraints related to fabrication requirements, to facilitate CNC milling and polishing of the designed surface. To address the issue of local minima, we formulate a face-based optimal transport problem between the current mesh and the target distribution, which makes effective large changes to the surface shape. The combination of our optimal transport update and rendering-guided optimization produces an optical surface design with a resulting image closely resembling the target, while the geometric constraints in our optimization help to ensure consistency between the rendering model and the final physical results. The effectiveness of our algorithm is demonstrated on a variety of target images using both simulated rendering and physical prototypes.

CCS Concepts: • **Computing methodologies** → **Shape modeling**; *Rasterization*; • **Theory of computation** → *Mathematical optimization.*

Additional Key Words and Phrases: Differentiable rendering, Computational design, End-to-end optimization, Inverse surface design

1 INTRODUCTION

Light plays an important role in our daily life, and people have been learning how to manipulate it for thousands of years. In this paper, we investigate the inverse design of free-form optical surfaces that can exactly transport the light emitted by a source onto a target pattern. It is an important problem in non-imaging optics, and has many applications where one wants to precisely control the direction and density of light rays such as in the field of art, architecture, medical equipment, and energy harvesting.

This problem has been studied in different fields including computer graphics [Meyron et al. 2018; Schwartzburg et al. 2014; Yue

et al. 2014], computational photography [Kingslake 1989; Sun et al. 2021], and geometrical optics [Budhu and Rahmat-Samii 2019; Šarbort and Tyc 2012]. In computer graphics, existing methods typically first determine certain auxiliary features of the desired optical surface shape, such as its normals [Schwartzburg et al. 2014] or visibility diagram [Meyron et al. 2018]. These features are then used to reconstruct the final surface. However, such approaches may fail to produce high-precision final results due to two factors. Firstly, these methods do not directly consider the difference between the resulting distribution and the target; consequently, errors introduced in feature computation and surface reconstruction may cause notable deviations between the resulting light pattern and the target. Additionally, these methods mainly aim to bring the simulation result close to the target light distribution, without considering the manufacturability of the designed surface.

To overcome these limitations and achieve desirable results in the physically fabricated final surface, we propose a method directly driven by the difference between the resulting pattern and the target pattern, with a rendering model consistent with the light behavior. Specifically, we adopt a triangle mesh representation for the optical surface, and optimize its shape to produce a reflected/refracted light distribution close to the target. To this end, we develop a novel differentiable rendering model that considers the reflection/refraction of light on each face according to its face normal, which provides a physically accurate description of the light transport without the need for sampling. Using this model, we formulate an optimization that directly considers the difference between the resulting image and the target image in terms of both pixel values and image gradients. In addition, we incorporate the fabrication requirement of the final surface by introducing a piecewise smooth regularization in the optimization, which facilitates the CNC milling and polishing of the final design. Our direct optimization, coupled with an accurate rendering model and geometric constraints, leads to final designs that not only closely match the target images in simulated rendering but also produce good results when physically fabricated.

*Corresponding author (juyong@ustc.edu.cn).

Authors' addresses: Yuou Sun, University of Science and Technology of China, Hefei, China, yosun@mail.ustc.edu.cn; Bailin Deng, Cardiff University, Cardiff, United Kingdom, DengB3@cardiff.ac.uk; Juyong Zhang, juyong@ustc.edu.cn, University of Science and Technology of China, Hefei, China.

Due to the nonconvex nature of our optimization formulation, it requires a proper initialization to reach full effectiveness. Therefore, we also develop an optimal transport (OT) based initialization strategy. Although some existing methods [Meyron et al. 2018; Schwartzburg et al. 2014] have utilized OT to compute the lens shape, they only perform OT once between the source intensity and the target image. Unlike these methods, we formulate an OT problem between the current face-based rendering result and the target image to establish their best correspondence, which is used to update the mesh shape and bring it closer to a desirable solution. A major benefit of our formulation is that the OT can be repeatedly applied during optimization, further improving its effectiveness in steering the mesh shape to avoid local minima.

From the perspective of redistributing light, our face-based OT update and rendering guided optimization are complementary and play different roles. The former establishes a correspondence from each face to a target image region with a matching total flux, producing an approximate face location and orientation to improve the result. The latter finetunes the mesh shape, in order to faithfully reproduce the details of the target image. Therefore, by combining these two techniques in an iterative manner, our final optical surface shape can produce an image with a close resemblance to the target.

We evaluate our method using both digital models and physical prototypes created via CNC milling. Experiments show that our approach can produce optical surfaces with high-fidelity reproduction of the target images, with a notable improvement in precision over existing approaches. Our work provides an effective solution for high-precision optical surface design for light control, which may benefit other application domains beyond computer graphics.

To summarize, our contributions include:

- A differentiable rendering model that accurately describes the resulting image for an optical surface represented as a mesh.
- An end-to-end algorithm that directly optimizes the optical surface shape, which reduces the difference between the target and result images while enforcing fabrication requirements, leading to a design that can precisely reproduce the target image.
- A face-based OT initialization strategy, which helps to avoid local minima and enable effective optimization.

2 RELATED WORK

Inverse surface design for light control. Optics design aims to shape optical components to create specific visual effects under given lighting conditions. Lenses, in particular, have been designed for various applications, such as vehicle headlamps [Zhu et al. 2013], LCD backlights [Zhu et al. 2019], and projectors [Damberg et al. 2016; Zhao et al. 2013]. For example, to achieve varied illumination in industrial settings, Ding et al. [2008] devised a lens that transforms LED light sources into rectangular illumination with better uniformity, while the lens developed by Chen et al. [2012] can alter it to parallel light.

Apart from lens design for daily usage, another line of work determines the geometry of lens surface that can form a particular caustic image through reflection or refraction. One of the earliest works is [Finckh et al. 2010], which determines the surface geometry from caustics using the simultaneous perturbation stochastic approximation optimization algorithm that is computationally demanding.

Papas et al. [2011] employ the EM algorithm instead to optimize micro-surfaces that distribute light onto Gaussian kernels representing the target image. However, their results can suffer from artifacts due to discretization and have difficulties handling low-intensity areas. Other researchers have simplified this problem using different techniques. Some construct the Monge-Ampère equation based on local energy conservation [Wu et al. 2013], whereas others represent the lens by integrating a normal field [Kiser et al. 2013] or formulating it as a Poisson reconstruction problem [Yue et al. 2014]. All approaches assume the lens shape as a continuous and smooth surface, significantly reducing the artifacts and improving image quality. Schwartzburg et al. [2014] further addressed the problem of generating high-contrast images which cannot be achieved through globally smooth surfaces. By utilizing optimal transport, they establish a non-bijective mapping between the incident light field and the target distribution, which is used to compute the surface normals and recover the lens shape. Their approach can produce entirely black regions, which was impossible with previous methods. Damberg and Heidrich [2015] revisited this challenge from the perspective of phase-only spatial light modulation, treating the spatial phase modulator as a freeform lens and developing an efficient optimization algorithm. Meyron et al. [2018] developed a parameter-free algorithm for caustic design. They utilize visibility diagrams to find a solution for optimal transport, which balances the distribution of light intensity with energy conservation. By intersecting a 3D Power diagram with planar or spherical domains, their method can directly obtain the surface shape.

Some other works have focused on reflective properties instead of refractive ones. Among them, Weyrich et al. [2009] achieved target patterns by designing a heightfield mirror. Wu et al. [2022] manipulated the shape and texture of a saucer to directly display an image on its surface, while also reflecting another image on a mirror cup. Shen et al. [2023] computed a mirror with engraved scratches, which can display distinct images when viewed from different angles. Tojo et al. [2023] optimized the surface shape to fulfill diverse surface reflectivity requirements.

It is worth mentioning that the concept of inverse design can also be extended to wave optics. Peng et al. [2017] formed a holographic reconstruction problem based on the Gerchberg-Saxton framework. In addition, Jang et al. [2020] designed the first full-color caustic generator using a freeform holographic optical element.

Differentiable rendering. Rendering simulates the appearance of a 3D scene based on lighting and object properties. It can be used to determine the light pattern resulting from a lens design. Unfortunately, traditional rendering procedures contain non-differentiable elements like rasterization and z-buffers, making it challenging to optimize object properties based on the simulated outcome directly. Recently, various research has been devoted to making the rendering process differentiable. Loper and Black [2014] approximated the gradient between pixel-to-vertex coordinates as the image color gradient. However, this method could not propagate gradients into occluded triangles, limiting vertices to receive gradient information solely from nearby pixels. Liu et al. [2019] proposed soft rasterization, which leverages probability distributions and aggregation
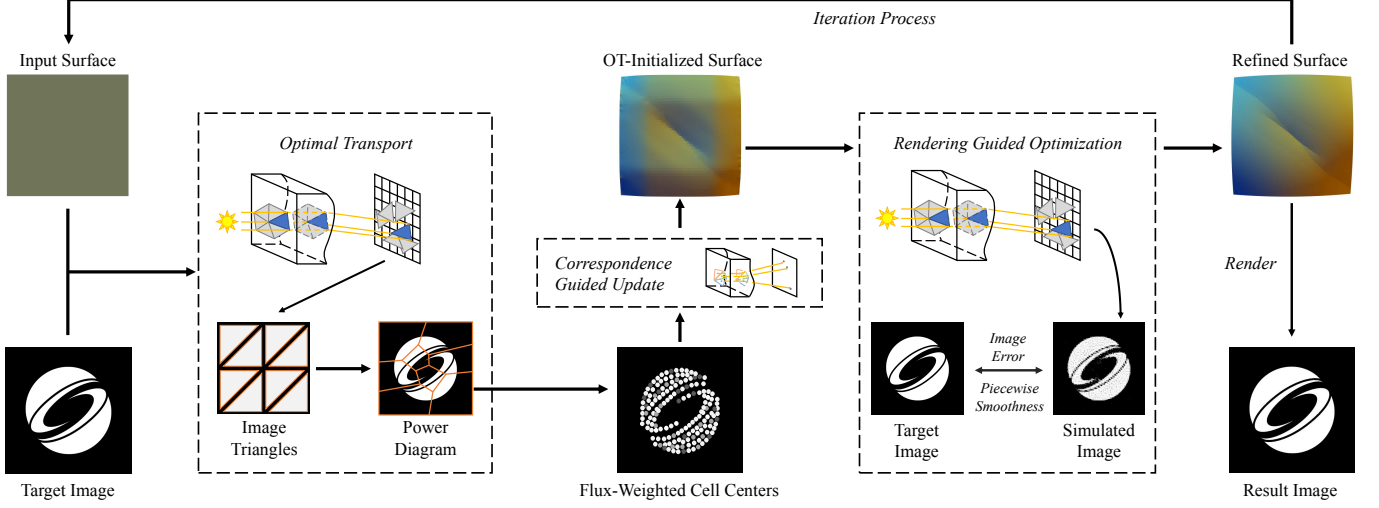
Fig. 2. Pipeline of our algorithm. We iterate between a face-based optimal transport that adjusts correspondence between mesh faces and target image regions, and a rendering guided optimization that improves the details of the resulting image and reduces its difference from the target.

functions to make the rendering pipeline differentiable. As a versatile renderer, Mitsuba 2 [Nimier-David et al. 2019] not only takes into account the complex and intricate properties of light but also ensures the rendering process is differentiable and adaptable to diverse applications. Wang et al. [2022] presented a lens design pipeline with the help of differentiable ray tracing and deep lens systems. The methodologies of both [Nimier-David et al. 2019] and [Wang et al. 2022] can be applied to various reverse design problems including caustic design as mentioned above.

Optimal transport. The lens design problem can also be treated as a problem of light redistribution. An effective tool for resolving density distribution correspondences is optimal transport (OT), which finds the most efficient way of transferring one mass distribution to another based on a cost function. This problem was initially explored by Monge [1781], and later generalized by Kantorovich [1942] to allow for mass splitting. One variant of OT particularly relevant to our problem is the semi-discrete OT, where one distribution is discrete and the other is continuous [Lévy 2015; Mérigot 2011]. Similar to [Schwartzburg et al. 2014] and [Meyron et al. 2018], we also utilize OT for lens design. However, unlike their continuous OT between the source and target distribution, we use a semi-discrete OT for correspondence between mesh faces and a target image.

3 METHOD

Given a light source and a target image and the refractive index of the lens material, we would like to design a lens shape that refracts the light to form an image on a receptive plane that is close to the target image. In the following, we first present our algorithm for a parallel light source. Our pipeline is shown in Fig. 2. Afterward, we discuss its extension to accommodate a point light source.

3.1 Measuring light intensity

Source flux. For parallel light, we follow existing methods and assume the front side of the lens (which faces the light source) is flat and perpendicular to the light direction (see Fig. 3). This avoids refraction by the front surface, thus we only need to optimize the back surface. We then identify the front surface with a planar domain $U \subset \mathbb{R}^2$, such that the back surface is a height field over U. In this paper, we represent the back surface as a triangle mesh. The intensity of the incident light can be represented as a distribution function f on U, where $f(\mathbf{s}) \geq 0 \ \forall \mathbf{s} \in U$, and $\int_{\mathbf{s} \in U} f(\mathbf{s}) d\mathbf{s} = 1$. Since the light does not change direction at the front surface, the flux passing through a triangle t_i on the back surface can be computed as

$$\Phi_i = \int_{\mathbf{s} \in P_U(t_i)} f(\mathbf{s}) d\mathbf{s}, \tag{1}$$

where $P_U(\cdot)$ denotes the projection onto U. For uniform light (i.e., $f(\mathbf{s})$ is constant), $\Phi_i = \frac{A(P_U(t_i))}{A(U)}$ where $A(\cdot)$ denotes the area.
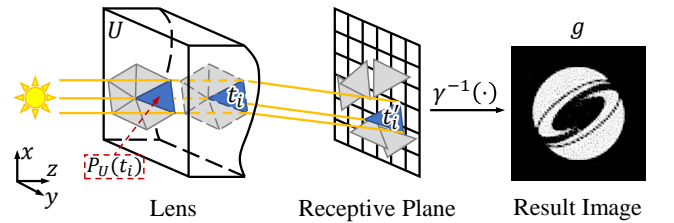
Fig. 3. Our rendering process. The light rays pass through the front surface U of the lens without refraction. At each triangle t_i on the back surfaces, the light rays refract according to the normal of t_i, forming a corresponding triangle t_i' on the receptive plane. The total flux of t_i' is determined by the area of the projection $P_U(t_i)$ from t_i to U. These refracted triangles collectively form the resulting image g after applying the inverse gamma correction $\gamma^{-1}(\cdot)$ (see Eq. (5)).

