COMPUTATIONAL METHODS IN FREEFORM OPTICAL TECHNOLOGIES

KEVIN CHEW FIGUEROA & ZENAS HUANG

EE 575 PROJECT - COMPUTATIONAL DIFFERENTIAL GEOMETRY

OUTLINE

I. INTRODUCTION & DESCRIPTION

II. THEORETICAL SETUP & FRAMEWORK

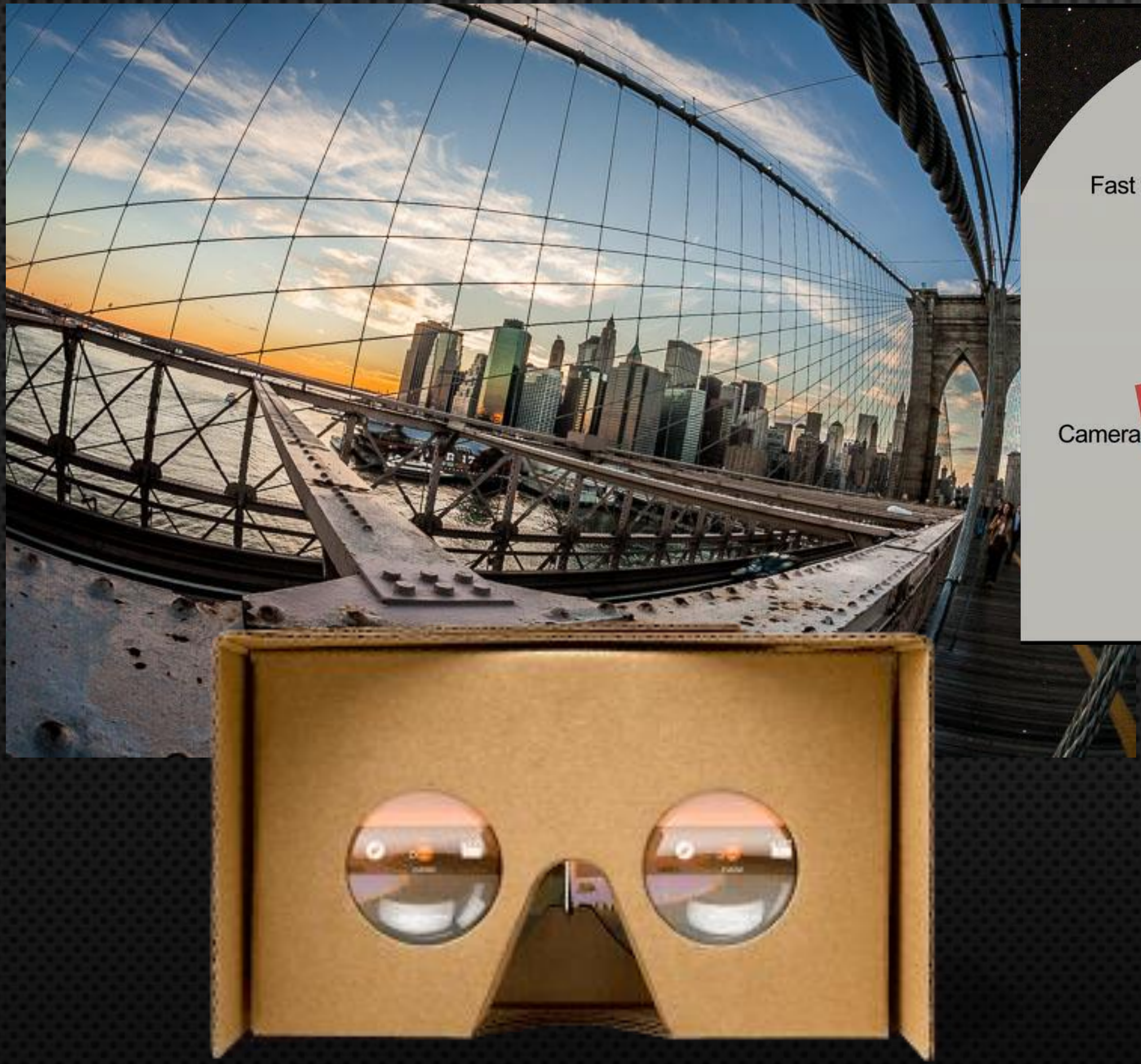
III. IMPLEMENTATION

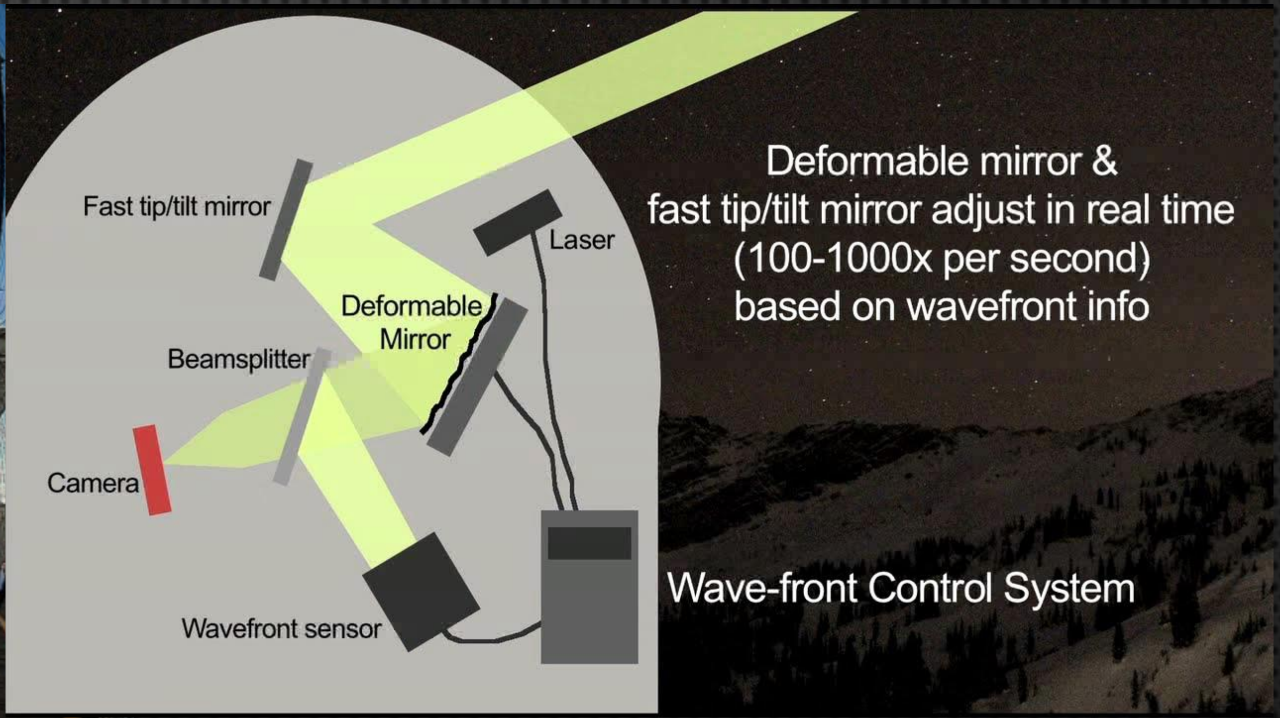
