# Computational Methods in Freeform Optical Technologies

KEVIN FIGUEROA*[1] AND ZENAS HUANG**[2]

[1] USC Viterbi School of Engineering, Department of Computer Science
[2] USC Viterbi School of Engineering, Department of Industrial & Systems Engineering
[*] Corresponding author: kcfiguer@usc.edu
[**] Corresponding author: zahuang@usc.edu

Compiled May 10, 2020

**Freeform optical technologies are an emerging field of geometrical optics that offer an array of advantages over traditional optical designs due to the greater precision afforded by relaxing constraints such as radial symmetry. This greater flexibility also presents new challenges in the manufacturing of these surfaces. In this project, we survey existing methods in the design of freeform lenses and then explore one approach to freeform surface design by scripting an implementation of the numerical method described by one paper.**

http://dx.doi.org/10.1364/ao.XX.XXXXXX

## 1. INTRODUCTION

From classical spherical forms used in ancient Greece's burning lenses, to more modern aspherical shaped lenses for magnifying imagery with minimized distortions. The shape of optical technologies has been evolving for thousands of years. Today, advances in computational tools and the introduction of Computer-Aided Design (CAD) has allowed research in optical technologies to more closely and discretely investigate novel shaped optical forms; thereby evolving their shape another step further.

By discarding the very simple rule of rotational symmetry, required in past optical forms, research in optical technologies has generalized its manufacturing constraints to allow for a cacophony of new geometric shapes and specialized forms, limited only by the laws of physics. Operating with this more generalized approach, these new Free of form optics, or Freeform Optics, have allowed for improved imaging technology performance with wider angle fields-of-view, lower distortions, improved MTF, better low light performance, overall higher resolution imagery, and even introduced new methodologies for image projection. In this project, we investigate methods for freeform lens design in image projection applications.

## 2. PROBLEM STATEMENT & LITERATURE SURVEY

### A. Problem Statement & Background

The core problem in freeform lens design is to find a means of computing a lens surface given known conditions on the target distribution pattern and light source. In physics and optics, this is known as the problem of Illumination control or the Inverse Refractor design.

A prerequisite to this problem requires a means of finding a way to represent the trajectory of light between two different points. There is a rich literature in physics describing the connection of the EM wave representation of light and its photon ray representation through the Eikonal equation. This nonlinear PDE is a solution to the 3D wave equation and describes wavefronts at different time points and that the integration of these eikonal solutions determine light trajectories. For brevity, we will not delve into the complete physics here and instead refer the reader to [1, 2]. The eikonal equations effectively describe information on all possible ray mappings that can be introduced by an optical surface element, however the important point for us is that there exists rigorous ray-theoretic approximations of light which we can rely on to represent the light paths and that these paths can be accurately numerically approximated for simulation methods since taking the zero limit of wavelength of the phase fronts yields the familiar rays which are normal to the wavefronts. This existence of a ray-theoretic representation of light paths has subsequently allowed for the development of a panoply of ray-mapping based technologies in geometrical optics for freeform lens manufacturing.

Given a monochrome light wave, assumed to satisfy the wave equation, that is expressed as:

$$u(x,y,z) = A(x,y,z)e^{ik\phi(x,y,z)} \qquad (1)$$

The phase of the wave is given by its eikonal $\phi(x,y,z)$ and the goal of illumination control problems is to identify the eikonal through samplings of the irradiance, $I = A^2$, at two different planar locations (i.e. source and target). This can be equivalently considered as the search for a mapping, $Z$, that satisfies the relation of the intensities between the two planes governed by:

$$I_t(x', y') = I(x, y)|J(Z(x, y))| \qquad (2)$$

In this way, the search for the appropriate Eikonal mapping is reduced to the computation of an appropriate optical surface that produces the correct transform of ray mapping from source intensity to target irradiance. This relationship along with assumptions about the paraxial approximation of light and the conservation of energy give rise to a class of mathematical problems known as prescribed Gaussian curvature problems that are modeled by a notoriously nonlinear 2nd-order Monge-Ampere elliptical PDE given in general form as:

$$Det\ D^2 z(x, y) - K_{Gauss}(1 + |Dz(x, y)|^2)^{\frac{n+2}{2}} = 0 \qquad (3)$$

Which tells us that at a point p the local curvature of the surface given by the determinant of the hessian must be given by the product of the hypersurface of dimension n+1 (in the planar source case n = 2) and it's Gaussian Curvature at point p [3, 4].

This sought after hypersurface of freeform lens surface is also known as a Monge Patch, Z, given by:

$$m : U \to \mathbb{R}^3, \quad U \subset \mathbb{R}^2, \quad m(x, y) = (x, y, z(x, y)). \qquad (4)$$

such that the Monge patch uniquely satisfies the given boundary conditions defined by the target distribution assuming all other parameters are constant and has a known Gaussian Curvature given by:

$$K_{Gauss} = \frac{z_{xx} z_{yy} - z_{xy}^2}{1 + z_x^2 + z_y^2}. \qquad (5)$$

which corresponds to the prescribed curvature equation for finding a 3D hypersurface in the n = 2D domain case. The goal then of most computational methods for freeform designs that are premised on the Monge-Ampere framework is to simulate a discrete set of local Monge-Patches that collectively define the optical surface.