Target flux. Given a grayscale target image, we compute a value $\widetilde{\Phi}^j$ for each pixel p^j to describe its flux distribution, such that $\sum_{j=1}^{n_p} \widetilde{\Phi}^j = 1$ where n_p is the number of pixels. Note that the flux of pixel p^j is represented not by its grayscale value $\widetilde{g}^j$, but by $\gamma(\widetilde{g}^j)$ where $\gamma(\cdot)$ is the gamma correction. Hence, we compute $\widetilde{\Phi}^j$ as

$$\widetilde{\Phi}^j = \frac{\gamma(\widetilde{g}^j)}{\widetilde{G}}, \tag{2}$$

where $\widetilde{G} = \sum_{j=1}^{n_p} \gamma(\widetilde{g}^j)$ is the total flux of all pixels.

3.2 Differentiable rendering

Given the back surface mesh, we propose a differentiable model to evaluate the flux distribution in a designated area on the receptive plane with the same pixel resolution as the target image (see Fig. 3). At each point of the back surface of the lens, the refracted light ray for an incident ray can be determined using Snell's law $\sin\beta / \sin\alpha = \eta$, where α, β are the angle between the surface normal line and the incident and refracted rays respectively, and η is the ratio between the refractive indices of the medium on two sides of the interface (see inset figure). The refracted ray direction $\mathbf{b}$ can be computed as (see Appendix A for a derivation):

$$\mathbf{b} = \mathbf{n}\sqrt{1 + \eta^2((\mathbf{n} \cdot \mathbf{a})^2 - 1)} + \eta(\mathbf{a} - (\mathbf{a} \cdot \mathbf{n})\mathbf{n}), \tag{3}$$

where $\mathbf{a}, \mathbf{n}$ are unit vectors for the incident ray direction and the surface normal, respectively. If the back surface is represented as a triangle mesh, then for each face t_i its interior points have the same surface normal

$$\mathbf{n}_i = \frac{(\mathbf{v}_i^2 - \mathbf{v}_i^1) \times (\mathbf{v}_i^3 - \mathbf{v}_i^1)}{\|(\mathbf{v}_i^2 - \mathbf{v}_i^1) \times (\mathbf{v}_i^3 - \mathbf{v}_i^1)\|}, \tag{4}$$

where $\mathbf{v}_i^1, \mathbf{v}_i^2, \mathbf{v}_i^3$ are the vertex positions of t_i in the positive orientation. Since incoming light rays remain parallel after passing through the front surface, it is evident that the outgoing rays from t_i are also parallel. The intersection between these rays and the receptive plane forms a triangle t_i', where each vertex of t_i' lies on the refracted ray from a corresponding vertex of t_i. Assuming that there is no energy loss during refraction and the flux inside t_i' is uniform, we allocate the source flux Φ_i of t_i to each pixel p^j on the receptive plane using the intersection area between t_i' and p^j:

$$\Phi_i^j = \frac{\Phi_i \cdot A(t_i' \cap p^j)}{A(t_i')}.$$

Since both t_i' and p^j are convex polygons, we calculate their intersection $t_i' \cap p^j$ by adopting the convex polygon intersection algorithm from [o'Rourke 1998], which has linear complexity with respect to the total number of polygon vertices. Moreover, a non-empty intersection $t_i' \cap p^j$ must also be a convex polygon, and its area $A(t_i' \cap p^j)$ can be easily calculated in closed form [o'Rourke 1998]. The total flux Φ^j for a pixel p^j is the sum of its allocated flux from all triangles on the back surface:

$$\Phi^j = \sum_{i=1}^{n_t} \Phi_i^j,$$

where n_t is the number of triangles. For uniform source flux, our model produces a physically accurate flux distribution on the pixels without sampling the light rays.

For each image triangle t_i', we use a scan-line algorithm within its bounding box to determine the pixels it intersects with. Moreover, the computation for each triangle is independent and can be carried out in parallel, enabling real-time results. Our model is differentiable w.r.t. the back surface shape, allowing us to optimize the lens shape using a gradient-based solver, as explained in the next subsection.

Unlike existing differentiable rendering methods [Nimier-David et al. 2019; Wang et al. 2022], our approach utilizes flux-based rendering rather than ray tracing, which allows us to compute the rendering result in closed forms without the need for sampling.

3.3 Rendering guided optimization

Given a mesh for the back surface, we optimize its vertex positions such that the refracted rays result in an image as close to the target image as possible. This section presents our formulation.

Optimization settings. We assume the incident light to be in the $+z$ direction. The domain U is chosen as $[0, W] \times [0, H]$ in the x-y space for a target image in $W \times H$ resolution. We triangulate U by sampling uniformly at grid positions and joining them with regular connectivity, and then assign a z-coordinate to each vertex (chosen as 0 by default) to obtain an initial mesh for the back surface. The receptive plane is fixed at the plane $z = z_{\text{focal}}$ where z_{focal} is a user-specified focal length, and its image region is chosen as $[0, W] \times [0, H]$ to align with the back surface. The front surface is fixed at the plane $z = z_{\text{front}}$ where z_{front} is specified by the user. (The exact value of z_{front} is irrelevant for parallel light but needs to be considered when using a point light source.) We use the mesh vertex coordinates as optimization variables. To maintain alignment with U, we fix the x- and/or y-coordinates of the boundary vertices at their initial values.

Image terms. For closeness between the resulting image and the target, we introduce a term to penalize their pixel value differences:

$$E_{\text{img}} = \sum_{j=1}^{n_p} (g^j - \widetilde{g}^j)^2, \tag{5}$$

where $g^j = \gamma^{-1}(\widetilde{G}\Phi^j)$ is the resulting pixel value computed from the flux distribution using Equation (2). Moreover, we use the following term to penalize the difference between their image gradients, in order to preserve the salient features in the target image:

$$E_{\text{grad}} = \|\mathbf{G}_x - \widetilde{\mathbf{G}}_x\|_F^2 + \|\mathbf{G}_y - \widetilde{\mathbf{G}}_y\|_F^2, \tag{6}$$

where $\mathbf{G}_{x,y}$ and $\widetilde{\mathbf{G}}_{x,y}$ represent the gradient matrices of the resulting image and target images, respectively. Additionally, we introduce a term to inhibit the refracted rays from reaching a location outside the designated imaging region on the receptive plane:

$$E_{\text{bdr}} = \sum_{i=1}^{n_t} \sum_{k=1}^{3} \|\widetilde{\mathbf{v}}_i^k - P_I(\widetilde{\mathbf{v}}_i^k)\|^2, \tag{7}$$

where $\widetilde{\mathbf{v}}_i^k \in \mathbb{R}^2$ is k-th vertex position of the image triangle t_i', and $P_I(\widetilde{\mathbf{v}}_i^k) \in \mathbb{R}^2$ is the closest point to $\widetilde{\mathbf{v}}_i^k$ from the imaging region.
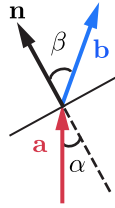
Barrier terms. For the result to be valid, our optimization variables need to satisfy some constraints. First, as mentioned in Sec. 3.1, the back surface should be a height field over the domain U. Thus the projections $\{P_U(t_i)\}$ of back surface triangles onto U should have no overlap except for the shared edges and vertices. This is equivalent to the condition that the following signed area is positive for each projected triangle:

$$\overline{A}(P_U(t_i)) = \frac{1}{2} \det([P_U(\mathbf{v}_i^2) - P_U(\mathbf{v}_i^1), \ P_U(\mathbf{v}_i^3) - P_U(\mathbf{v}_i^1)]).$$

Additionally, very small triangles may cause numerical issues during optimization and should be avoided. Thus we introduce a barrier term $f_{\text{area}}\left(\overline{A}(P_U(t_i))\right)$ for the signed area $\overline{A}(P_U(t_i))$, where

$$f_{\text{area}}(a) = \begin{cases} 0 & \text{if } a \geq \epsilon_2, \\ \left(\frac{\epsilon_2 - \epsilon_1}{a - \epsilon_1} - 1\right)^2 & \text{if } \epsilon_1 < a < \epsilon_2, \\ +\infty & \text{otherwise,} \end{cases} \tag{8}$$

and ϵ_1, ϵ_2 are threshold values that satisfy $\epsilon_2 > \epsilon_1 > 0$. Here $f_{\text{area}} \rightarrow +\infty$ when $a \rightarrow \epsilon_1$, thus the term ensures each signed area is larger than ϵ_1. Moreover, f_{area} decreases to zero when the signed area is larger than ϵ_2, to avoid affecting triangles that are large enough.

In addition, the refracted ray direction in Eq. (3) is not well-defined if $1 + \eta^2((\mathbf{n} \cdot \mathbf{a})^2 - 1) < 0$. In this case, *total internal reflection* occurs and the light is reflected back to the lens interior. We introduce another barrier term $f_{\text{tir}}(t_i)$ for each triangle to prevent such scenarios:

$$f_{\text{tir}}(t_i) = \begin{cases} \left(1 + \eta^2((\mathbf{n}_i \cdot \mathbf{a})^2 - 1)\right)^{-1} & \text{if } 1 + \eta^2((\mathbf{n}_i \cdot \mathbf{a})^2 - 1) > 0, \\ +\infty & \text{otherwise.} \end{cases} \tag{9}$$

Interestingly, it can be shown that the term f_{tir} not only prevents total internal reflection but also ensures the refracted light rays do not intersect with the lens surface again (see Apendix B for a proof). Thanks to this property, there is no need to consider further refraction, which helps to simplify our formulation.

Applying the two barrier terms to all triangles, we obtain

$$E_{\text{barr}} = \sum_{i=1}^{n_t} f_{\text{area}}\left(\overline{A}(P_U(t_i))\right) + f_{\text{tir}}(t_i).$$

Smoothness terms. For a given image, there may be multiple optical surface shapes that can achieve the target distribution of light intensity (see Fig. 4 for a schematic example). We prefer optical surfaces with smoother shapes, since it can be difficult for CNC milling to accurately reproduce a mesh surface with rapid variation of normals within a local area [Zou 2021]. Additionally, optical surfaces often require polishing as part of their fabrication process. An optical surface design with a rough shape can be more difficult to polish than a smooth surface, especially on a surface represented as a high-resolution mesh: on such surfaces, a rough area may contain small concavities that are not reachable by the polishing tool or small convexities that can be easily affected when the surrounding area is polished; these can lead to under-polishing or over-polishing, both of which will affect the resulting image quality. On the other hand, we cannot expect the surface to be globally smooth in general, as sharp creases may be necessary to achieve some light patterns; for example, if the target image contains two bright areas separated
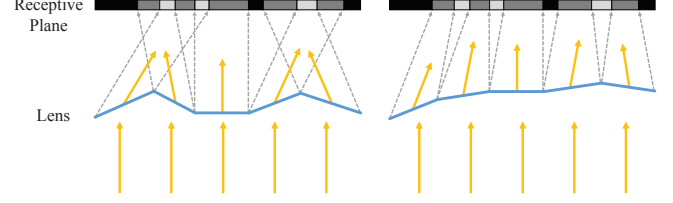

Fig. 4. An illustration of different lens shapes that produce the same distribution of light. We prefer a smoother lens shape as it facilitates fabrication.

by a pure black stripe, then it cannot be generated from a globally smooth lens surface as the refracted lights from such a lens shape would result in a single continuous bright region. Therefore, we need to enforce *piecewise* smoothness for the back surface mesh, such that it is smooth in most areas while allowing for some sharp edges that separate different smooth patches.

A common way to enforce smoothness is to introduce a regularization term for the surface curvature. However, as the mesh vertex positions determine the curvature in a highly non-linear way, directly regularizing the curvature as a function of vertex positions would result in a complex term that is not easy to optimize. Instead, we propose a more tractable approach based on the Weingarten map, a linear operator that captures local surface geometry. Intuitively, the Weingarten map at a point describes how the surface normal changes as we move along different tangent directions. We introduce auxiliary variables on each mesh face to represent the local Weingarten map, and formulate a target function term to adaptively enforce its consistency with the normals of adjacent faces belonging to the same smooth patch; this allows sharp edges to emerge between different patches. Moreover, as the Weingarten map determines the local surface curvature, we further utilize the auxiliary variables to introduce another target function term that regularizes the curvature on each mesh face. These two terms combined enforce the piecewise smoothness of the mesh surface.