IV. CONCLUSIONS & FUTURE DIRECTIONS

Introduction

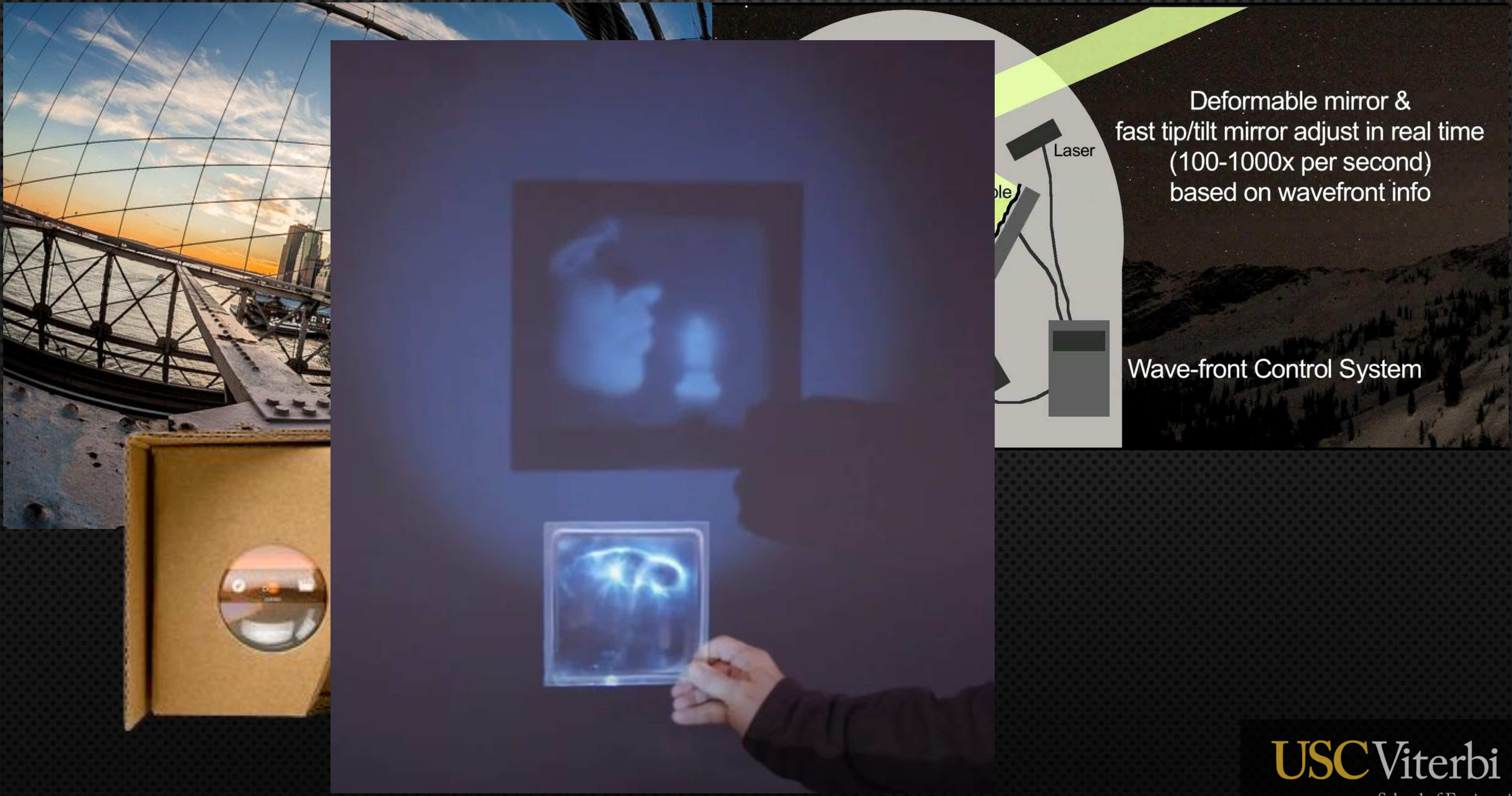
- Optical Applications
  - Corrective Lenses
  - Image Formation
  - AR/VR Systems
  - Telescopes
  - Precision Cameras
  - LED Display

USC Viterbi
School of Engineering





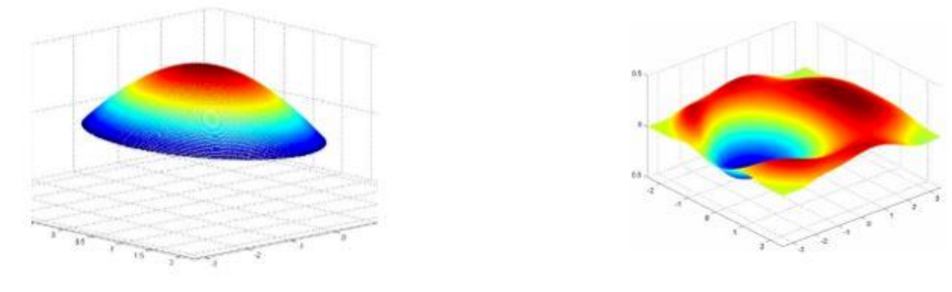
USC Viterbi
School of Engineering



USC Viterbi
School of Engineering

Introduction

- Freeform Lenses
  - In optics, freeform lenses are optical surfaces that are designed without rigid radial dimensions such as translational or rotational symmetry about axes normal to any mean plane.