In general, the proof of the existence and uniqueness of solutions for Monge-Ampere Boundary Value Problems is regarded as a hard problem in mathematics [5]. But for the Elliptic Monge-Ampere PDE, certain classes of finite-difference numerical schemes that are degenerate elliptic and locally Lipschitz do admit of unique solutions which is true for the case of the single surface quadratic approximation scheme that we attempted to implement. [6].

## B. Literature Review

Here we present a brief review of a few popular methods for designing freeform optical surfaces.

### B.1. Simultaneous Multiple Surfaces

Although typically used for 2D optical surface design, the simultaneous multiple surfaces (SMS) method has been successfully extended as a method of designing freeform optical lenses in 3D . The SMS3D method begins with the choice of an initial surface, typically a supporting Cartesian Oval, which images planar wavefronts onto a set of desired target points. In particular, given a set of rays whose trajectories are normal to a family of surfaces representing the wavefront, then a cartesian oval surface is chosen which satisfies the mapping of these rays to the desired target points. The procedure then proceeds by generating SMS-chains which are a set of successively computed spatial points defined by the intersection of ray trajectories and their normal vectors. Parametrized curves passing through these points called SMS-ribs are then computed and joined together to form the SMS surface that is typically approximated using the well-known Non-uniform Rational B-Spline or NURBS algorithm [7].

Since each of these added surfaces are computed concurrently, the SMS3D method designs a composite optical surface that is comprised of different surface functions. While this method is adaptable and has high degrees of freedom, it's reliance of the initial Cartesian oval positioning and choice of ray mapping mean that it is highly sensitive to the initial lens choice and may be prone to the development of cusp singularities. No existing literature on the computational cost or performance analysis was found on these SMS based methods.[5]

### B.2. Transport Optimization based approaches

Due to the difficulties associated with solving a highly nonlinear PDE such as the Monge-Ampere, another approach to freeform lens design is based on an equivalent reformulation of the original MA PDE as an Optimal Transport problem known as the Monge-Kantorovich Problem. In this framework, the light field that focuses into the prescribed target region is recast as a photon mass transport problem where the cost function corresponds to an L2 distance between a point on the light source region and a point in the target region and energy conservation guarantees that total energy between source and target regions must be the same. [8]

$$\begin{aligned} &\min \iint_\Omega C((x, y), T(x, y)) I(x, y)\, dx\, dy \\ S/T : \quad & T : I(x, y) \to I_t(x', y') \end{aligned} \qquad (6)$$

Where $I(x, y)$ is the initial source intensity distribution and $I_t(x', y')$ is the target irradiance distribution and the cost, C, is a quadratic cost function of the L2 distance between the photon source position to its final target irradiance position. Multiplying this cost by the source intensity and integrating over the domain gives a measure of the total work required for photon transport along the ray trajectory. Thus, the optimal $T^*$ is an Eikonal mapping which minimizes this work integral. The constraints in this transport approach are also described by the irradiance pairings of sampled source to target points, thus making the resultant resolution and performance of this approach dependent on the size of the chosen grid discretization. Then, given the appropriate refractive index of the lens material, n, the height of the optical surface can then be found as a function of the Eikonal ray map, T, at specified points based on the following general relation with u serving as a irradiance position vector:

$$z(u) = \frac{T^*(u)}{(n-1)} \qquad (7)$$

Empirical experiments with this method by [9–11] have shown that this approach to freeform surface computation obtains roughly $O(N^2)$ performance with N the number of sampled points forming the constraint matrix. While this is a popular approach in the literature, most optics work premised on the Transport Problem framework has dealt with reflectors and is predominantly theoretically focused without much indication of explicit constraint formulations or direct application of solution algorithms. [8, 12]

### B.3. Finite Difference Newton Methods for Direct Solution of Elliptical Monge-Ampere

A third popular method is to use finite difference approximations and generate picard iterations to approximate a solution to the elliptical Monge-Ampere PDE through the use of a traditional grid and standard finite difference discretization of Monge-Ampere which has a local linearization expressible as [12]:

$$\begin{aligned} A(z_{xx}z_{yy} - z_{xy}^2) + Bz_{xx} + Cz_{yy} + Dz_{xy} + E = 0 \\ BC : x_t = f_1(x, y, z, z_x, z_y) \\ y_t = f_2(x, y, z, z_x, z_y) \end{aligned} \tag{8}$$

The typical drawback to this approach is sensitivity to the initial solution ansatz and potential for convergence to a sub-optimal local minima [8, 13]. The standard Newton Methods used for solving Elliptic Monge-Ampere PDEs were reported to have approximately quadratic $O(N^2)$ performance with N the size of the chosen discretization grid which is reported to be slightly faster than Gauss-Seidel based approaches [6].

### B.4. On comparison of methods

Since the Optimal Transport and MA PDE finite difference approches are mathematically equivalent problems we find that their performance is roughly comparable with minor variation depending on the transport formulation or the choice of solver algorithm. However, the SMS3D approach does not consider the Monge-Ampere in computing its freeform surface so it would not be reasonable to draw a comparison of its convergence to a solution with the Optimal Transport and MA PDE based approaches.