More specifically, for a point $\mathbf{p}$ on a smooth surface, the Weingarten map is a linear map $W_{\mathbf{p}} : T_{\mathbf{p}} \mapsto T_{\mathbf{p}}$ within the tangent space $T_{\mathbf{p}}$ at $\mathbf{p}$, and it maps each tangent vector $\mathbf{d} \in T_{\mathbf{p}}$ to the differential of the surface normal at $\mathbf{p}$ along $\mathbf{d}$. The map $W_{\mathbf{p}}$ can be represented as a 2×2 symmetric matrix with respect to an orthonormal basis of $T_{\mathbf{p}}$ [Spivak 1999]. Thus, to discretize the Weingarten map, we first choose for each mesh face t_i two orthonormal basis vectors $\mathbf{e}_{i,1}$ and $\mathbf{e}_{i,2} = \mathbf{n}_i \times \mathbf{e}_{i,1}$, where $\mathbf{e}_{i,1}$ is the unit vector for an edge of t_i and $\mathbf{n}_i$ is the unit face normal as defined in Eq. (4). Then we introduce three auxiliary variables $a_i, b_i, c_i \in \mathbb{R}$ to represent the symmetric matrix for the Weingarten map under this basis:

$$\mathbf{M}_i = \begin{bmatrix} a_i & c_i \\ c_i & b_i \end{bmatrix}. \tag{10}$$

The matrix $\mathbf{M}_i$ needs to be consistent with the face normals in the neighborhood of t_i. We discretize this condition in a way similar to [Rusinkiewicz 2004]. First, we associate the normal $\mathbf{n}_i$
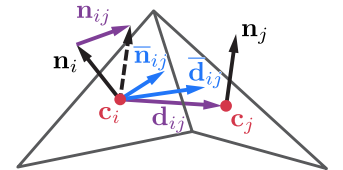
of each face t_i to its centroid $\mathbf{c}_i$. Then, for each adjacent face t_j that shares a common edge with t_i, we compute the difference vectors

$$\mathbf{d}_{ij} = \mathbf{c}_j - \mathbf{c}_i, \quad \mathbf{n}_{ij} = \mathbf{n}_j - \mathbf{n}_i, \tag{11}$$

between the positions and normals of their centroids, and project them onto the tangent plane of t_i to derive their local coordinates

$$\overline{\mathbf{d}}_{ij} = \mathbf{B}_i^T \mathbf{d}_{ij}, \quad \overline{\mathbf{n}}_{ij} = \mathbf{B}_i^T \mathbf{n}_{ij}, \tag{12}$$

where $\mathbf{B}_i = [\mathbf{e}_{i,1}, \mathbf{e}_{i,2}] \in \mathbb{R}^{3\times2}$ is the tangent basis matrix for t_i (see the inset figure above). The matrix $\mathbf{M}_i$ should map $\overline{\mathbf{d}}_{ij}$ to $\overline{\mathbf{n}}_{ij}$ approximately. Thus, we measure the consistency between $\mathbf{M}_i$ and the geometry of t_i and t_j via the term

$$\delta(t_i, t_j) = \frac{\|\mathbf{M}_i \overline{\mathbf{d}}_{ij} - \overline{\mathbf{n}}_{ij}\|^2}{\|\overline{\mathbf{d}}_{ij}\|^2}. \tag{13}$$

In addition, if t_i and t_j belong to the same smooth patch, then the geometry of t_i and t_j should be consistent with the Weingarten matrix at t_j as well. Thus, we define the consistency error term for the common edge e_{ij} between t_i and t_j as

$$h(e_{ij}) = \delta(t_i, t_j) + \delta(t_j, t_i).$$

To achieve piecewise smoothness, we should reduce the value of h on most edges, while allowing for large values on some sharp edges that separate different smooth patches. In other words, we need to achieve sparsity of the values $\{h(e_{ij})\}$. To this end, we use the Welsch function [Holland and Welsch 1977] as a robust norm to promote sparsity, and derive an edge smoothness term as

$$E_{\text{edge}} = \sum_{e_{ij} \in \mathcal{E}_I} \Psi_\nu(\sqrt{h(e_{ij})}),$$

where $\mathcal{E}_I$ denotes the set of internal mesh edges, Ψ_ν is the Welsch function

$$\Psi_\nu(x) = 1 - \exp(-\frac{x^2}{2\nu^2}), \tag{14}$$

and ν is a user-specified parameter. The Welsch function attains the minimum value at $x = 0$, is monotonically increasing on $[0, +\infty)$, and is bounded from above by 1. In this way, it penalizes the errors $\{h(e_{ij})\}$ while limiting the influence of large error values, thus allowing for sharp edges.

Moreover, on a smooth surface, the mean curvature at a point equals half the trace of its Weingarten map [do Carmo 1976]. Thus we derive a discrete mean curvature measure for the face t_i from the matrix $\mathbf{M}_i$ as

$$H_i = \frac{a_i + b_i}{2}.$$

Using this relation, we define a surface smoothness term that penalizes the mean curvature:

$$E_{\text{face}} = \sum_{i=1}^{n_t} (H_i)^2 A_i, \tag{15}$$

where A_i is the area of face t_i. Note that E_{face} can be considered as a discrete integral of the squared mean curvature over the mesh. We combine E_{face} and the term E_{edge} introduced previously to enforce smoothness within each patch while allowing for sharp edges to emerge between adjacent patches.

Finally, to avoid poor triangulation, we also introduce a Laplacian smoothness term for the projections of the mesh vertices $\{\mathbf{v}_j\}$ onto the height field domain U:

$$E_{\text{lap}} = \sum_{j \in \mathcal{V}_I} \left\| P_U(\mathbf{v}_j) - \frac{1}{|\mathcal{N}_j|} \sum_{i \in \mathcal{N}_j} P_U(\mathbf{v}_i) \right\|^2, \tag{16}$$

where $\mathcal{V}_I$ denotes the index set of interior vertices, and $\mathcal{N}_j$ is the index set of neighbor vertices for vertex j.

Our overall smoothness term is a weighted sum

$$E_{\text{smooth}} = E_{\text{face}} + \tau_1 E_{\text{edge}} + \tau_2 E_{\text{lap}}. \tag{17}$$

Overall formulation and numerical solving. Combining the terms proposed above, we derive an optimization problem

$$\min\ \lambda_1 E_{\text{img}} + \lambda_2 E_{\text{grad}} + \lambda_3 E_{\text{bdr}} + \lambda_4 E_{\text{smooth}} + \lambda_5 E_{\text{barr}}, \tag{18}$$

where $\{\lambda_i\}$ are user-specified weights. We solve it with an L-BFGS solver [Liu and Nocedal 1989]. For efficiency, we evaluate the term values and gradients in parallel on the GPU, and use automatic differentiation to compute the derivatives. Additionally, the starting mesh must satisfy the conditions enforced by the barrier terms. This is achieved with a flat surface parallel to the front.

Some existing approaches [Meyron et al. 2018; Schwartzburg et al. 2014; Yue et al. 2014] indirectly optimize the lens surface mesh using auxiliary features (such as normal fields) derived from the target image, which can lead to accumulated errors. In contrast, our optimization is directly driven by the difference between the rendered image and the target, both in terms of per-pixel intensity and image gradient. This end-to-end formulation allows fine-grained optimization of the surface to accurately reproduce the target.

3.4 Face-based optimal transport initialization

Due to the nonconvex nature of our optimization problem, a proper initialization is often necessary to help obtain an optimal solution. This can be seen in the first column of Figure 10, where initializing the mesh with a trivial flat surface produces unsatisfactory results. This is because L-BFGS is inherently a local solver that makes a relatively small change in each iteration, and may stop at a local minimum before reaching a better solution far away from the initial point. Therefore, we need a procedure to make larger changes and move the current solution closer to a desirable result, to avoid getting stuck at local minima. Similar to [Schwartzburg et al. 2014], we consider our lens shape optimization as re-allocating the flux on the receptive plane to achieve a target distribution, and use optimal transport (OT) to determine a correspondence that guides the allocation. However, unlike [Schwartzburg et al. 2014] that perform OT between the source and target flux distributions, we formulate an OT problem between the current face-based distribution and the target, and modify the current mesh accordingly.

Optimal transport. Recall that each face t_i forms a triangle t_i' on the image plane with a total flux Φ_i. To facilitate our formulation, we simplify each t_i' to a point at its centroid $\mathbf{c}_i'$ with a flux Φ_i. Each pixel p^j of the target image has a total flux $\widetilde{\Phi}^j$ given by Eq. (2), and we consider each point $\mathbf{s}$ in the interior of p^j to have a flux $\widetilde{\Phi}_{\mathbf{s}} = \widetilde{\Phi}^j / A_p$, where A_p is the pixel area. Consider a partition of the image region I into n_t cells $\{\Omega_i\}$, each corresponding to one point

$\mathbf{c}'_i$, so that the flux at $\mathbf{c}'_i$ can be transported to Ω_i to match the target flux distribution (see Fig. 5), i.e., Φ_i is the same as the total target flux $\widetilde{\Phi}_T(\Omega_i)$ within Ω_i, where the latter is computed as

$$\widetilde{\Phi}_T(\Omega_i) = \sum_{j=1}^{n_p} \frac{\widetilde{\Phi}^j \cdot A(p^j \cap \Omega_i)}{A(p^j)}. \qquad (19)$$

Among all such partitions, we search for one with the least cost of flux transport, which can be described as

$$\min_{\{\Omega_i\}} \quad \sum_{i=1}^{n_t} \int_{\mathbf{s} \in \Omega_i} \left\| \mathbf{s} - \mathbf{c}'_i \right\|^2 \widetilde{\Phi}_{\mathbf{s}} d\mathbf{s},$$

$$\text{s.t.} \quad \Phi_i = \widetilde{\Phi}_T(\Omega_i), \quad \bigcup_{i=1}^{n_t} \Omega_i = I.$$

Here the transport cost from $\mathbf{c}'_i$ to Ω_i is measured by the integrated squared distance between $\mathbf{c}'_i$ and each point in Ω_i, weighted by the flux at the point. It has been shown in [Lévy 2015] that the solution can be represented using a power diagram generated from the points $\mathbf{c}'_1, \ldots, \mathbf{c}'_{n_t}$ with appropriate weights $\mathbf{w} = (w_1, \ldots, w_{n_t})$, such that $\Omega_i = I \cap \mathcal{P}_i^{\mathbf{w}}$ where $\mathcal{P}_i^{\mathbf{w}}$ is the power cell corresponding to $\mathbf{c}'_i$:

$$\mathcal{P}_i^{\mathbf{w}} = \{\mathbf{x} \mid \|\mathbf{x} - \mathbf{c}'_i\|^2 - w_i \le \|\mathbf{x} - \mathbf{c}'_j\|^2 - w_j, \forall j = 1, \cdots, n_t\}.$$

The weights $\mathbf{w}$ is the minimizer of a convex energy [Lévy 2015]:

$$H(\mathbf{w}) = \sum_{i=1}^{n_t} \left(-\int_{\mathbf{s} \in \mathcal{P}_i^{\mathbf{w}}} (\|\mathbf{x} - \mathbf{c}'_i\|^2 - w_i) \widetilde{\Phi}_{\mathbf{s}} d\mathbf{s} - \Phi_i w_i \right),$$

whose gradient has the form $\partial H(\mathbf{w})/\partial w_i = \widetilde{\Phi}_T(\mathcal{P}_i^{\mathbf{w}}) - \Phi_i$. We minimize $H(\mathbf{w})$ with an L-BFGS solver, using CGAL [The CGAL Project 2023] to generate the power diagram. To evaluate H accurately, we convert it to a quartic polynomial using Green's theorem and Newton-Leibniz formula (see Appendix C).

Correspondence guided update. After deriving the partition $\{\Omega_i\}$ using OT, we update the back surface mesh such that the image t'_i of each triangle aligns better with its corresponding region Ω_i in terms of both location and total flux, which helps to bring the resulting flux distribution closer to the target (see inset figure). To this end, we formulate an optimization

$$\min \ \gamma_1 E_{\text{align}} + \gamma_2 E_{\text{flux}} + \lambda_3 E_{\text{smooth}} + \gamma_4 E_{\text{barr}}, \qquad (20)$$

where $\{\gamma_i\}$ are user-specified weights. Here

$$E_{\text{align}} = \sum_{i=1}^{n_t} \|\mathbf{c}'_i - \widetilde{\mathbf{c}}_i\|^2 \qquad (21)$$

enforces alignment between the centroid $\mathbf{c}'_i$ of t'_i and the flux-weighted centroid $\widetilde{\mathbf{c}}_i$ of Ω_i:

$$\widetilde{\mathbf{c}}_i = \frac{\sum_{j=1}^{n_p} \mathbf{C}(\Omega_i \cap p^j) \widetilde{\Phi}^j A(\Omega_i \cap p^j)}{\sum_{j=1}^{n_p} \widetilde{\Phi}^j A(\Omega_i \cap p^j)}, \qquad (22)$$

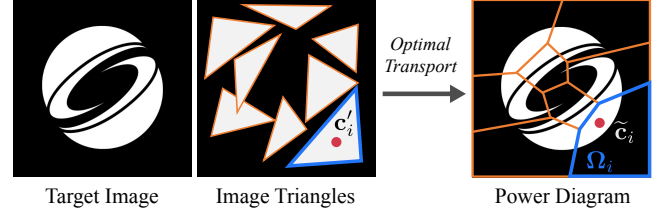Target Image · Image Triangles · Power Diagram

Fig. 5. An illustration of our discrete optimal transport process. Given the current image triangles resulting from our rendering model, we associate the total flux of each triangle to its centroid $\mathbf{c}'_i$, and optimize a partition of the target image such that the flux at $\mathbf{c}'_i$ can be minimally transported to its corresponding cell Ω_i in the partition to match the target image. Then we compute the flux-weighted weighted centroid $\widetilde{\mathbf{c}}_i$ of each cell to guide the update of the lens shape.

where $\mathbf{C}(\cdot)$ denotes the centroid. The term

$$E_{\text{flux}} = \sum_{i=1}^{n_t} (\Phi_i - \Phi_i^{\text{old}})^2 \qquad (23)$$

maintains the total flux of each triangle, where Φ_i^{old} denotes the flux of t_i before the update. E_{smooth} and E_{barr} are the same as in Eq. (18). The problem (20) is solved using L-BFGS.

Unlike [Schwartzburg et al. 2014] where OT is performed only once, our face-based OT formulation can be repeatedly applied to the mesh. In our implementation, we adopt a coarse-to-fine strategy, starting from a coarse input mesh and a downsampled target image, and then alternate between OT and our rendering guided optimization for six iterations. Then we subdivide the mesh and increase the target image resolution, and repeat the alternation between OT and optimization. This process is repeated until the final resolution is reached. In this way, our method first generates a coarse mesh with a resulting image that roughly aligns with the target, then gradually refines and improves its details for closer resemblance as the resolution increases. Fig. 6 shows an example of this process. In addition, following existing work that utilizes the Welsch function as robust norm [Zhang et al. 2022], we gradually decrease the parameter ν in Eq. (14), to incrementally reduce the influence of large errors and allow sharp edges to emerge during the optimization.

Fig. 7 showcases an example of the final optimized lens surface for the target image in Fig. 1, along with a close-up view of its triangulation. Thanks to the smoothness terms and barrier terms in our rendering-guided optimization, our method produces a high-quality mesh representation that contains the sharp edges needed for reproducing the discontinuities in the target image, while maintaining the smoothness within each patch and avoiding poor triangulation.

3.5 Point light source

Besides refraction of parallel light, our approach can be extended to handle a point light source and reflection. In the following, we present the extension to a point light source. Further discussion about reflection can be found in Appendix D.