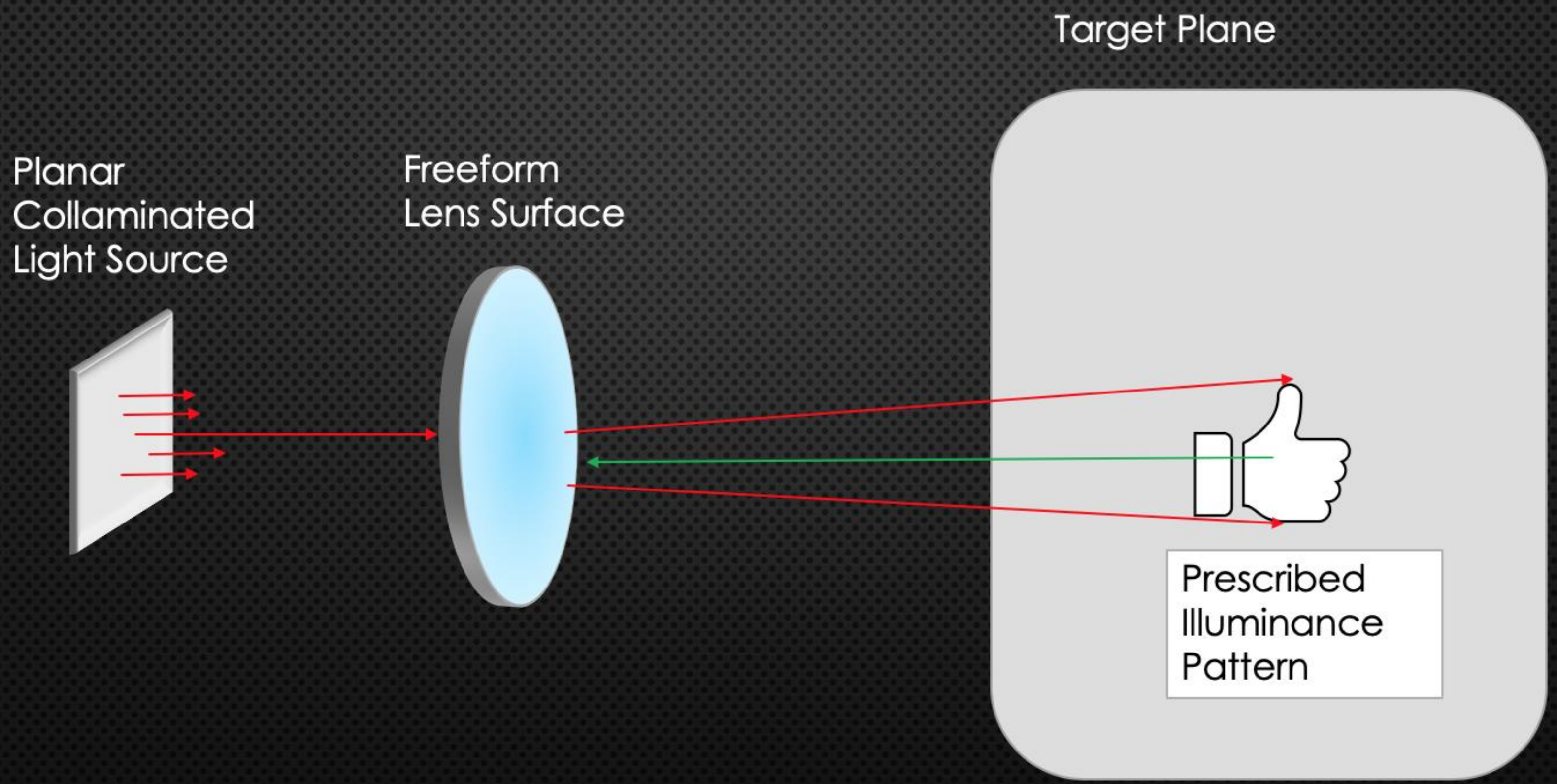
Traditional Rotationally Symmetrical Lens