## 3. METHOD DESCRIPTION & IMPLEMENTATION

For this portion of the project, we chose to write a MATLAB implementation of a specific numerical method that was originally developed by [14]. Here we present a brief summary of the key aspects of this and characterize our approach to the implementation. Our full MATLAB script is provided in the appendix to this report. Our approach is original in that we reproduced this approach to freeform surface generation solely based on the provided mathematical expressions where no source code was provided for the method within the original paper or related any public repositories.

In this method, we consider the simplified case of finding a single freeform surface formulated as the solution to an elliptical Monge-Ampere PDE. Specifically, we consider a simple collimated light beam propagating outwards along the z-axis from a planar light source and define parameters for a target distribution on a plane at a set distance away from our desired lens. In the general case of freeform design the source and target can be any arbitrary kind of grid, but as a simplification we assume a uniform grid for both. The problem is them modeled as a mesh distortion in which the beam is incident on the entrance of our lens at z = 0, which coincides with the source plane and is then refracted by the freeform surface upon exit where the optical surface,$z_L$, located a distance L from the source is a function of our planar source.

Analytically, we can describe the relation of the intensities between two planes as the mapping of light as it leaves our lens surface and its destination on the target plane through a conformal mapping which is encoded via the Jacobian by the following:

$$dS'(x', y') = dS(x, y)|J(x, y)| \tag{9}$$

Where the primed values denote the mesh element of light at the target plane as a function of the source coordinates and the Jacobian is defined as:

$$J = \begin{bmatrix} \frac{\partial f_1(x,y)}{\partial x} & \frac{\partial f_1(x,y)}{\partial y} \\ \frac{\partial f_2(x,y)}{\partial x} & \frac{\partial f_2(x,y)}{\partial y} \end{bmatrix}. \tag{10}$$

with $x_t = f_1$ and $y_t = f_2$ being the target coordinates as a function of the source light coordinates such that the desired unknown lens function $z_L$ can be described in component terms by $f_1$ and $f_2$.

Although in this formulation we consider the mapping from surface to target, due to the bi-directionality of light, we can also consider this as a mapping from source plane to surface with the result that the orientation of our lens changes. Using this relationship, the authors of the method assume a paraxial (i.e. small angle) approximation to Snell's refraction law holds and are able to encode the refractive effect of the lens into the scalar function K:

$$K(z_{Lx}, z_{Ly}) = \frac{n - \sqrt{1 + (z_{Lx}^2 + z_{Ly}^2)(1 - n^2)}}{z_{Lx}^2 + z_{Ly}^2 + 1} \tag{11}$$

where n denotes a scalar ratio that accounts for the refractive indices of input and output media of the lens surface and the $z_L$ values are partial derivatives of our lens surface. A lengthy derivation then yields a the following expression for the lens surface as a Monge-Ampere type PDE:

$$|z_{Lxx}z_{Lyy} - z_{Lxy}^2| = \frac{I(x,y)}{I_T(f_1, f_2)} \frac{\sqrt{1 + (z_{Lx} + z_{Ly})(1 - n^2)}}{K(z_{Lx}, z_{Ly})^2} \tag{12}$$

The left side of equation (12) is the familiar Hessian and the right side is then discretized and compacted in a variable $g$ given by:

$$g^{(i,j)} = \frac{I(x_i, y_j)}{I_t(f_1(x_i, y_i), f_2(x_i, y_i))} \frac{\sqrt{1 + (z_{Lx} + z_{Ly})(1 - n^2)}}{K(z_{Lx}, z_{Ly})^2} \tag{13}$$

where the ratio between the unknown irradiance functions $I$ and $I_t$ define the conformal mapping between the distortion of light from source to target and serves to represent the Jacobian of our surface. Finally, the authors then convert the lens surface Monge-Ampere into a quadratic through a finite difference approximation scheme and arrive at the following surface formula by solving the quadratic representation and selecting the negative root to guarantee convexity. This yields the following expression (14):

$$z_L^{(i,j)} = \frac{av_1^{(i,j)} + av_1^{(i,j)} - \sqrt{(av_1^{(i,j)} + av_1^{(i,j)})^2 + (\frac{av_3^{(i,j)} - av_4^{(i,j)}}{2})^2 + h^4 g^{(i,j)}}}{2}$$

This expression is then solved on a uniform square grid using a compact neighborhood stencil with each of the *av* variables in (14) representing an arithmetic average of neighboring grid points used to account for the second derivatives of the $z_L$ surface and account for the local patch curvature described by the Hessian. The authors also note that this averaging method was also chosen in order to arrive at a computationally tractable quadratic expression for the lens surface. The second derivatives are related to these *av* averages by the following:

$$z_{Lxx} = \frac{2(av_1 - z_L)}{h^2}, z_{Lyy} = \frac{2(av_2 - z_L)}{h^2}, z_{Lxy} = \frac{(av_3 - av_4)}{2h^2} \qquad (15)$$

Finally, to start the design an initial plano-spherical lens is specified as a starting solution given by:

$$z_{L0} = \frac{r^2}{R_c + \sqrt{R_c^2 - r^2}} \qquad (16)$$

where $R_c$ represents the lens' radial curvature and is defined by

$$R_c = \frac{D_0(n-1)z_T}{2T_{max}} \qquad (17)$$

where $D_0$ is the lens size, $z_T$ is the target plane distance, and $T_{max}$ prescribes a maximal radial distance on our target distribution so that the initial lens choice accounts for the maximal boundary of the target pattern.