With a given point light source, the light refracts at the front surface, and we cannot directly obtain the incident light direction for a back surface mesh vertex in a closed form in general: the

Increasing Mesh Resolution

512x512	32x32	32x32	32x32	32x32	64x64	128x128	512x512

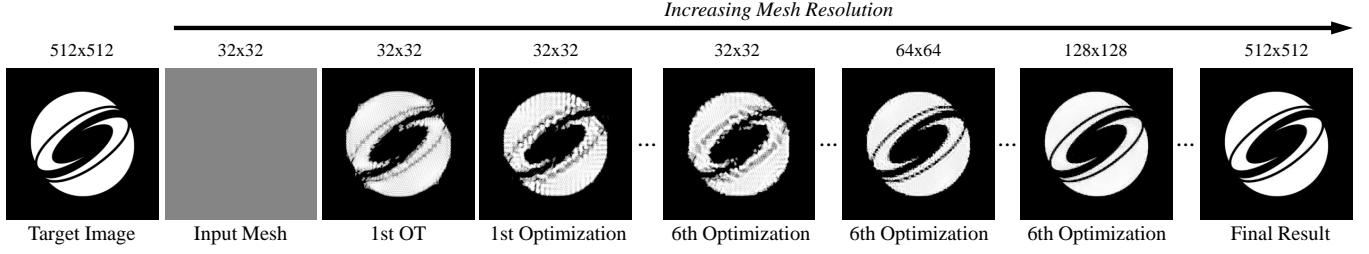| Target Image | Input Mesh | 1st OT | 1st Optimization | 6th Optimization | 6th Optimization | 6th Optimization | Final Result |

Fig. 6. Our iterative optimization process. We show the resulting images derived from the intermediate results of our optimization process. The numbers on top are the resolution of the target image (the leftmost column) or the intermediate mesh (the other columns). The label at the bottom indicates the completed numbers of iterations for OT or rendering guided optimization at the current resolution.

computation of such a direction would involve solving a quartic equation instead. To avoid this problem, we propose an alternative representation of the lens shape. We consider the general case that the front surface is fixed and is a height field over a domain U in the x-y space with a parametric form $z = h(x, y)$, and the back surface is a triangle mesh. Additionally, we assume the following:

(A1) The rays refracted at the front surface define a one-to-one correspondence $\mathcal{H}$ from the front surface to the back surface.

(A2) The front surface has no self-occlusion w.r.t. the light source.

(A3) The direction vector of any refracted light ray has a positive z-coordinate.

Based on Assumption (A1), the back surface mesh induces a triangulation on the front surface, such that each vertex $\mathbf{v}_i^f$ of the front surface corresponds to a vertex $\mathbf{v}_i^b = \mathcal{H}(\mathbf{v}_i^f)$ on the back surface (see inset figure). Let z_i^b be the z-coordinate of $\mathbf{v}_i^b$, and (x_i^f, y_i^f) be the x- and y-coordinates of $\mathbf{v}_i^f$. Then using the parametric form of the front surface, we can derive the position and the normal of $\mathbf{v}_i^f$ as

$$\mathbf{v}_i^f = (x_i^f, y_i^f, h(x_i^f, y_i^f)), \quad \mathbf{n}_i^f = (-h_x(x_i^f, y_i^f), -h_y(x_i^f, y_i^f), 1).$$

The incident light direction at $\mathbf{v}_i^f$ is $\mathbf{a}_1^f = \mathbf{v}_i^f - \mathbf{q}$ where $\mathbf{q}$ is the light source position. We can then compute the refracted light direction $\mathbf{b}_i^f$ at $\mathbf{v}_i^f$ using Eq. (3). By definition, $\mathbf{b}_i^f$ is also the incident light direction at the back surface vertex $\mathbf{v}_i^b$. Moreover, as $\mathbf{v}_i^f$ lies on the ray emitted from $\mathbf{v}_i^f$ along direction $\mathbf{b}_i^f$, there exists a value t_i with

$$\mathbf{v}_i^b = \mathbf{v}_i^f + t_i \mathbf{b}_i^f. \tag{24}$$

Based on Assumption (A3), it is easy to know that

$$t_i = \frac{z_i^b - h(x_i^f, y_i^f)}{z_i^r}, \tag{25}$$

where z_i^r is the z-coordinate of $\mathbf{b}_i^f$. Substituting (25) into (24), we obtain the coordinates of $\mathbf{v}_i^b$. To summarize, the position and incident light direction for each back surface vertex $\mathbf{v}_i^b$ can be parameterized
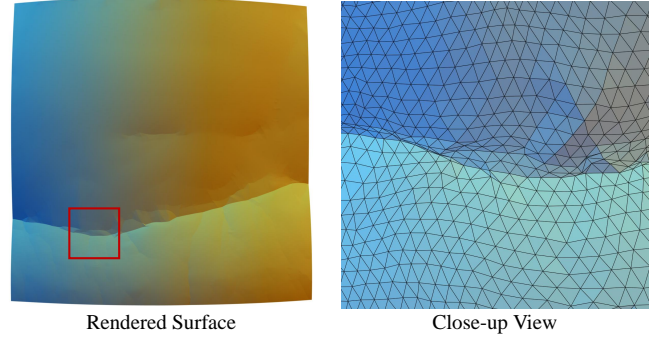

Rendered Surface   Close-up View

Fig. 7. The optimized lens shape for the target image in Fig. 1. The left image shows the full optimized surface, while the right image provides a close-up view of the triangulation in the region highlighted by the red rectangle. The mesh contains sharp edges that are necessary for reproducing the discontinuities in the target image, while maintaining the smoothness within each patch and avoiding poor triangulation.

by its z-coordinate together with the (x, y)-coordinates of its corresponding front surface vertex $\mathbf{v}_i^f$. Therefore, we use these parameters as variables to encode the lens shape and perform optimization.

It is worth noting that the mesh representation in Sec. 3.3 is a special case of the representation described here. In particular, with a parallel light source, the front surface vertices have the same (x, y)-coordinates as their corresponding back surface vertices. Therefore, the variables in our generalized representation are exactly the coordinates of the back surface vertices, which is the same as in Sec. 3.3.

Besides the representation, the calculation of flux for a back surface triangle must also be changed when using a point light source. We assume the light source to be Lambertian (i.e., with the same intensity along different directions). Then from Assumption (A2) we can compute the flux for a back surface triangle t_i^b as:

$$\Phi_i = \frac{S(t_i^f)}{\sum_{j=1}^{n_t} S(t_j^f)},$$

where t_i^f is the corresponding front surface triangle for t_i^b, and $S(\cdot)$ denotes the solid angle w.r.t. the light source.

Table 1. Experimental settings of our results, and the MAE compared to the target image. The results in the last two rows are based on refraction of a point light source and reflection of parallel light respectively, while the other results are based on refraction of parallel light.

Image	#Pixels	#Vertices	Lens Size (cm)	Image Size (cm)	Focal Length (cm)	MAE ($\times 10^{-3}$)
SIGGRAPH	512^2	641×737	10×10×1.7	10×10	30	1.029
Einstein	512^2	641×737	10×10×1.7	10×10	30	2.128
Train	256^2	641×737	10×10×1.6	10×10	30	3.470
Butterfly	256^2	641×737	10×10×1.0	10×10	30	0.054
Calligraphy	800×400	833×481	20×10×1.4	20×10	40	6.235
Newton	256^2	641×737	10×10×0.8	10×10	30	0.001
Butterfly (pt.)	256^2	641×737	10×10×0.9	20×20	60	1.586
Newton (refl.)	256^2	641×737	10×10×0.7	10×10	60	0.000

4 RESULTS

In this section, we verify the effectiveness of our algorithm, and showcase physical prototypes to demonstrate its practicality. All results in this paper are computationally simulated except for those explicitly identified as physical prototype photographs.

Implementation details. The experimental settings for the examples presented in this paper are summarized in Table 1. For the example in the last row, the initial reflective surface has an incident angle of 30° and is parallel to the receptive plane. The weights for Eqs. (18) and (20) are reported in Appendix E.

4.1 Results of our method

Accuracy of our rendering model. In Fig. 8, we validate the accuracy of our triangle-based rendering by comparing our rendered results of the final image with those generated by the physically-based renderer LuxCoreRender[1], using examples of refraction (with both parallel and point light sources) and reflection. It is evident that our triangle-based rendering closely resembles the ray-traced rendering in both scenarios, confirming the reliability of our rendering model. In contrast to LuxCoreRender, our approach achieves real-time rendering by computing only the position of each triangle, without the need for complex sampling techniques to trace individual light rays. Additionally, our rendering model is differentiable, allowing seamless integration into our end-to-end optimization.

In Fig. 8, we also evaluate the difference between the rendered images and the target image using their *mean absolute error* (MAE):

$$\text{MAE} = \frac{1}{n_p} \sum_{i=1}^{n_p} e_i, \tag{26}$$

where e_i is the pixel-wise absolute error defined as:

$$e_i = \frac{|p^i - q^i|}{R}, \tag{27}$$

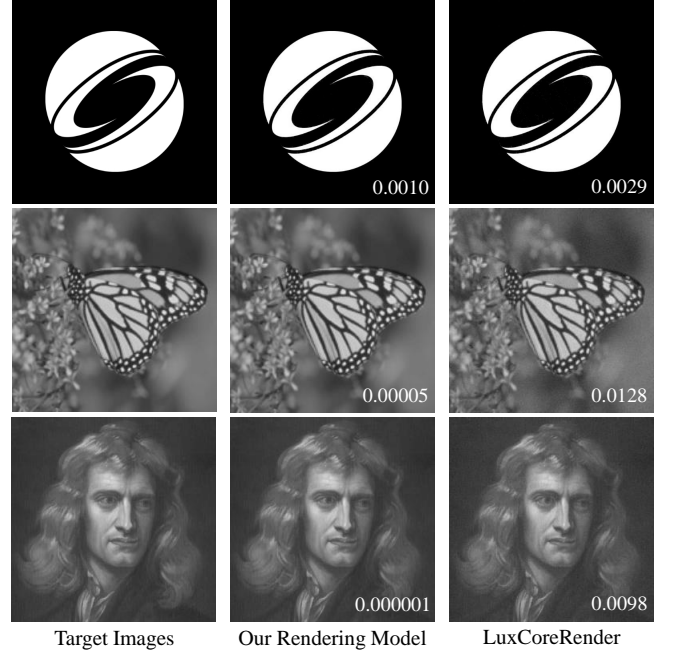
[1]https://luxcorerender.org/

Fig. 8. Accuracy of our rendering model. The three examples from top to bottom are based on refraction of parallel light, refraction of point light, and reflection of parallel light, respectively. The number in the lower right corner of each image represents the MAE with the target image. Newton portrait courtesy of Isaac Newton Institute for Mathematical Sciences, University of Cambridge.

with p^i, q^i being corresponding pixels from the images to be compared, and R being the maximum possible pixel value. The MAE values show that regardless of the rendering method used, our resulting image is very close to the target image.

Effectiveness of our optimization. We evaluate the effectiveness of our optimization approach by comparing the target image with the resulting image generated using our rendering model, measuring their MAE. The MAE values for different examples are shown in the last column of Table 1. Furthermore, Fig. 9 visualizes the pixel errors for some examples using color-coding. Both the MAE values and the pixel error maps confirm that our method successfully computes lens shapes that produce images nearly identical to the target.

4.2 Effectiveness of algorithmic components

In Fig. 10, we conduct controlled experiments to verify the effectiveness of each component in our algorithm.

The column "Without OT Initialization" in Fig. 10 highlights the importance of initialization. For high-contrast images such as the SIGGRAPH logo, it is challenging to rely solely on image errors to move the light from a black region to a distant area where it is needed. For the grayscale image, although a simple plane initialization without OT achieves a satisfactory result, there are still slight tone discrepancies compared to the target image, and wrinkled artifacts that affect the overall visual quality.
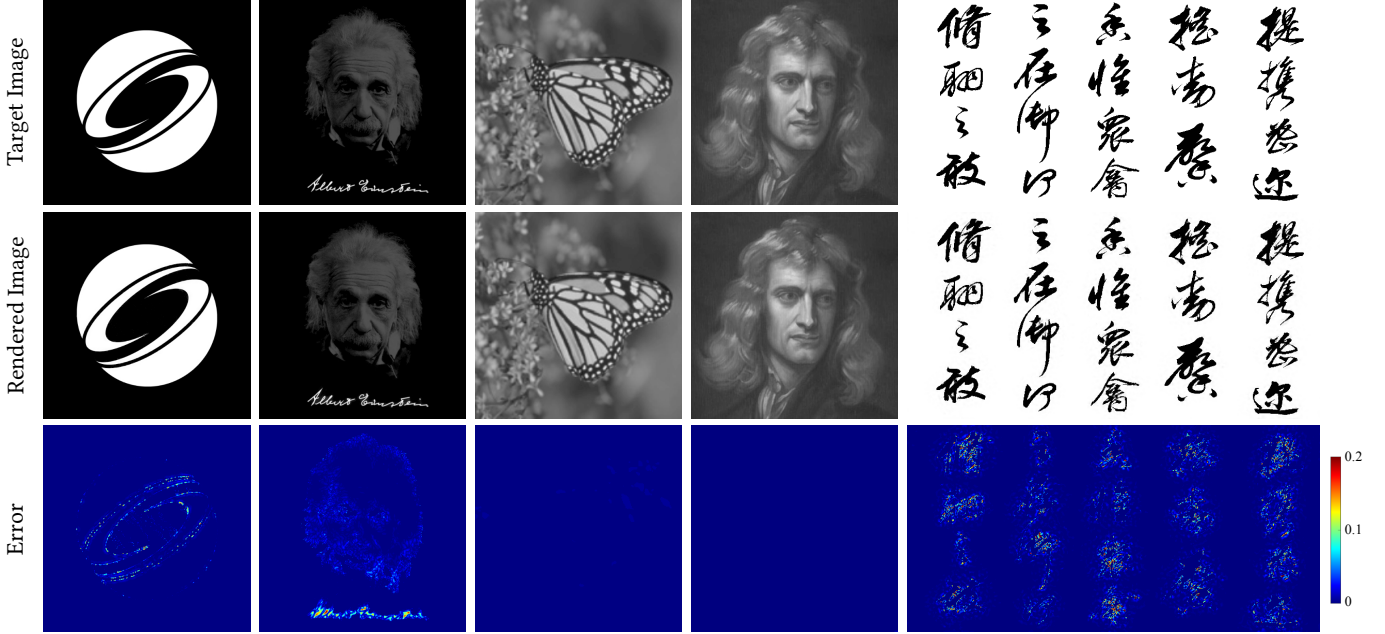
Fig. 9. Our optimization produces results with their rendered image very close to the target images. From top to bottom: target images, rendered resulting images of our lens, and color-coding of pixel-wise absolute errors defined in Eq. (27). Einstein portrait © Magnum/IC photo; Newton portrait courtesy of Isaac Newton Institute for Mathematical Sciences, University of Cambridge; Calligraphy image source: National Palace Museum, Taipei, CC BY 4.0.