Not a free-form lens

Non-rotationally Symmetrical Lens

Free-form Lens

GENERAL SETUP



GIVEN AN ILLUMINATION SOURCE AND AN IRRADIANCE DISTRIBUTION ON A PLANE. HOW CAN WE COMPUTE THE SURFACE OF THE LENS NEEDED TO GENERATE THE TARGET DISTRIBUTION?

FRAMEWORK

EIKONAL EQUATION:

$$\nabla^2 E = c^2(x, y, z)$$

SOLUTIONS TO THE 3D WAVE EQUATION THAT REPRESENT LIGHT WAVEFRONT AT DIFFERENT TIMES.

INTEGRATING THE EIKONAL AT DIFFERENT SPATIAL POINTS YIELDS AN OPTICAL RAY PATH LENGTH

$$\int_A^B \nabla^2 E \; de \approx \sum_{i \in [A,B]} n_i d_i$$

FIND A REFRACTIVE OPTICAL SURFACE WHICH GIVES US RAY PATHS THAT SATISFY THE CONDITIONS FROM THE SOURCE (A) AND TARGET ENDPOINT LOCATIONS (B).

USC Viterbi School of Engineering

FRAMEWORK

THE COORDINATES OF A LIGHT RAY ON THE TARGET PLANE ARE RELATED TO THE SOURCE POINTS (X,Y) BY A VECTOR VALUED FUNCTION Z(X, Y) SO THAT:

$$x_t = z_1(x, y) \; ; y_t = \; z_2(x, y)$$

SNELL'S LAW AND THE JACOBIAN DEFINES A CONFORMAL MAPPING FROM THE FREEFORM LENS SURFACE TO THE TARGET PLANE:

$$dx_t \, dy_t = |\, J(\mathrm{Z})|\ \mathrm{DXDY}$$

WHERE:

$$J(z) = \begin{bmatrix} \frac{\partial z_1}{\partial x} & \frac{\partial z_1}{\partial y} \\ \frac{\partial z_2}{\partial x} & \frac{\partial z_2}{\partial x} \end{bmatrix}$$

MORE PRECISELY, THE PHYSICS INTERPRETATION DESCRIBES THE TRANSFORM BETWEEN IRRADIANCE DISTRIBUTIONS OF THE TARGET AND FREEFORM SURFACES:

$$dI_t(x_t, y_t) = |\, J(\mathrm{Z})| dI(\mathrm{X, Y})$$

USC Viterbi
School of Engineering

FRAMEWORK

GIVEN:

$$dI_t(x_t, y_t) = |J(\mathrm{Z})|\, \mathrm{DI}(\mathrm{X}, \mathrm{Y}) \qquad \text{(Conformal Map of Irradiance Distributions)}$$

$$\iint I_t(x_t, y_t)\ dx_t dy_t = \iint I(x, y)\, dxdy \qquad \text{(Conservation of Energy)}$$

THESE CONDITIONS IMPLY A NONLINEAR MONGE-AMPERE ELLIPTICAL PDE :

$$\det D^2 z = K(\bar{x})(1 + |Dz|^2)^{\frac{n+2}{2}} \qquad \text{(Prescribed Curvature Problem)}$$

ALSO EXPRESSED IN A LOCALLY LINEARIZED FORM AS:

$$A\left(z_{xx}z_{yy} - {z_{xy}}^2\right) + Bz_{xx} + Cz_{yy} + Dz_{xy} + E = 0$$

$$BC: \begin{cases} x_t = f_1(x, y, z, z_x, z_y) \\ y_t = f_2(x, y, z, z_x, z_y) \end{cases} : \partial S \rightarrow \partial T$$

WHERE THE TARGET COORDINATES ARE RE-EXPRESSED AS FUNCTIONS OF THE RAY SOURCE AND THEIR INCIDENCE AT THE OPTICAL SURFACE.