Once the parameters for the initial lens are specified, we compute the surface according to the expression for $z_L^{(i,j)}$ in an iterative fashion. Although the original paper claims that a Gauss-Seidel solver was applied to compute the resulting mesh, we could not identify a suitable simple coefficient matrix to vectorize the Monge-Ampere problem. As a result, our approach can be considered as an "element-wise" Gauss-Seidel where neighboring surface points are updated successively with nested for loops.

## 4. RESULTS

Here we display in Figures (1)-(6) result of running our implementation of the chosen algorithm to generate a single freeform lens. We first initialize our target distribution with an arbitrary image pattern (e.g. USC logo) that we convert to black and white. Then we set simulation hyperparameters for our target, lens grid, finite difference stepsize and total iterations along with physical optical parameters such as refractive index, maximal target radius, Target plane distance, and lens size.

In this example, we use a refractive index of 1.49, a unit lens of 1, target radius of 10, and target distance of 2 for the physical parameters.

For the hyperparameters we use a lens grid of size of 9801 and a target grid size of 160000 with step sizes of 0.01 to run for a total of 50 iterations. The resultant lens is shown in figures 3 and 4. We then observe the convergence of this algorithm towards the desired lens by plotting the sum of pointwise deltas between each full surface iteration against the total number of iterations as shown in Figure 7.