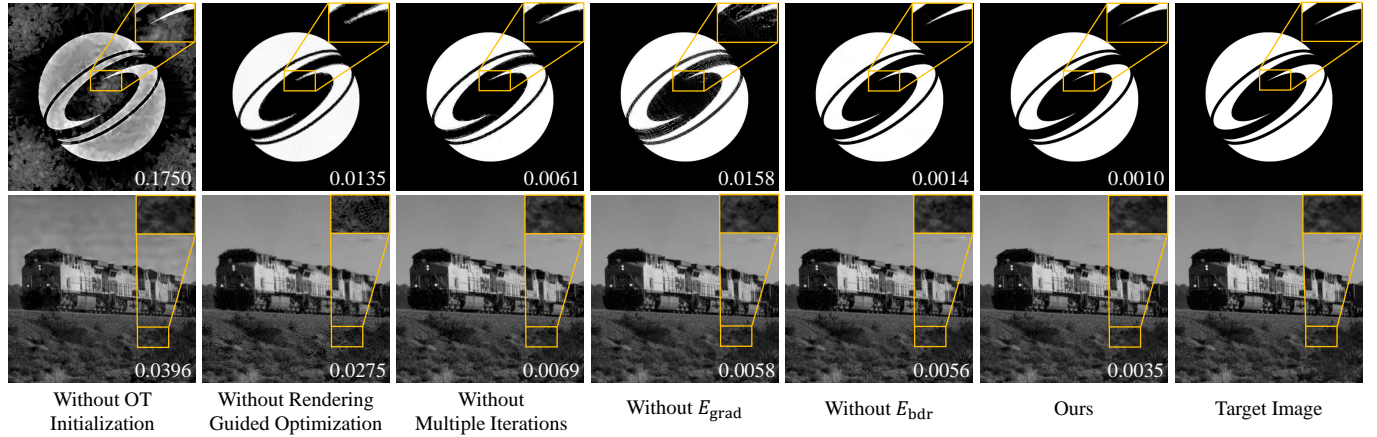| Without OT Initialization | Without Rendering Guided Optimization | Without Multiple Iterations | Without E_{grad} | Without E_{bdr} | Ours | Target Image |

Fig. 10. Comparison between different variants of our method, with some algorithmic components removed. The number in the lower right corner of each image represents the MAE with the target image.

To verify the necessity of rendering guided optimization, we attempt iterative optimization using only OT initialization. The results are shown in the column "Without Rendering Guided Optimization" of Fig. 10. Although this approach achieves the black background for the SIGGRAPH logo, it fails to recover the sharp boundaries of the white regions fully. For the grayscale image, undesirable black gaps emerge in some parts of the image. Both issues are due to the lack of rendering guided optimization, designed to preserve fine details from the target image.

In the column "Without Multiple Iterations", we investigate the impact of iterative optimization by only performing a single round of OT followed by six rendering-based optimization rounds (i.e., using only one OT while significantly increasing the iteration count of rendering-based optimization) for each mesh resolution. This leads to higher MAE values and lower quality of the resulting image, because an optimization trajectory may encounter multiple local minima, which requires more than a single OT correction to arrive at a desirable solution. In comparison, our approach of alternating between these two procedures (as shown in the column "Ours") can
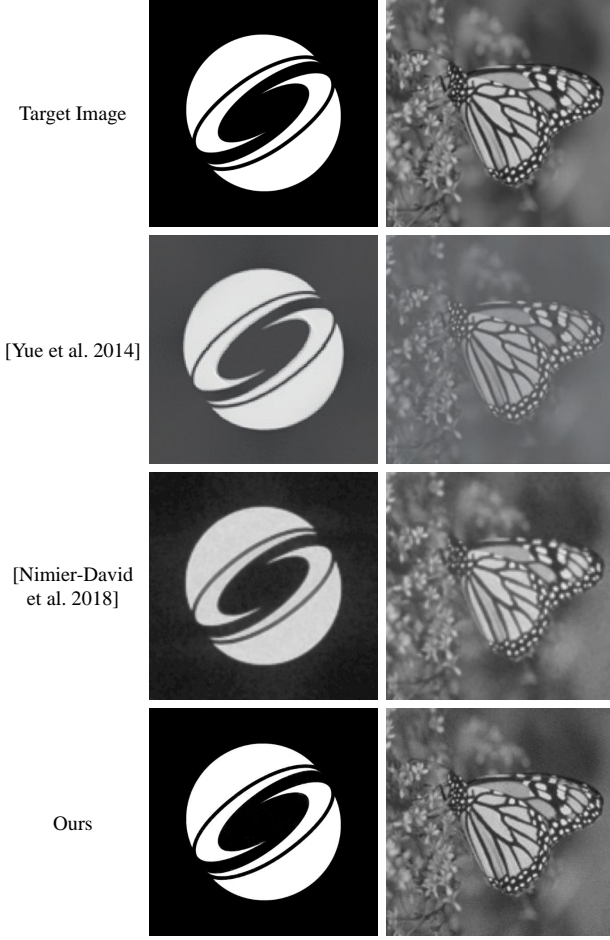
Fig. 11. Comparison with [Yue et al. 2014] and [Nimier-David et al. 2019]. The results of [Yue et al. 2014] are taken from their paper, while the results of [Nimier-David et al. 2019] are generated using Mitsuba 3.
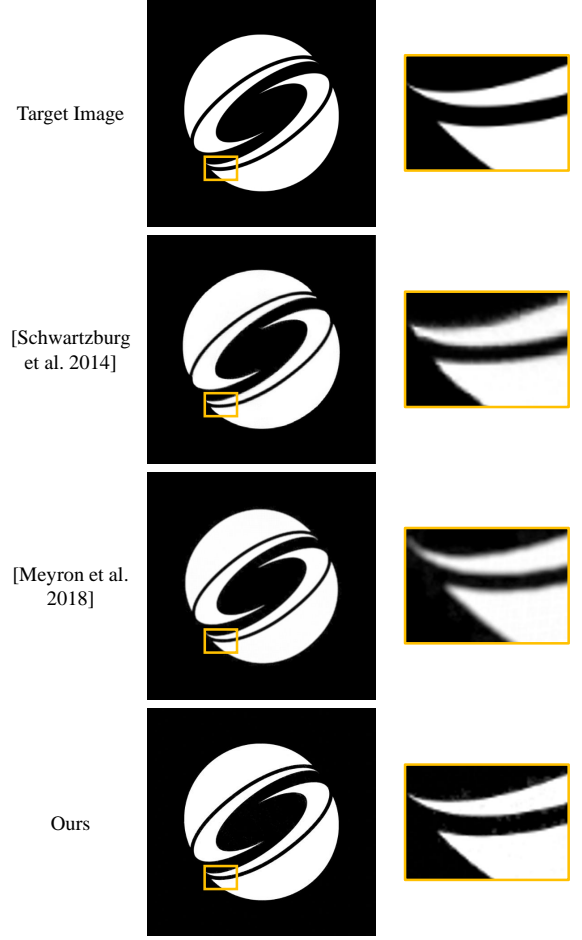

Fig. 12. Comparison with [Schwartzburg et al. 2014] and [Meyron et al. 2018] using results from their papers. Some areas are enlarged to better show the details.

fully leverage their strengths, continually updating and refining the current solution, allowing us to achieve a rendered image almost identical to the target.

We also verify the necessity of the target function terms in our optimization formulation (18). In the column "Without E_{grad}", we perform the optimization without the term E_{grad}. This results in a weaker capability to preserve image features and causes an increase in MAE, which is particularly noticeable in high-contrast areas. In the column "Without E_{bdr}", we remove the term E_{bdr} from the target function, which also results in higher MAE values. This is because the lack of E_{bdr} allows the optimizer to move some lights out of the imaging area in order to reduce other non-image terms in the target function, which can cause deviation from the target image. Finally, later in Sec. 4.4 and Fig. 16, we also demonstrate the necessity of the term E_{smooth} for ensuring the quality of the physical results.

4.3 Comparison with existing methods

In Figs. 11, 12 and 13, we compare our method with existing methods for caustics design, including [Yue et al. 2014], [Schwartzburg et al. 2014], [Meyron et al. 2018] and [Nimier-David et al. 2019], using the rendered images of the results. For a fair comparison, the images of our method are all generated using LuxCoreRender rather than our rendering model. The images of [Yue et al. 2014], [Schwartzburg et al. 2014] and [Meyron et al. 2018] are taken from their papers, while the images of [Nimier-David et al. 2019] are generated using Mitsuba 3[2].

Fig. 11 compares our method with [Yue et al. 2014] and [Nimier-David et al. 2019]. Unlike our approach, both methods fail to produce rendered images with entirely black backgrounds effectively. For a grayscale target image, our method also achieves higher fidelity in the rendered image.

---

[2]We follow the example given in https://mitsuba.readthedocs.io/en/stable/src/inverse_rendering/caustics_optimization.html.
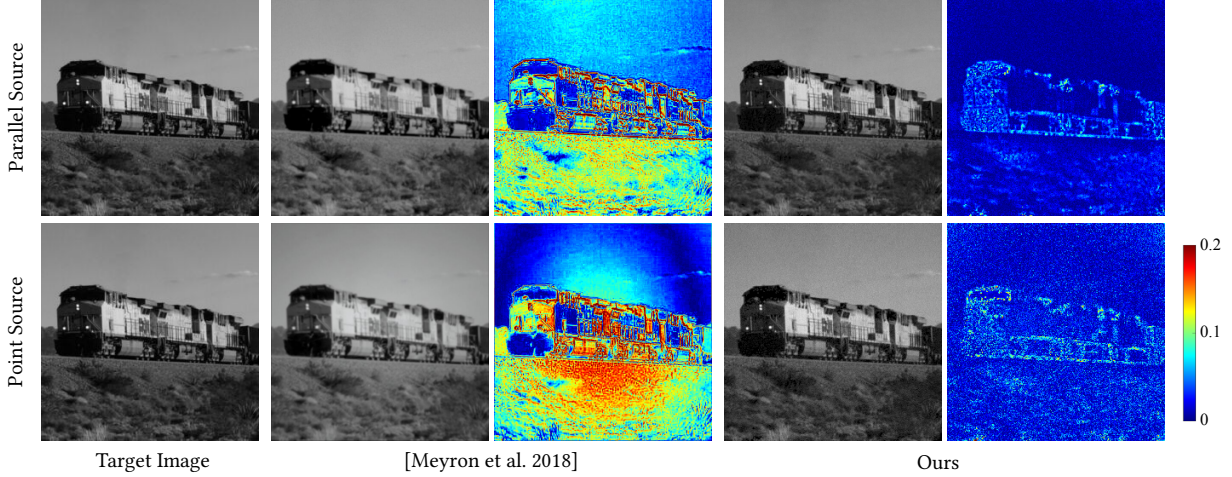
Fig. 13. Comparison with [Meyron et al. 2018]. The color coding shows pixel-wise absolute error (Eq. (27)) compared to the target image.

We also compare our work with [Schwartzburg et al. 2014] and [Meyron et al. 2018] in Fig. 12 using high-contrast target images. Although all three methods produce results of impressive quality, a closer examination reveals both [Schwartzburg et al. 2014] and [Meyron et al. 2018] produce narrower gaps between the two white regions than the target image. This is potentially due to accumulated errors through the process of first deriving intermediate features from the target image and then reconstructing the lens surface mesh from these features. Thanks to our differentiable rendering model and end-to-end optimization directly driven by the image difference, our approach can more effectively fine-tune the surface to match the target caustics, avoiding such discrepancies. Fig. 13 further compares our method with [Meyron et al. 2018] on a grayscale target image using both a parallel light source and a point light source, and visualizes their pixel errors compared to the target. Our results are notably closer to the target under both light sources. The difference between the two methods is most notable in the lower part of the images where there are rich details. The results from [Meyron et al. 2018] smooth out some details and cause large pixel errors in the region. In contrast, our method can preserve the details thanks to the gradient term in our optimization formulation.

A comprehensive quantitative comparison is presented in Table 2. In addition to the MAE (Eq. (26)), we have also evaluated the Structural Similarity Index Measure (SSIM) [Wang et al. 2004], which serves as a perceptual quality metric. SSIM measures the similarity between two images based on their structural information, luminance, and contrast. We use the target image as the reference for computing SSIM, with higher values indicating better perceived image quality and a maximum value of 1 corresponding to an exact match. The SSIM is calculated using the scikit-image library [van der Walt et al. 2014] with its default parameter values. The results from [Yue et al. 2014] are not included in Table 2, as we are unable to obtain the original result images to perform precise alignment with the target image for a fair comparison. The experimental results demonstrate that our method significantly outperforms existing approaches across both evaluation metrics, indicating a closer

Table 2. Quantitative comparison with existing methods using the MAE metric in Eq. (26) and the SSIM [Wang et al. 2004] with respect to the target image. The results in the last row are based on refraction of a point light source, while the others are based on refraction of parallel light.

	MAE $(\times 10^{-3})\downarrow$ | SSIM$\uparrow$			
Image	[Schwartzburg et al. 2014]	[Meyron et al. 2018]	[Nimier-David et al. 2019]	Ours
SIGGRAPH	27.71 | 0.829	/	178.9 | 0.200	**2.873** | **0.964**
Einstein	25.10 | 0.773	/	64.56 | 0.124	**12.32** | **0.840**
Train	/	49.52 | 0.825	45.13 | 0.717	**13.18** | **0.969**
Train (pt.)	/	65.54 | 0.849	/	**23.21** | **0.938**

match to the target image in terms of both pixel-wise accuracy and overall perceptual quality.

4.4 Physical prototypes

We showcase several physical prototypes in Fig. 1, Fig. 14, Fig. 15 and Fig. 16. Our supplementary video also includes their demonstration. All models are fabricated in acrylic (IOR: 1.49) using a JDGR100 milling machine in 3-axis mode. After rough milling, the final pass is done using a 0.4mm ball-end mill with 0.06mm pitch at 23,000 RPM. Afterward, the pieces are polished manually using polishing wax and a cloth wheel. For all models with a parallel light source, we use a white LED torch placed approximately 10 meters away from the lens as the light source. For the model with a point light course, we use a small white spherical LED (OSRAM OSLON SSL 150) with a diameter of about 3mm and positioned at 60cm from the lens. It should be noted that such physical light sources inevitably deviate from the strictly uniform parallel light and the Lambertian point light that are assumed by our formulation, which can cause differences between the simulated rendering and the physical results.