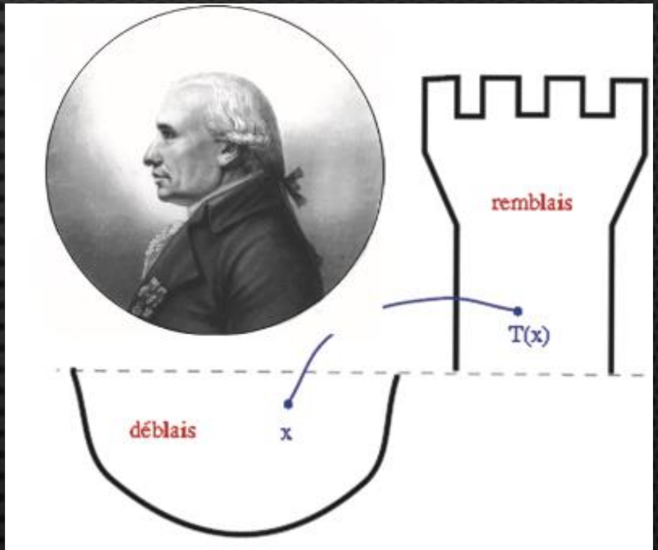
FRAMEWORK

Taking the view of light as a photon mass, Energy conservation also implies an Optimal Mass Transport problem (Monge-Kantorovich Formulation):

$$\int c(\bar{x}, T(\bar{x}))I(\bar{x})\,d\bar{x}$$

S/T:

$$T: I(\bar{x}) \rightarrow I_t(\bar{x}_t)$$



Where the mass-preserving transform $T(\bar{x})$ expresses a set of constraints for the Irradiance correspondence between each source to target spatial pairing.

$T(\bar{x})$ has also been shown to be the gradient of the MA PDE solutions.

Existing Methods in Literature

- Simultaneous Multiple Surface Method
- MA PDE and Newton Methods
- MK Transport as Linear Program
- NURBS Approaches

USC Viterbi School of Engineering

FRAMEWORK

THE GOAL IS TO COMPUTE A CHART CONSISTING OF LOCAL MONGE PATCHES DEFINED AS

$m: U \to \mathbb{R}^3, \mathrm{U} \subseteq \mathbb{R}^2$ WHERE $m(x,y) = \big(x, y, z(x,y)\big)$ UNIQUELY SATISFIES THE BOUNDARY CONDITIONS.

FROM THE ELLIPTICAL MA PDE AND THE FUNDAMENTAL FORMS, THE MONGE PATCHES SHOULD HAVE:

$$\det D^2 z = K(\bar{x})(1 + |Dz|^2)^{\frac{n+2}{2}} \Leftrightarrow K_{Gaussian} = \frac{z_{xx}z_{yy} - {z_{xy}}^2}{(1 + {z_x}^2 + {z_y}^2)^2}$$

IN OUR CASE, WE FOUND AN EXPLICIT EXPRESSION OF MONGE-AMPERE USED IN LENS DESIGN:

$$\left|z_{Lxx}z_{Lyy} - {z_{Lxy}}^2\right| = |J(z)| \frac{\sqrt{1 + ({z_{Lx}}^2 + {z_{Ly}}^2)(1 - n^2)}}{K(z_{Lx}, z_{Ly})^2}$$

FRAMEWORK

Using the Lens Expression of Monge-Ampere we compute the surface as a function of Finite Difference Approximations:

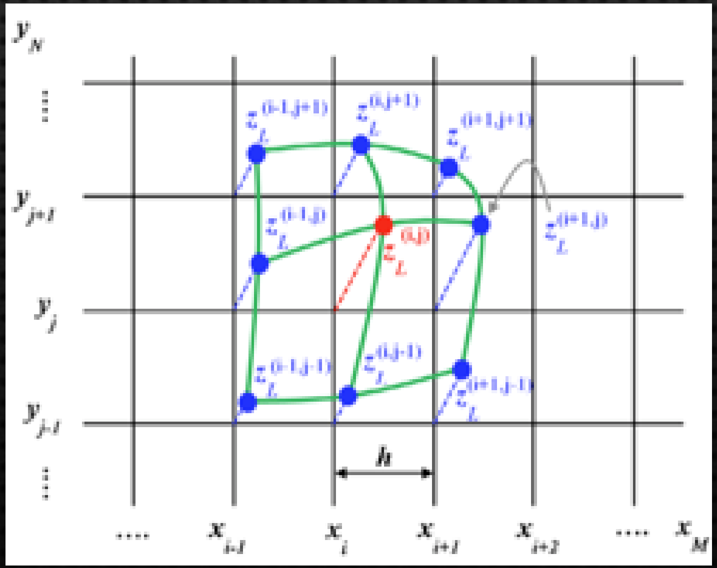
$$z_L^{(i,j)} = \frac{av_1^{(i,j)} + av_2^{(i,j)} - \sqrt{(av_1^{(i,j)} - av_2^{(i,j)})^2 + \left(\frac{av_3^{(i,j)} - av_4^{(i,j)}}{2}\right)^2 + h^4 g^{(i,j)}}}{2}$$

Solving each point using An iterative Method Based on Gauss-Seidel.

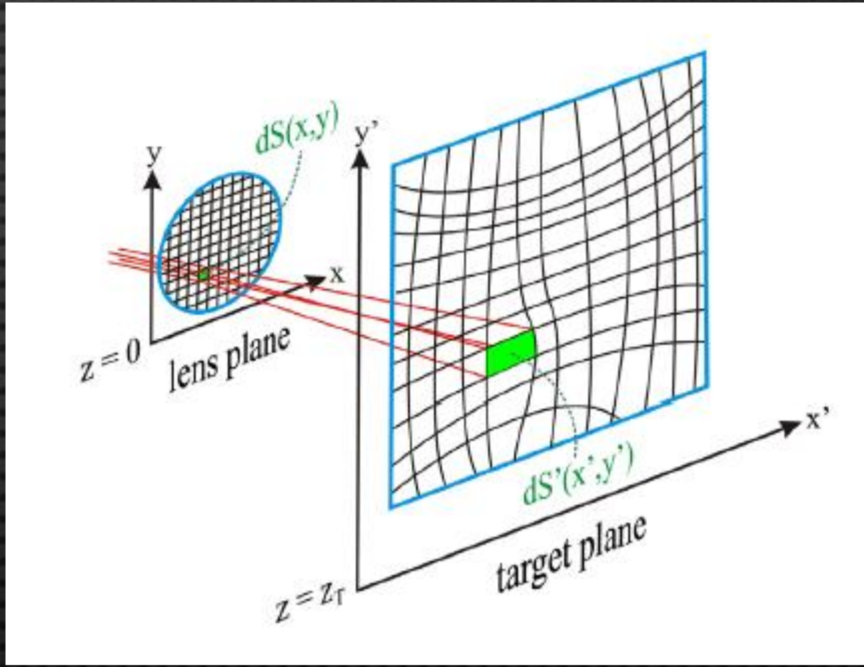
For t = (1:Iters)