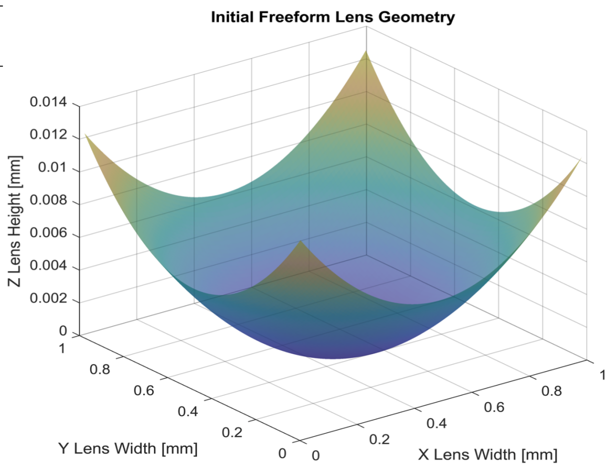

**Fig. 1.** Surface Plot of Initial Lens

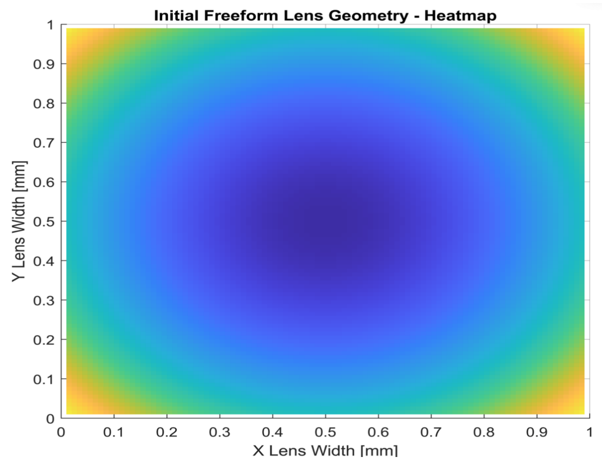

**Fig. 2.** Heat Map of Initial Lens

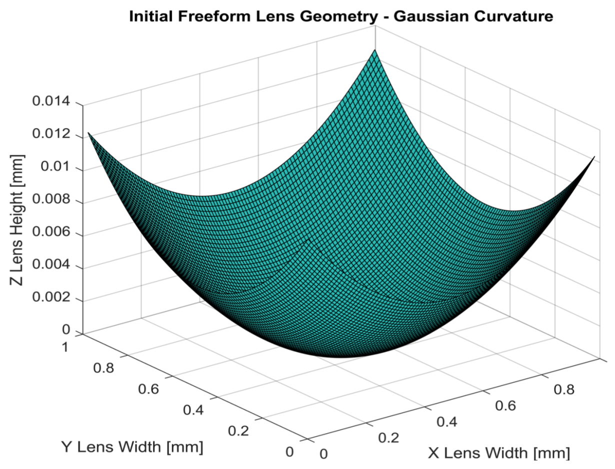

**Fig. 3.** Color Plot of Initial Lens Gaussian Curvature

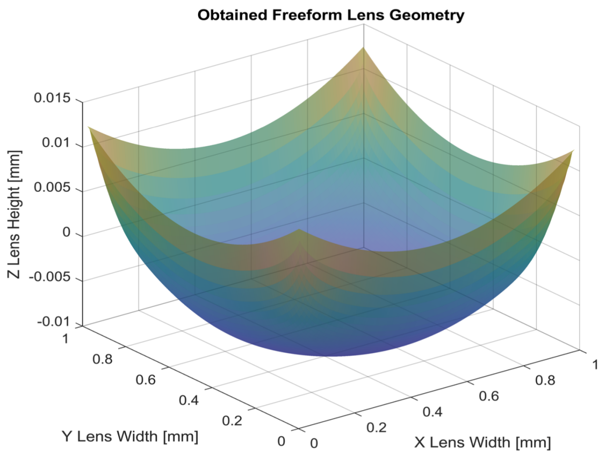

**Fig. 4.** Surface Plot of Obtained Lens

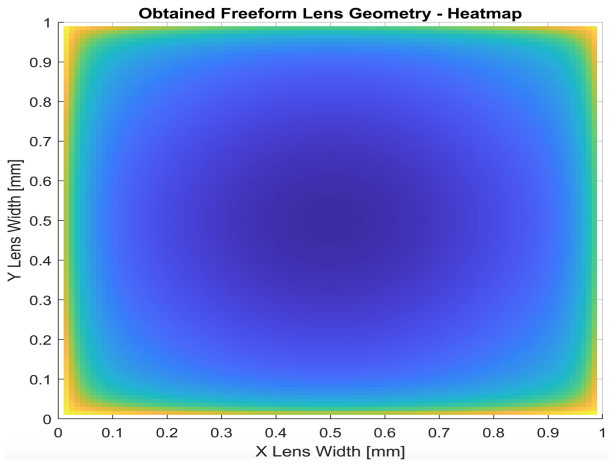

**Fig. 5.** Heat Map of the Obtained Lens

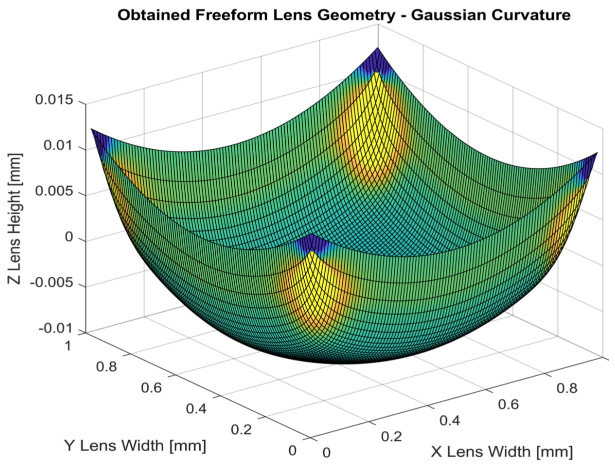

**Fig. 6.** Color Plot of Obtained Lens Gaussian Curvature

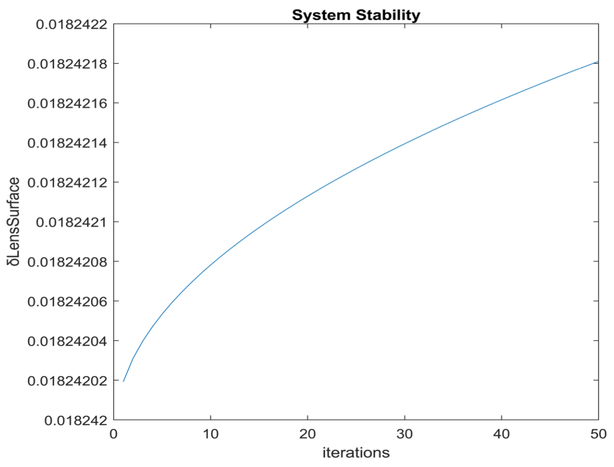

**Fig. 7.** Numerical Method Stability

## 5. DISCUSSION

The results of our project confirm the viability of this finite difference based approach as a means of generating a freeform optical surface for manufacturing. However, in developing our implementation of this algorithm, we encountered a number of limitations that prevented us from reproducing the same results as the original paper. The principal limitation was the lack of a well-defined irradiance function defined over the target domain. The lack of an explicit irradiance function meant that it was impossible to precisely replicate the illuminance of our desired target pattern since this information is central to defining the expression of our Jacobian transform between the source and target meshes as per Eq. (2) & (9). This hindrance was further confounded by the lack of a true light source as we had no actual surface power values for the light. While the authors provide an explicit mapping for the mesh distortion on the target surface elements, they do not provide any indication on the actual irradiance functions they used in their target verification experiments, nor is any usable sample data provided from which they might have approximated an irradiance function. Consequently, this left us unable to produce a freeform surface whose curvature was able to precisely reproduce a target distribution pattern owing to the fact that the Gaussian Curvature term, as in Equation (3), would have been a function of the Jacobian (which is represented in the lens case by a ratio of irradiance functions defined over the surfaces per Equations (12) & (13)).

In order to circumvent this limitation and produce a viable lens, we therefore experimented with different simplifying assumptions about the irradiance functions before eventually deciding on a Jacobian defined via a ratio of uniform unit irradiance across the source plane and the pixel-intensity value of our binarized image as a stand-in for true target irradiance (taking black positions as 0.1 and grid locations corresponding to white pixel positions as 1.1). This gave us a Jacobian that was essentially a simple linear scaling entirely defined by the pixel intensity values of the target pattern. While this is obviously an unrealistic representation of a Jacobian mapping, it was effective in allowing the algorithm to converge to a freeform lens. Although this assumption allowed us to produce the lens, it is also clearly unable to produce the high resolution illuminance pattern of our chosen target image. However, the obtained lens still adheres to the freeform paradigm in producing a solution to the prescribed curvature problem where the change in the Gaussian curvature can be clearly seen in comparing the Gaussian Curvature of the initial and obtained optical surfaces in Figures 3 and 6. Moreover, this implementation is still able to accommodate a relaxed version of target boundary conditions by selecting the maximal target radius (defined as the distance from the pattern center to its furthest edge) so that the lens solution is guaranteed to produce a ray magnification that encloses the desired pattern.