Fig. 1 and Fig. 14 show the rendering results of a few models created by our method, as well as caustic images produced by their physical prototypes. To facilitate comparison with the rendered

Parallel Light

Parallel Light

Point Light

Physical Prototypes   Rectified Results   Rendered Results
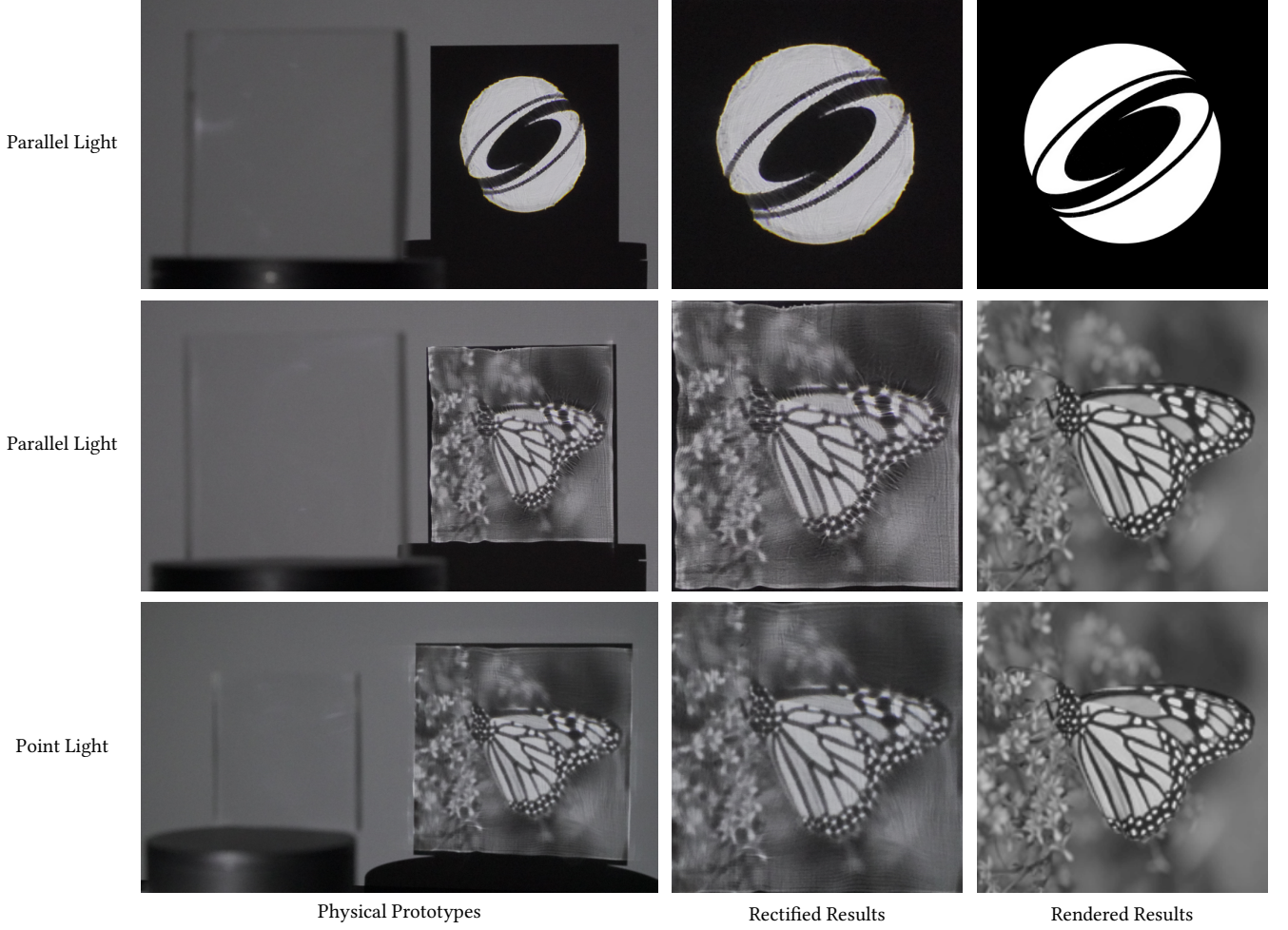
Fig. 14. We fabricate physical prototypes for lens shapes optimized using our method for a parallel light source and a point light source. The photos in the column "Rectified Results" are produced by rectifying the caustic image within the leftmost photos, to provide a frontal view that facilitates comparison with the rendered results. Despite the deviation between our physical lights from ideal parallel and point lights and the imperfections arising from the fabrication process, the real images produced by the prototypes are close to the images generated from our rendering model.

results, we have rectified the photos of the physical caustic images in Fig. 14 to provide a frontal view of the imaging area. Despite the deviation between our physical light sources and their mathematical models, and the slight imperfections due to CNC tolerance and polishing, the physical prototypes still produce images with a strong resemblance to the target.

Fig. 15 compares physical results from our method and the method of [Schwartzburg et al. 2014] on the same target image. Both results are fabricated using the same settings. The caustic images in the real photos in the leftmost column have also been rectified to obtain a front view that facilitates comparison with the target image. We can see that the result of [Schwartzburg et al. 2014] has notable distortions compared to the target image, especially in the silhouette region around the hairs. This is because in [Schwartzburg et al. 2014], the lens shape is not directly optimized based on the rendering image; instead, they first compute a normal field matching

the target image and then reconstruct a mesh from the normals. As their normal field is computed from OT alone, it is not guaranteed to be integrable; as a result, the normals of the reconstructed mesh may deviate from the OT normal field, causing distortions in the real image. In contrast, our optimization is directly driven by the rendering image and achieves better preservation of the target image. In addition, our physical prototype contains fewer fabrication artifacts in the resulting image, thanks to our piecewise smoothness constraints that facilitate fabrication.

Fig. 16 further demonstrates the importance of our piecewise smoothness regularization. For the same Einstein target image, we compute and fabricate two lens shapes using our method with and without the smoothness term while keeping all other components the same. The optimized surface without the smoothness term is notably rougher and contains a large number of small concavities and convexities. Moreover, although both surfaces can produce images
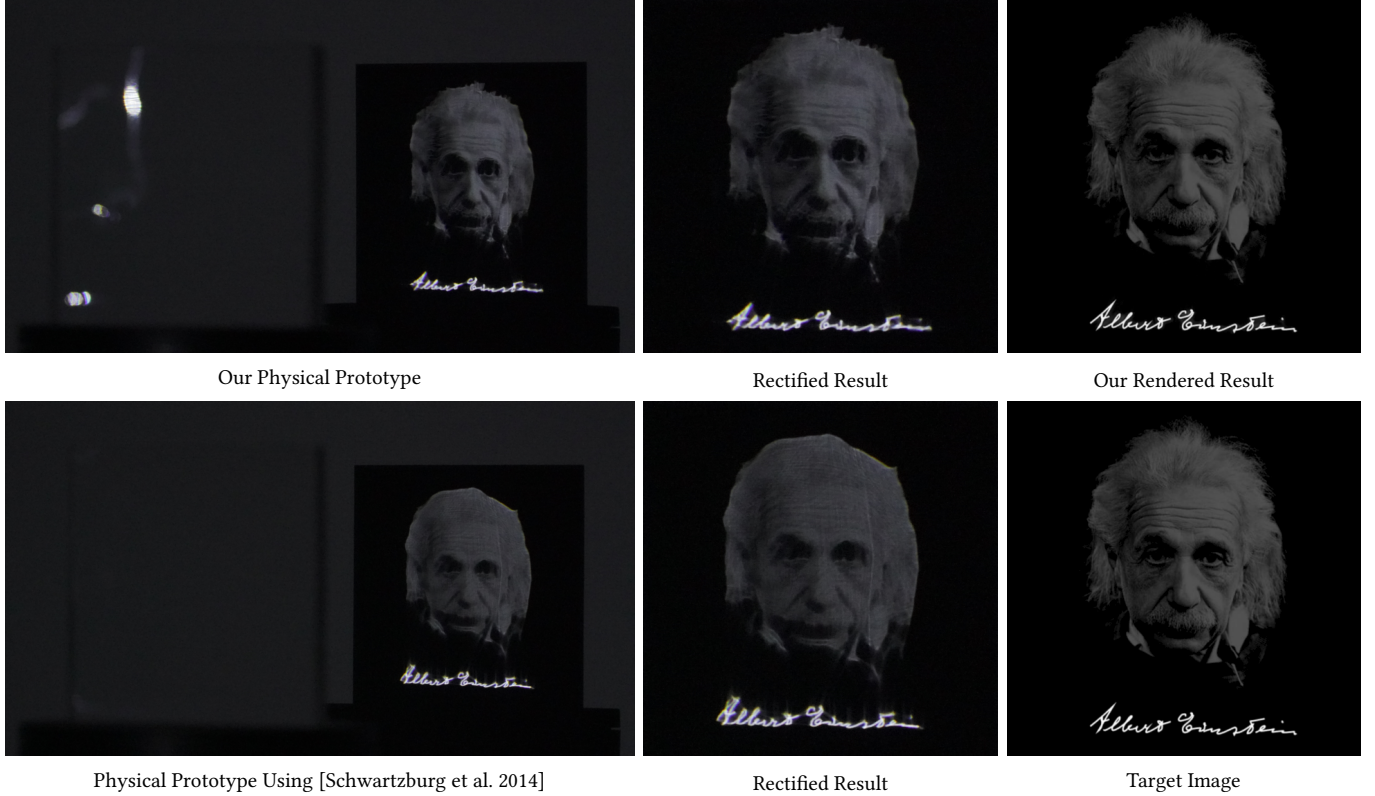
Fig. 15. Comparison of physical prototypes produced using our method and the method of [Schwartzburg et al. 2014]. The resulting mesh model of [Schwartzburg et al. 2014] was provided by the authors. Both results are physically realized using the same fabrication settings. The photos in the middle column are produced by rectifying the caustic image within the leftmost photos to a frontal view. The real caustic image from the result of [Schwartzburg et al. 2014] exhibits more distortion compared to the target, and contains more fabrication artifacts. Einstein portrait © Magnum/IC photo.

closely matching the target in simulated rendering, the real image from the physical prototype without the smoothness term deviates significantly from the target, while the one with the smoothness term still maintains strong resemblance to the target. This example shows that our piecewise smoothness regularization ensures the optimized surface can be physically fabricated with sufficient accuracy, helping to achieve a desirable final result.

While our method has demonstrated promising results in both simulated renderings and physical prototypes, there are still discrepancies between the two due to various factors in the fabrication process. First, our assumption of ideal light sources (strictly uniform parallel light or Lambertian point light) deviates from the physical light sources used in our experiments, which can lead to differences in the observed caustic patterns. Second, the finite precision of the CNC machine and the use of a spherical milling tool introduce approximations in reproducing the optimized surface geometry. In particular, the milling tool cannot perfectly reproduce sharp concave features, resulting in unintended light leakage between adjacent bright areas in the real caustic image (e.g., the SIGGRAPH Logo example in Fig. 14). Third, the manual polishing process can cause additional distortions due to non-uniform material removal and

over-polishing. For example, in the second row of Fig. 14, over-polishing has altered the surface normals at the boundary of the back surface and caused the refracted light rays to deviate from their intended direction and move toward the interior of the image, resulting in notable black areas along the image boundary. It is important to note that such fabrication imperfections are not accounted for in our optimization pipeline, which assumes an ideal manufacturing process. Incorporating these real-world constraints and limitations into the computational design framework is an important direction for future research, as will be discussed in Section 5.

5 CONCLUSION AND DISCUSSION

We presented an end-to-end optical surface optimization algorithm directly driven by the difference between the rendering result and the target image. To solve this challenging problem, we proposed a hybrid strategy combining optimal transport for global structure alignment and rendering guided optimization for local refinement. To ensure the quality of the final physical result, we also introduce a piecewise smoothness regularization to facilitate CNC milling and polishing of the designed surface. Extensive experiments verified the effectiveness of our method.

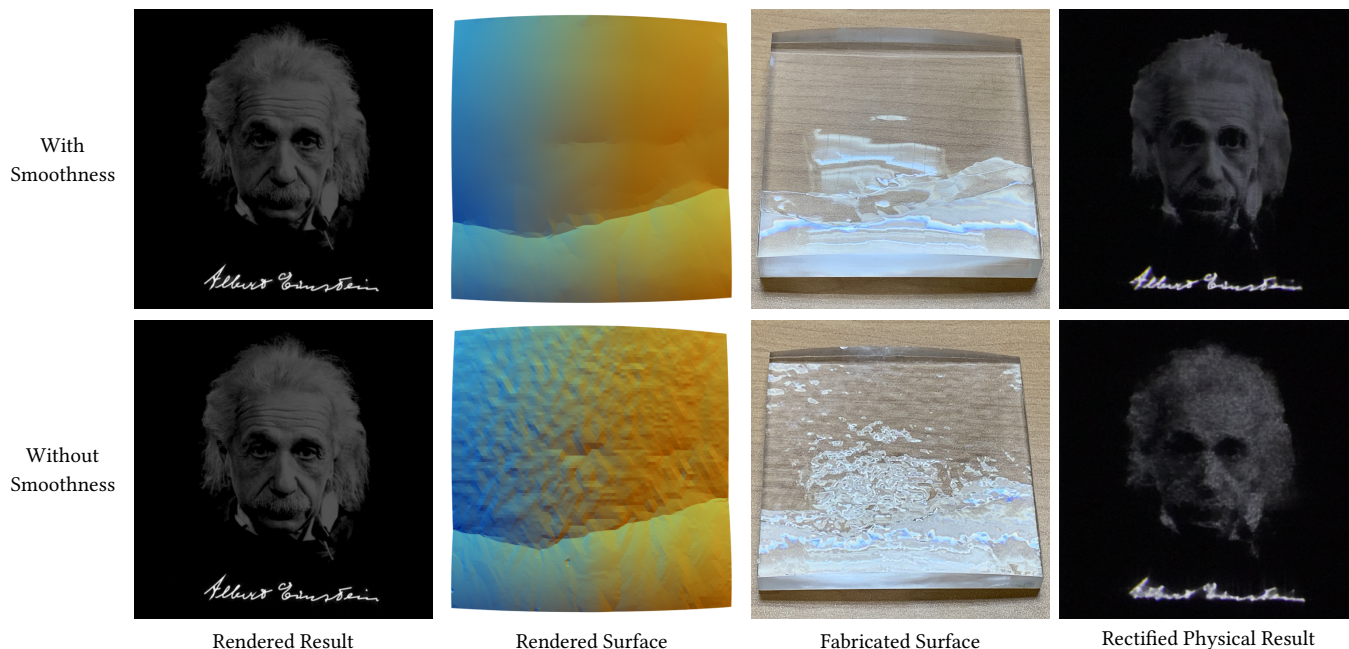| | Rendered Result | Rendered Surface | Fabricated Surface | Rectified Physical Result |

Fig. 16. We optimize and fabricate lens shapes using our method for the same target image, with and without the piecewise smoothness regularization. For the result without the smoothness term, there is a significant deviation between the caustic image rendered from the optimized surface and the real image produced by the physical prototype. Einstein portrait © Magnum/IC photo.