## 6. CONCLUSION

In summary, our project explores the application of differential geometry to the field of geometrical optics. In particular, we examined the way in which differential geometry offers an important theoretical framework for the design of freeform lenses used in controlled image projections through the prescribed curvature problems expressed by Monge-Ampere Boundary Value problems. Next, we reviewed the literature on commonly used approaches for designing these lens surfaces (including the equivalent Monge-Kantorovich Mass Transport Problem). Finally, we attempted a MATLAB implementation of a numerical method for the Monge-Ampere PDE as a means of computing a freeform lens surface when provided a target distribution and planar light source and subsequently presented the resultant lens.

While this project has been served to overview important applications of computational differential geometry in freeform optics, the limitations we encountered with our implementation suggest several potential directions for future work. Notably, further work is needed to examine what methods we can employ to best define or otherwise approximate realistic irradiance functions on digital images which can then be incorporated into the implementation in a manner so that our obtained lens is able to precisely produce the desired illuminance pattern of the target image. In addition to this, the relationship between the algorithms' performance and convergence as they relate to hyperparameters such as grid size, total iterations, step size, lens size, target distance, and maximal target radius also warrants additional experimentation to better understand the influence of each factor on the quality of the obtained lens. Finally, as a natural extension to further work on the irradiance function inputs, a deeper investigation of the freeform lens computation with multiple light sources will be useful for the understanding of how the computed lenses change in the presence of color wavelengths as in the case of an RGB target image.

## 7. PROJECT ACKNOWLEDGEMENTS

This project was a joint effort. The implementation and paper were worked on by both authors working in tandem and in parallel collaboration over the course of multiple zoom sessions. Lastly, we would like to thank Professor Joshi for advising on the project and for teaching this course on Computational Differential Geometry.