Our method still has some limitations that we plan to address in future work. First, although our piecewise smoothness regularization helps to improve the consistency between the simulated rendering and physical results, imperfections can still arise during fabrication. In the future, we plan to extend our approach to fully consider the impact of fabrication in the optimization, i.e., to formulate an optimization based on fabrication results instead of simulated rendering. One potential approach is to include the milling and polishing processes into our differentiable pipeline, and jointly optimize the lens shape and the CNC tool path to reduce artifacts in physical prototypes. In addition, the mesh connectivity is fixed in the current optimization pipeline, which limits the expressive ability of the surface to some extent. Therefore, dynamic update of the mesh connectivity is another research direction worth exploring.

ACKNOWLEDGMENTS

REFERENCES

Jordan Budhu and Yahya Rahmat-Samii. 2019. A novel and systematic approach to inhomogeneous dielectric lens design based on curved ray geometrical optics and particle swarm optimization. *IEEE Transactions on Antennas and Propagation* 67, 6 (2019), 3657–3669.

Jin-Jia Chen, Te-Yuan Wang, Kuang-Lung Huang, Te-Shu Liu, Ming-Da Tsai, and Chin-Tang Lin. 2012. Freeform lens design for LED collimating illumination. *Optics express* 20, 10 (2012), 10984–10995.

Gerwin Damberg, James Gregson, and Wolfgang Heidrich. 2016. High brightness HDR projection using dynamic freeform lensing. *ACM Transactions on Graphics (TOG)* 35, 3 (2016), 1–11.

Gerwin Damberg and Wolfgang Heidrich. 2015. Efficient freeform lens optimization for computational caustic displays. *Opt. Express* 23, 8 (Apr 2015), 10224–10232. https://doi.org/10.1364/OE.23.010224

Yi Ding, Xu Liu, Zhen-rong Zheng, and Pei-fu Gu. 2008. Freeform LED lens for uniform illumination. *Optics express* 16, 17 (2008), 12958–12966.

Manfredo P. do Carmo. 1976. *Differential Geometry of Curves and Surfaces.* Prentice-Hall.

Manuel Finckh, Holger Dammertz, and Hendrik PA Lensch. 2010. Geometry construction from caustic images. In *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part V 11.* Springer, 464–477.

Paul W Holland and Roy E Welsch. 1977. Robust regression using iteratively reweighted least-squares. *Communications in Statistics-theory and Methods* 6, 9 (1977), 813–827.

Changwon Jang, Olivier Mercier, Kiseung Bang, Gang Li, Yang Zhao, and Douglas Lanman. 2020. Design and fabrication of freeform holographic optical elements. *ACM Transactions on Graphics (TOG)* 39, 6 (2020), 1–15.

LV Kantorovich. 1942. On translation of mass (in Russian), C R. In *Doklady. Acad. Sci. USSR*, Vol. 37. 199–201.

Rudolf Kingslake. 1989. *A history of the photographic lens.* Academic press.

Thomas Kiser, Michael Eigensatz, Minh Man Nguyen, Philippe Bompas, and Mark Pauly. 2013. Architectural caustics—controlling light with geometry. In *Advances in architectural geometry 2012.* Springer, 91–106.

Bruno Lévy. 2015. A Numerical Algorithm for L_2 Semi-Discrete Optimal Transport in 3D. *ESAIM: Mathematical Modelling and Numerical Analysis-Modélisation Mathématique et Analyse Numérique* 49, 6 (2015), 1693–1715.

Dong C Liu and Jorge Nocedal. 1989. On the limited memory BFGS method for large scale optimization. *Mathematical programming* 45, 1-3 (1989), 503–528.

Shichen Liu, Tianye Li, Weikai Chen, and Hao Li. 2019. Soft rasterizer: A differentiable renderer for image-based 3d reasoning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision.* 7708–7717.

Matthew M Loper and Michael J Black. 2014. OpenDR: An approximate differentiable renderer. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VII 13.* Springer, 154–169.

Quentin Mérigot. 2011. A multiscale approach to optimal transport. In *Computer Graphics Forum*, Vol. 30. Wiley Online Library, 1583–1592.

Jocelyn Meyron, Quentin Mérigot, and Boris Thibert. 2018. Light in power: a general and parameter-free algorithm for caustic design. *ACM Transactions on Graphics (TOG)* 37, 6 (2018), 1–13.

Gaspard Monge. 1781. Mémoire sur la théorie des déblais et des remblais. *Mem. Math. Phys. Acad. Royale Sci.* (1781), 666–704.

Merlin Nimier-David, Delio Vicini, Tizian Zeltner, and Wenzel Jakob. 2019. Mitsuba 2: A Retargetable Forward and Inverse Renderer. *Transactions on Graphics (Proceedings of SIGGRAPH Asia)* 38, 6 (Dec. 2019). https://doi.org/10.1145/3355089.3356498

Joseph o'Rourke. 1998. *Computational Geometry in C* (2nd ed.). Cambridge University Press.

Marios Papas, Wojciech Jarosz, Wenzel Jakob, Szymon Rusinkiewicz, Wojciech Matusik, and Tim Weyrich. 2011. Goal-based caustics. In *Computer Graphics Forum*, Vol. 30. Wiley Online Library, 503–511.

Yifan Peng, Xiong Dun, Qilin Sun, and Wolfgang Heidrich. 2017. Mix-and-match holography. *ACM Trans. Graph.* 36, 6 (2017), 191–1.

Szymon Rusinkiewicz. 2004. Estimating curvatures and their derivatives on triangle meshes. In *Proceedings. 2nd International Symposium on 3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004*. IEEE, 486–493.

Martin Šarbort and Tomáš Tyc. 2012. Spherical media and geodesic lenses in geometrical optics. *Journal of Optics* 14, 7 (2012), 075705.

Yuliy Schwartzburg, Romain Testuz, Andrea Tagliasacchi, and Mark Pauly. 2014. High-contrast computational caustic design. *ACM Transactions on Graphics (TOG)* 33, 4 (2014), 1–11.

Pengfei Shen, Ruizeng Li, Beibei Wang, and Ligang Liu. 2023. Scratch-based Reflection Art via Differentiable Rendering. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2023)* 42, 4 (2023), 1–12.

Michael Spivak. 1999. *A Comprehensive Introduction to Differential Geometry* (3rd ed.). Vol. 2. Publish or Perish.

Qilin Sun, Congli Wang, Fu Qiang, Dun Xiong, and Heidrich Wolfgang. 2021. End-to-end complex lens design with differentiable ray tracing. *ACM Trans. Graph* 40, 4 (2021), 1–13.

The CGAL Project. 2023. *CGAL User and Reference Manual* (5.5.2 ed.). CGAL Editorial Board. https://doc.cgal.org/5.5.2/Manual/packages.html

Kenji Tojo, Ariel Shamir, Bernd Bickel, and Nobuyuki Umetani. 2023. Stealth Shaper: Reflectivity Optimization as Surface Stylization. In *ACM SIGGRAPH 2023 Conference Proceedings*. 1–10.

Stéfan van der Walt, Johannes L. Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D. Warner, Neil Yager, Emmanuelle Gouillart, Tony Yu, and the scikit-image contributors. 2014. scikit-image: image processing in Python. *PeerJ* 2 (6 2014), e453. https://doi.org/10.7717/peerj.453

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

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. 2004. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Transactions on Image Processing* 13, 4 (2004), 600–612.

Tim Weyrich, Pieter Peers, Wojciech Matusik, and Szymon Rusinkiewicz. 2009. Fabricating microgeometry for custom surface reflectance. *ACM Transactions on Graphics (TOG)* 28, 3 (2009), 1–6.

Kang Wu, Renjie Chen, Xiao-Ming Fu, and Ligang Liu. 2022. Computational mirror cup and saucer art. *ACM Transactions on Graphics (TOG)* 41, 5 (2022), 1–15.

Rengmao Wu, Peng Liu, Yaqin Zhang, Zhenrong Zheng, Haifeng Li, and Xu Liu. 2013. A mathematical model of the single freeform surface design for collimated beam shaping. *Optics express* 21, 18 (2013), 20974–20989.

Yonghao Yue, Kei Iwasaki, Bing-Yu Chen, Yoshinori Dobashi, and Tomoyuki Nishita. 2014. Poisson-based continuous surface generation for goal-based caustics. *ACM Transactions on Graphics (TOG)* 33, 3 (2014), 1–7.

Juyong Zhang, Yuxin Yao, and Bailin Deng. 2022. Fast and Robust Iterative Closest Point. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44, 7 (2022), 3450–3466.

Shuang Zhao, Kai Wang, Fei Chen, Zong Qin, and Sheng Liu. 2013. Integral freeform illumination lens design of LED based pico-projector. *Applied Optics* 52, 13 (2013), 2985–2993.

Xiangbing Zhu, Qian Zhu, Han Wu, and Chun Chen. 2013. Optical design of LED-based automotive headlamps. *Optics & Laser Technology* 45 (2013), 262–266.

Zhengbo Zhu, Yiming Yan, Shili Wei, Zichao Fan, and Donglin Ma. 2019. Compact freeform primary lens design based on extended Lambertian sources for liquid crystal display direct-backlight applications. *Optical Engineering* 58, 2 (2019), 025108–025108.

Qiang Zou. 2021. Robust and efficient tool path generation for machining low-quality triangular mesh surfaces. *International Journal of Production Research* 59, 24 (2021), 7457–7467.

A DERIVATION OF REFRACTION DIRECTION

Given the incident ray direction $\mathbf{a}$ and the surface normal $\mathbf{n}$ that are both unit vectors, we want to compute a refracted ray direction vector $\bar{\mathbf{b}}$ such that the vector

$$\mathbf{b}' = \bar{\mathbf{b}} - \mathbf{n}$$

is orthogonal to $\mathbf{n}$ (see inset figure).

We first note that the vector

$$\mathbf{a}' = \mathbf{a} - (\mathbf{a} \cdot \mathbf{n})\mathbf{n}$$

is parallel to $\mathbf{b}'$, and the ratio between their norms satisfies

$$\frac{\|\mathbf{b}'\|}{\|\mathbf{a}'\|} = \frac{\tan \beta}{\sin \alpha}$$

since $\mathbf{a}$ and $\mathbf{n}$ are both unit vectors. Using Snell's law, we have $\sin \beta = \eta \sin \alpha$. Therefore,

$$\mathbf{b}' = \frac{\tan \beta}{\sin \alpha}\mathbf{a}' = \frac{\sin \beta}{\sin \alpha \cos \beta}\mathbf{a}' = \frac{\eta}{\cos \beta}\mathbf{a}'.$$

Thus, we have

$$\bar{\mathbf{b}} = \mathbf{n} + \mathbf{b}' = \mathbf{n} + \frac{\eta}{\cos \beta}\mathbf{a}' = \frac{\mathbf{n}\cos \beta + \eta \mathbf{a}'}{\cos \beta}.$$

We can ignore the scaling factor in the denominator and simply use

$$\mathbf{b} = \mathbf{n}\cos \beta + \eta \mathbf{a}' = \mathbf{n}\cos \beta + \eta(\mathbf{a} - (\mathbf{a} \cdot \mathbf{n})\mathbf{n}) \tag{28}$$

to represent the refracted ray direction. Moreover, $\cos \beta$ can be computed from Snell's law as:

$$\cos \beta = \sqrt{1 - \sin^2 \beta} = \sqrt{1 - \eta^2 \sin^2 \alpha} \\ = \sqrt{1 + \eta^2(\cos^2 \alpha - 1)} = \sqrt{1 + \eta^2((\mathbf{n} \cdot \mathbf{a})^2 - 1)}. \tag{29}$$

Substituting Eq. (29) into Eq. (28), we have

$$\mathbf{b} = \mathbf{n}\sqrt{1 + \eta^2((\mathbf{n} \cdot \mathbf{a})^2 - 1)} + \eta(\mathbf{a} - (\mathbf{a} \cdot \mathbf{n})\mathbf{n}). \tag{30}$$

B PROOF OF NO RE-INTERSECTION BETWEEN REFRACTED LIGHT RAYS AND THE LENS SURFACE

In the following, we show that the target function term f_{tir} in Eq. (9) ensures that an outgoing refracted ray from the back surface of the lens will not intersect with the surface again.

Recall that the back surface of the lens is represented as a height field over a planar domain U that is identified with the front surface, and the light rays hitting the back surface are orthogonal to U. For any point $\mathbf{p}$ on the back surface, let l_{in} and l_{out} be the incoming and outgoing light rays, respectively. If l_{out} is parallel to l_{in}, then l_{out} is also orthogonal to U, and $\mathbf{p}$ will be the only intersection point between l_{out} and the back surface because the back surface is a height field over U.

If l_{out} is not parallel to l_{in}, then any intersection point between l_{out} and the back surface must lie on the intersection curve C between
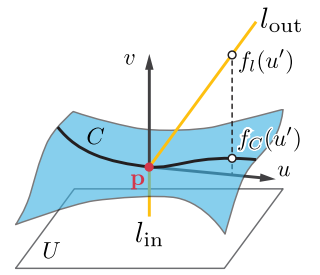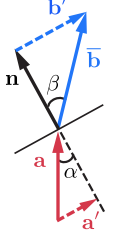
the back surface and the plane P spanned by l_{in} and l_{out}. In the following, we will show that $\mathbf{p}$ is the only intersection point between l_{out} and C, and so it is also the only intersection point between l_{out} and the back surface. To do so, we first construct 2D coordinate system (u, v) within the plane P such that (see inset figure):

- the origin of the coordinate system is at $\mathbf{p}$;
- the u-axis is parallel to the front surface, while the v-axis is parallel to l_{in} and points to the receptive plane;
- the ray l_{out} is represented with an equation

$$v = f_l(u) = Ku, \qquad u \geq 0, \tag{31}$$

where $K > 0$ is the slope of l_{out}.

Within this coordinate system, the curve C can be represented via

$$v = f_C(u)$$

where $f_C(0) = 0$, and we only need to show that there is no intersection between l_{out} and C on the interval $u \in (0, +\infty)$.