## 8. APPENDIX: MAIN MATLAB SCRIPT

```
clf;
clear all;

% Read in a square image, binarize, and resize
img = imread("USC_logo.jpg");
%https://www.norc.org/PublishingImages/
%Media%20Logos/USC—Logo.jpg
imshow(img);
img_bw = im2bw(img);

global target_img;
target_img = imcomplement(img_bw);

%Turn the following section on to lower
%resolution of target image
downSampleScalar = 1;
target_img = imresize(target_img, ...
     [floor(size(target_img, 1)/downSampleScalar), ...
     floor(size(target_img, 2)/downSampleScalar)]);

target_img = imcomplement(target_img);
%Turn this on if you wish to add noise to the ...
     target image:
target_img = target_img + 0.1;
target_img(target_img>1) = 1.0;

%Set parameters
%stepsize:
global h;
h = 1;

% lens surface sample dimensions:
global sampleSize;
sampleSize = 99;
global stepSizeH;
%stepSizeH = 1;
stepSizeH = double(0.01);

% targetDistance
global zT;
%zT = sampleSize*2;
zT = 2.0;
%zT = 20;

% targetSize
global Tmax;
Tmax = 10;
%Tmax = 500;

% Refractive Indicies:
refractiveIndexIn = 1.49; %PMMA
refractiveIndexOut = 1.0; %Vacuum
global n;
n = refractiveIndexIn/refractiveIndexOut;

% Stability:
total_itr = 10;
stabilityContainer = zeros(total_itr,1, 'double');

% Debug Containers:
global f1Container;
f1Container = [1];
global f2Container;
f2Container = [1];

global zLxContainer;
zLxContainer = [1];
global zLyContainer;
zLyContainer = [1];

%Initial plano spherical lens surface:
global zLens;

%lensSize:
D0 = 0.990;

%Generate table outputs for reporting:
%PhysicalParametersTable:
LensSize = D0; %sideLength
TargetSize = Tmax*2; %sideLength
TargetDistance = zT;
PHYSICAL_PARAMETERS = table(n, LensSize, ...
     TargetSize, TargetDistance)

%SampleParametersTable:
LensSampleSize=sampleSize*sampleSize;
TargetSampleSize = ...
     size(target_img,1)*size(target_img,2);
stepSize = stepSizeH;
iterations = total_itr;
SAMPLING_PARAMETERS = table(LensSampleSize, ...
     TargetSampleSize, stepSize, iterations)
```

```

%TargetImage:
figure; imshow(target_img)
axis on
title('Target Distribution')
ylabel('TargetSampleHeight')
xlabel('TargetSampleWidth')

%D0 = double(sampleSize)/100;
%D0 = double(sampleSize)/5;
[X,Y] = meshgrid(0.01: 0.01:D0, 0.01:0.01:D0);
gridDescritization = (D0*(n-1)*zT) / (1/(2*Tmax));

%lensCoordinateCenter
global lensCoordCenter;
lensCoordCenter = double(D0)/2;
circleEq = ((X-lensCoordCenter).^2)+ ...
     (Y-lensCoordCenter).^2;
%circleEq = ((X).^2)+ (Y).^2;
zLens = double(circleEq) ./ ( gridDescritization + ...
     sqrt( double((gridDescritization^2) - circleEq)));
original_zLens = zLens;
figure;
surf (X,Y,original_zLens,"EdgeColor","none");
title('Initial Freeform Lens Geometry')
zlabel('Z Lens Height [mm]')
ylabel('Y Lens Width [mm]')
xlabel('X Lens Width [mm]')
alpha(0.5);

%2D Heatmap:
figure;
surf (X,Y,original_zLens,"EdgeColor","none");
title('Initial Freeform Lens Geometry - Heatmap')
zlabel('Z Lens Height [mm]')
ylabel('Y Lens Width [mm]')
xlabel('X Lens Width [mm]')
view(2)

global i1;
global j1;

for itr = 1: total_itr
    zLensPrev = zLens;
    for i1 = 2:sampleSize-1
        for j1 = 2:sampleSize-1

            av1 = avfunc(zLens(i1+1,j1), ...
                 zLens(i1-1,j1));
            av2 = avfunc(zLens(i1,j1+1), ...
                 zLens(i1,j1-1));
            av3 = avfunc(zLens(i1+1,j1+1), ...
                 zLens(i1-1,j1-1));
            av4 = avfunc(zLens(i1-1,j1+1), ...
                 zLens(i1+1,j1-1));
            g = gfunc(i1,j1);

            zLens(i1,j1) = 0.5*(av1 + av2 - sqrt( ( ...
                 (av1-av2)^2 ) + ...
                 (((av3-av4)*0.5)^2) + ...
                 (stepSizeH^4) * (g) )  );
        end
    end
    stabilityContainer(itr) = ...
         stabilityfunc(zLensPrev, zLens);
end

% OutputLens:
[Xoutput,Youtput] = meshgrid(0.01: 0.01:D0, ...
     0.01:0.01:D0);
figure;
surf(Xoutput,Youtput,zLens,"EdgeColor","none");
title('Obtained Freeform Lens Geometry')
zlabel('Z Lens Height [mm]')
ylabel('Y Lens Width [mm]')
xlabel('X Lens Width [mm]')
alpha(0.5);
%Obtained Heatmap:
figure;
surf(Xoutput,Youtput,zLens,"EdgeColor","none");
title('Obtained Freeform Lens Geometry - Heatmap')
zlabel('Z Lens Height [mm]')
ylabel('Y Lens Width [mm]')
xlabel('X Lens Width [mm]')
view(2)

% System Stability:
% The convergence of our numerical solution over ...
     time[iterations]:

stabilityDomain = 1: total_itr;
fit = polyfit( transpose(stabilityDomain) , ...
     stabilityContainer, 1);
plot(polyval(fit,stabilityContainer));
title('System Stability')
ylabel('?LensSurface')
xlabel('iterations')

% *Generated Helper Functions:*
% surface stability vs. iteration:

function stabilityCalc = stabilityfunc(prevLens, ...
     currentLens)
global sampleSize;
stabilityCalc = double(sum( ...
     double(abs(double(double(currentLens) - ...
     double(prevLens)))), 'all' )) / ...
     double(sampleSize);
end

% Transform Equation 14 which has an unknown ...
     continous function for zL(x,y)
% into a quadratic formula for zL:

function h4g = h4gfunc(i1,j1, av1, av2, av3, av4)

global zLens;
h4g = 4*( av1-zLens(i1,j1) )*( av2-zLens(i1,j1) ) - ...
     ( ( (av3-av4)^2 )*0.25 );

end

% Average Representations of Second Partials for ...
     Hessian:
function avCalc= avfunc(input1, input2)
avCalc = double(input1 + input2)/2;
end

% Right Side of PDE with I_0/I_T = Jacobian

function gCalc = gfunc(inputI, inputJ)
%function [gCalc,f1,f2] = gfunc(inputI, inputJ)

global n;
zLx = zLxfunc(inputI, inputJ);
zLy = zLyfunc(inputI, inputJ);

K = scalarfunc(zLx, zLy);

jacob = jacobianfunc(zLx, zLy, K);
%[jacob,f1,f2] = jacobianfunc(zLx, zLy, K);

gCalc = jacob * double(sqrt(double(1 + ((zLx^2) + ...
     (zLy^2))*(1-n^2)))) / (K^2);


end

% Finite Difference Approximations for the Lens ...
     surface:
function zLx = zLxfunc(inputI, inputJ)
global stepSizeH;
global zLens;
zLx = ( 1/double(2*stepSizeH) ) * ( ...
     zLens(inputI+1,inputJ) - ...
     zLens(inputI-1,inputJ) );
end

function zLy = zLyfunc(inputI, inputJ)
global stepSizeH;
global zLens;
zLy = ( 1/double(2*stepSizeH) ) * ( ...
     zLens(inputI,inputJ+1) - ...
     zLens(inputI,inputJ-1) );
end

% Scalar Function for Refraction:
function K = scalarfunc(zLx, zLy)

global n;
```

```
K = double( n − sqrt( double(1+ ( (zLx^2) + (zLy^2) ...
      )*(1−n^2) ) ) ) / double( (zLx^2) + (zLy^2) + ...
      1 );

end

function jacob = jacobianfunc(zLx, zLy, K)
%function [jacob,f1,f2] = jacobianfunc(zLx, zLy, K)
global zT;
global target_img;
global lensCoordCenter;
global f1Container;
global f2Container;
global zLxContainer;
global zLyContainer;

global i1;
global j1;

i1_int = int16(i1);
j1_int = int16(j1);
f1 =  ceil( real (zT * double((zLx) * ...
      K)/double(1−K))) +i1_int+1;
f2 =  ceil( real (zT * double((zLy) * ...
      K)/double(1−K))) +j1_int+1;

zLxContainer = [zLxContainer; zLx];
zLyContainer = [zLyContainer; zLy];

f1Container = [f1Container; f1];
f2Container = [f2Container; f2];

%Simplifying Assumption used because of lack of ...
      actual Irradiance Function
jacob = double(1) / double(target_img(f1, f2));

end
```

|

## REFERENCES

1. W. E. B. A. C. P. G. D. S. A. W. W. Born, M., *Principles of Optics: Electromagnetic Theory of Propagation, Interference and Diffraction of Light.* (Cambridge University Press, 1999).
2. M. Bass, C. DeCusatis, J. Enoch, V. Lakshminarayanan, G. Li, C. Macdonald, V. Mahajan, and E. Van Stryland, *Handbook of Optics, Third Edition Volume I: Geometrical and Physical Optics, Polarized Light, Components and Instruments* (McGraw-Hill Inc., USA, 2009), 3rd ed.
3. W. G. . W. Y. Rubinstein, J., "Ray mappings and the weighted least action principle," J. Math. Ind. .
4. R. Wu, L. Xu, P. Liu, Y. Zhang, Z. Zheng, H. Li, and X. Liu, "Freeform illumination design: a nonlinear boundary problem for the elliptic monge&#x2013;amp&#xe9;re equation," Opt. Lett. **38**, 229–231 (2013).
5. K. Brix, Y. Hafizogullari, and A. Platen, "Designing illumination lenses and mirrors by the numerical solution of monge–ampère equations," J. Opt. Soc. Am. A **32**, 2227 (2015).
6. B. D. Froese and A. M. Oberman, "Convergent finite difference solvers for viscosity solutions of the elliptic monge-ampère equation in dimensions two and higher," (2010).
7. P. Benitez, R. M. Arroyo, and J. C. Minano, "Design in 3D geometry with the simultaneous multiple surface design method of nonimaging optics," **3781**, 12 – 21 (1999).
8. J. Wojtanowski, "Optimal mass transportation problem and freeform optics design – the identity of optimization scheme and the numerical solution method," Opt. Appl. **48**, 399–412 (2018).
9. L. L. Doskolovich, D. A. Bykov, A. A. Mingazov, and E. A. Bezus, "Optimal mass transportation and linear assignment problems in the design of freeform refractive optical elements generating far-field irradiance distributions," Opt. Express **27**, 13083–13097 (2019).
10. L. L. Doskolovich, A. A. Mingazov, D. A. Bykov, E. S. Andreev, and E. A. Bezus, "Variational approach to calculation of light field eikonal function for illuminating a prescribed region," Opt. Express **25**, 26378–26392 (2017).
11. D. A. Bykov, L. L. Doskolovich, A. A. Mingazov, E. A. Bezus, and N. L. Kazanskiy, "Linear assignment problem in the design of freeform refractive optical elements generating prescribed irradiance distributions," Opt. Express **26**, 27812–27825 (2018).
12. R. Wu, Z. Yaqin, M. Sulman, Z. Zheng, P. Benitez, and J. Miñano, "Initial design with l2 monge-kantorovich theory for the monge–ampère equation method in freeform surface illumination design," Opt. Express **22** (2014).
13. C. Bösel and H. Gross, "Single freeform surface design for prescribed input wavefront and target irradiance," J. Opt. Soc. Am. A **34**, 1490–1499 (2017).
14. J. Wojtanowski, "Efficient numerical method of freeform lens design for arbitrary irradiance shaping," J. Mod. Opt. **65**, 1019–1032 (2018).
15. L. Doskolovich, D. Bykov, A. Mingazov, and E. Bezus, "Optimal mass transportation and linear assignment problems in the design of freeform refractive optical elements generating far-field irradiance distributions," Opt. Express **27**, 13083 (2019).
16. C. Bösel, N. G. Worku, and H. Gross, "Ray-mapping approach in double freeform surface design for collimated beam shaping beyond the paraxial approximation," Appl. Opt. **56**, 3679 (2017).
17. C. R. Prins, J. H. M. Ten Thije Boonkkamp, J. van Roosmalen, W. L. Jzerman, and T. W. Tukker, "A monge–ampère-solver for free-form reflector design," SIAM J. on Sci. Comput. **36**, B640–B660 (2014).
18. J. Sasián, D. Reshidko, and C.-L. Li, "Aspheric/freeform optical surface description for controlling illumination from point-like light sources," Opt. Eng. **55**, 1 – 5 (2016).
19. D. Bianchi, M. Sánchez del Río, and C. Ferrero, "Ray tracing of optical systems using nurbs surfaces," Phys. Scripta **82**, 015403 (2010).
20. A. Karakhanyan and X.-J. Wang, "On the reflector shape design," J. Differ. Geom. **84**, 561–610 (2010).