To this end, we note that due to the term f_{tir} in Eq. (9), the unit normal vector $\mathbf{n}$ at any point on the back surface must satisfy

$$(\mathbf{n} \cdot \mathbf{a})^2 > 1 - \eta^{-2}, \tag{32}$$

where $\mathbf{a}$ is the unit vector for the incoming light direction and coincides with the $+v$ axis direction in the 2D coordinate system mentioned above. As the back surface is a height-field surface, we must have $\mathbf{n} \cdot \mathbf{a} > 0$. Therefore, Eq. (32) implies that

$$\mathbf{n} \cdot \mathbf{a} > \sqrt{1 - \eta^{-2}},$$

meaning that the angle between the back surface normal and the vector $\mathbf{a}$ must be smaller than a bound

$$\theta_{\max} = \arccos(\sqrt{1 - \eta^{-2}}) = \arcsin(\eta^{-1}) \tag{33}$$

at any point on the back surface. As a result, at any point on the back surface, the angle θ' from any line in its tangent plane to the front surface plane must satisfy

$$0 \leq \theta' < \theta_{\max}. \tag{34}$$

Moreover, for the local height function $f_C(u)$ of the curve C, the derivative $f'_C(u)$ satifies

$$|f'_C(u)| = \tan(\theta_u)$$

where θ_u is the angle between the tangent line of C at u and the front surface plane. Then, Eq. (34) means that

$$|f'_C(u)| < \tan(\theta_{\max}).$$

As a result, for any $u' > 0$, we have

$$f_C(u') = f_C(0) + \int_0^{u'} f'_C(u) du \leq \int_0^{u'} |f'_C(u)| du < u' \cdot \tan(\theta_{\max}). \tag{35}$$

Meanwhile, at any point on the back surface, the angle between the refracted ray direction and the back surface normal must be no larger than $\pi/2$, since the term f_{tir} ensures that there is no total internal reflection. Therefore, the angle θ_{out} between the refracted ray and the front surface plane must satisfy

$$\theta_{\text{out}} > \theta_{\max}. \tag{36}$$

As a result, the slope K of ray l_{out} in Eq. (31) must satisfy

$$K > \tan(\theta_{\max}),$$

so we have

$$f_l(u') = Ku' > u' \cdot \tan(\theta_{\max}) \tag{37}$$

for any $u' > 0$. Then, using Eqs. (35) and (37), we have

$$f_l(u') > f_C(u'), \qquad \forall u' > 0.$$

In other words, away from the point $\mathbf{p}$, the outgoing ray l_{out} is always above the curve C. Therefore, $\mathbf{p}$ is the only intersection point between l_{out} and the back surface.

C EVALUATION OF ENERGY $H(\mathbf{w})$

The main difficulty in evaluating $H(\mathbf{w})$ comes from the integral

$$\int_{\mathbf{s} \in \mathcal{P}_i^{\mathbf{w}}} (\|\mathbf{x} - \mathbf{c}'_i\|^2 - w_i) \widetilde{\Phi}_{\mathbf{s}} d\mathbf{s}$$

$$= \int_{\mathbf{s} \in \mathcal{P}_i^{\mathbf{w}}} \|\mathbf{x} - \mathbf{c}'_i\|^2 \widetilde{\Phi}_{\mathbf{s}} d\mathbf{s} - w_i \int_{\mathbf{s} \in \mathcal{P}_i^{\mathbf{w}}} \widetilde{\Phi}_{\mathbf{s}} d\mathbf{s}.$$

Since the other components in $H(\mathbf{w})$ are trivial to evaluate, in the following we focus on the computation of two integrals,

$$\int_{\mathbf{s} \in \mathcal{P}_i^{\mathbf{w}}} \widetilde{\Phi}_{\mathbf{s}} d\mathbf{s} \tag{38}$$

and

$$\int_{\mathbf{s} \in \mathcal{P}_i^{\mathbf{w}}} \|\mathbf{x} - \mathbf{c}'_i\|^2 \widetilde{\Phi}_{\mathbf{s}} d\mathbf{s}. \tag{39}$$

For Eq. (38), we compute the integral by accumulating the contribution from each pixel:

$$\int_{\mathbf{s} \in \mathcal{P}_i^{\mathbf{w}}} \widetilde{\Phi}_{\mathbf{s}} d\mathbf{s} = \sum_{j=1}^{n_p} \left(\int_{\mathbf{s} \in \mathcal{P}_i^{\mathbf{w}} \cap p^j} \frac{\widetilde{\Phi}^j}{A_p} d\mathbf{s} \right)$$

$$= \sum_{j=1}^{n_p} \frac{\widetilde{\Phi}^j}{A_p} \left(\int_{\mathbf{s} \in \mathcal{P}_i^{\mathbf{w}} \cap p^j} d\mathbf{s} \right) \tag{40}$$

$$= \sum_{j=1}^{n_p} \frac{\widetilde{\Phi}^j A(\mathcal{P}_i^{\mathbf{w}} \cap p^j)}{A_p},$$

where $A(\cdot)$ denotes the area. Note that since both $\mathcal{P}_i^{\mathbf{w}}$ and p^j are both convex polygons, if their intersection is not empty then $\mathcal{P}_i^{\mathbf{w}} \cap p^j$ must be a convex polygon. It is easy to determine the intersection area $A(\mathcal{P}_i^{\mathbf{w}} \cap p^j)$ in this case.

Similarly, for Eq. (39) we accumulate the contributions from all pixels:

$$\int_{\mathbf{s} \in \mathcal{P}_i^{\mathbf{w}}} \|\mathbf{x} - \mathbf{c}'_i\|^2 \widetilde{\Phi}_{\mathbf{s}} d\mathbf{s}$$

$$= \sum_{j=1}^{n_p} \left(\int_{\mathbf{s} \in \mathcal{P}_i^{\mathbf{w}} \cap p^j} \|\mathbf{x} - \mathbf{c}'_i\|^2 \frac{\widetilde{\Phi}^j}{A_p} d\mathbf{s} \right) \tag{41}$$

$$= \sum_{j=1}^{n_p} \frac{\widetilde{\Phi}^j}{A_p} \int_{\mathbf{s} \in \mathcal{P}_i^{\mathbf{w}} \cap p^j} \|\mathbf{x} - \mathbf{c}'_i\|^2 d\mathbf{s}.$$

Since $\frac{\widetilde{\Phi}^j}{A_p}$ is a given constant for each p^j, we only need to consider the calculation of

$$\int_{\mathbf{s}\in\mathcal{P}_i^{\mathbf{w}}\cap p^j} \left\|\mathbf{x}-\mathbf{c}_i'\right\|^2 d\mathbf{s}. \tag{42}$$

Recall the following fact from the Green's Theorem: if both P and Q have have continuous partial derivatives on the domain D and its boundary L, then

$$\iint_D \left(\frac{\partial Q}{\partial x} - \frac{\partial P}{\partial y}\right) = \oint_{L^+} Pdx + Qdy. \tag{43}$$

We would like to apply this relation to convert the integral in Eq. (42) into a line integral. To this end, we need to find two functions P and Q, such that

$$\frac{\partial Q}{\partial x} - \frac{\partial P}{\partial y} = \left\|\mathbf{x}-\mathbf{c}_i'\right\|^2.$$

A symmetric solution is $Q = x(y-y_0)^2$ and $P = -y(x-x_0)^2$ where $(x_0, y_0) = \mathbf{c}_i'$. Then we can apply Eq. (43) to each domain $\mathcal{P}_i^{\mathbf{w}} \cap p^j$. Note that $\mathcal{P}_i^{\mathbf{w}} \cap p^j$ is a convex polygon if it is not empty. Therefore, we only need to consider the right-hand-side of (43) on each segment of its boundary. Suppose (x_1, y_1) and (x_2, y_2) are two adjacent vertices of this polygon in counterclockwise order, then we have

$$\begin{aligned}
&\int_{y_1}^{y_2} Qdy \\
&= \int_{y_1}^{y_2} x(y-y_0)^2 dy \\
&= \int_{y_1}^{y_2} (ky+b)(y-y_0)^2 dy \\
&= \int_{y_1}^{y_2} ky^3 + (b-2ky_0)y^2 + (ky_0^2 - 2by_0)y + by_0^2 \, dy \\
&= \left(\frac{k}{4}y^4 + \frac{b-2ky_0}{3}y^3 + \frac{ky_0^2 - 2by_0}{2}y^2 + by_0^2 y\right)\bigg|_{y_1}^{y_2} \\
&= \frac{x_2 - x_1}{4}(y_1 + y_2)(y_1^2 + y_2^2) \\
&\quad + \frac{x_1 y_2 - x_2 y_1 - 2y_0(x_2 - x_1)}{3}(y_1^2 + y_2^2 + y_1 y_2) \\
&\quad + \frac{y_0^2(x_2 - x_1) - 2y_0(x_1 y_2 - x_2 y_1)}{2}(y_1 + y_2) \\
&\quad + y_0^2(x_1 y_2 - x_2 y_1).
\end{aligned} \tag{44}$$

Here we make use of the linear relation $x = ky+b$, where $k = \frac{x_2-x_1}{y_2-y_1}$ and $b = \frac{x_1 y_2 - x_2 y_1}{y_2 - y_1}$.

Similarly, we can deduce:

$$\begin{aligned}
&\int_{x_1}^{x_2} Pdx \\
&= \int_{x_1}^{x_2} -y(x-x_0)^2 dx \\
&= \int_{x_1}^{x_2} (mx+c)(x-x_0)^2 dx \\
&= \int_{x_1}^{x_2} mx^3 + (c-2mx_0)x^2 + (mx_0^2 - 2cx_0)x + cx_0^2 \, dx \\
&= \left(\frac{m}{4}x^4 + \frac{c-2mx_0}{3}x^3 + \frac{mx_0^2 - 2cx_0}{2}x^2 + cx_0^2 x\right)\bigg|_{x_1}^{x_2} \\
&= \frac{y_1 - y_2}{4}(x_1 + x_2)(x_1^2 + x_2^2) \\
&\quad + \frac{x_1 y_2 - x_2 y_1 + 2x_0(y_2 - y_1)}{3}(x_1^2 + x_2^2 + x_1 x_2) \\
&\quad + \frac{x_0^2(y_1 - y_2) - 2x_0(x_1 y_2 - x_2 y_1)}{2}(x_1 + x_2) \\
&\quad + x_0^2(x_1 y_2 - x_2 y_1),
\end{aligned} \tag{45}$$

where we make use of the linear relation $-y = mx + c$, with $m = -\frac{y_2-y_1}{x_2-x_1} = -\frac{1}{k}$ and $c = \frac{x_1 y_2 - x_2 y_1}{x_2 - x_1} = \frac{b}{k}$.

To summarize, for computing the integral (42), we apply Eqs. (44) and (45) to each boundary edge of $\mathcal{P}_i^{\mathbf{w}} \cap p^j$ in the counter-clockwise direction and sum their values. In addition, we only consider pixels inside the bounding box of $\mathcal{P}_i^{\mathbf{w}}$, to avoid unnecessary intersection.

D EXTENSION TO REFLECTION

For reflection surfaces, we can also set the starting surface to a plane with a normal $\mathbf{n} = (0, 0, 1)$. When setting the light source, we must also ensure the light does not strike the reflective surface straight on, as this would cause the reflected light to align with the incoming light. In those cases, if a receptive plane is placed in the path of the reflected light, it could block the incoming light. Typically, for parallel light we can set an inclination angle between the incoming light direction and the initial surface normal, and place the receptive plane on the other side of the normal to avoid this issue.

At a surface point with a surface normal $\mathbf{n}$ and an incident light direction $\mathbf{a}$, it is easy to derive the reflected light direction as

$$\mathbf{b} = \mathbf{a} - 2(\mathbf{a}\cdot\mathbf{n})\mathbf{n}.$$

E FURTHER DETAILS OF IMPLEMENTATION

In this section, we present more details of our implementation.

For all examples, we first downsample the target image to a resolution of 32^2 (50×25 for the Calligraphy image) and use the coarsest mesh as the initial stage. The mesh resolution is independent of the image resolution, which means that we can freely choose the mesh resolution based on the tradeoff between optimization time and expressive power. In our examples, we set the mesh resolution to 1, 1.25, or 2.5 times the image resolution depending on our needs. For the butterfly example, we used a mesh with a resolution of 2.5 times the image resolution, which resulted in a very small MAE.

We use 4 threads of an Intel Xeon E5-2690 to compute the power diagram using CGAL, and an NVIDIA GeForce RTX 3090 for all

the optimizations. At the highest resolution, it takes approximately two hours to perform OT, five minutes for correspondence guided update, and an hour for rendering guided optimization.

In E_{edge}, the parameter ν for the Welsch function controls its sensitivity to outliers. The smaller it is, the closer the Welsch function is to the ℓ_0-norm, and the less influential the outliers are to the optimization. For the sharp edges to emerge, it is necessary to set ν to a sufficiently small value. On the other hand, setting a small ν from the beginning could prevent the optimization from improving edge smoothness, as the errors at most edges could be treated as outliers and have little influence on the optimization. To facilitate optimization, we set a relatively large ν at the beginning so that all edges can affect the optimization, then gradually decrease it to a small value to allow sharp edges to emerge. Specifically, we first set the maximum and minimum values $\alpha_{\max}, \alpha_{\min}$ of the ratio

between ν and the typical scale D_e of the residual $\sqrt{h(e_{ij})}$ on an edge. We set $\nu = \alpha_{\max} \cdot D_e$ at the beginning of the optimization. Afterward, when the mesh is subdivided, we decrease ν by a fixed ratio $\gamma = (\frac{\alpha_{\min}}{\alpha_{\max}})^{\frac{1}{k-1}}$ where k is the number of mesh resolutions, such that ν is set to $\alpha_{\min} \cdot D_e$ for the highest resolution.

In our rendering guided optimization, the weights in the optimization target function are set to

$$(\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5) = (1 \times 10^2, 1 \times 10^3, 1 \times 10^{-3}, 2 \times 10^1, 1 \times 10^{-8}).$$

For the correspondence guided update, we set

$$(\gamma_1, \gamma_2, \gamma_3, \gamma_4) = (1 \times 10^0, 1 \times 10^1, 2 \times 10^1, 1 \times 10^{-14}).$$

For the piecewise smoothness term E_{smooth}, we set

$$(\tau_1, \tau_2) = (3 \times 10^0, 2 \times 10^{-1}).$$