$$z_L^{t+1} = f(z_L^t, z_L^{t-1});$$

End

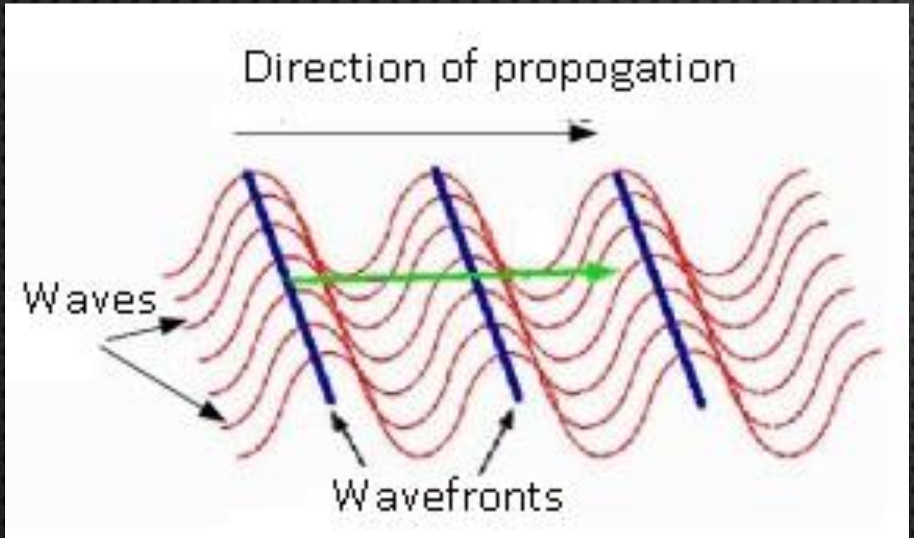


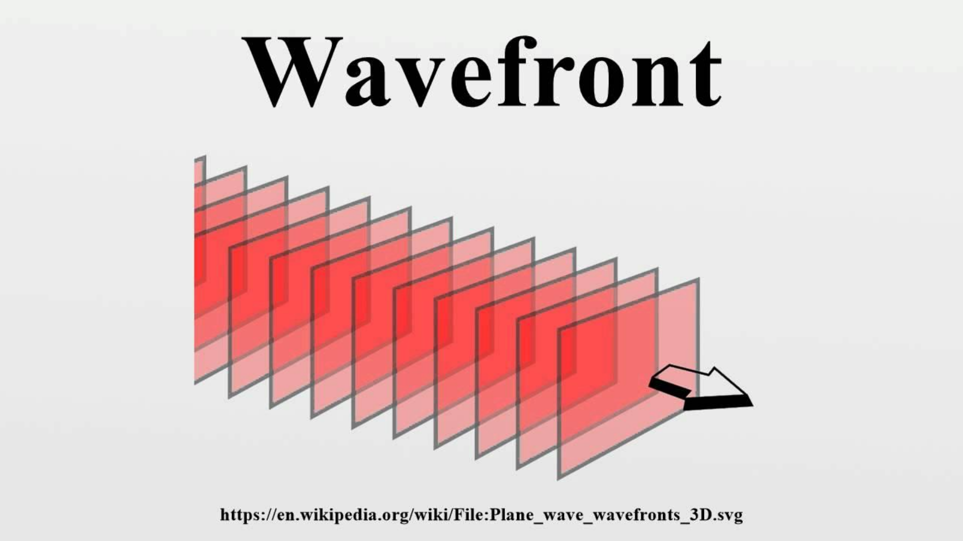
IMPLEMENTATION

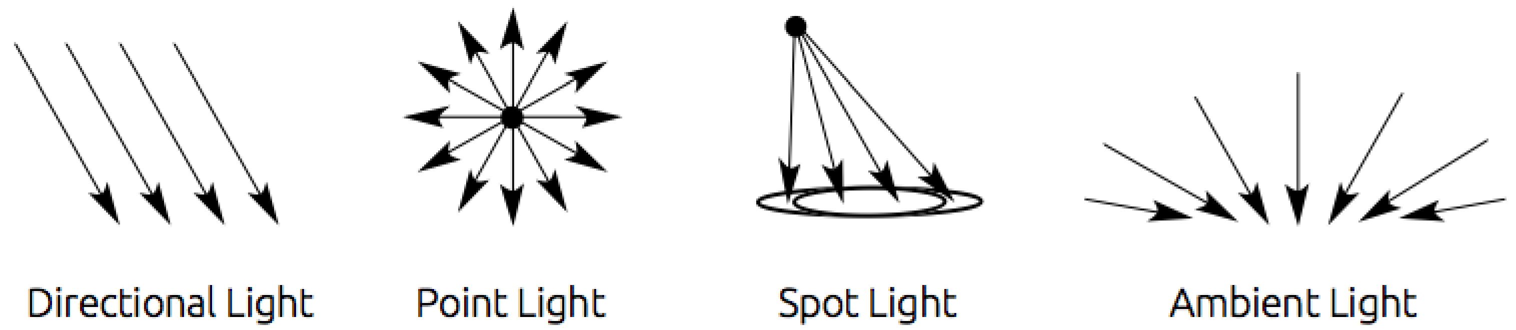


USC Viterbi
School of Engineering

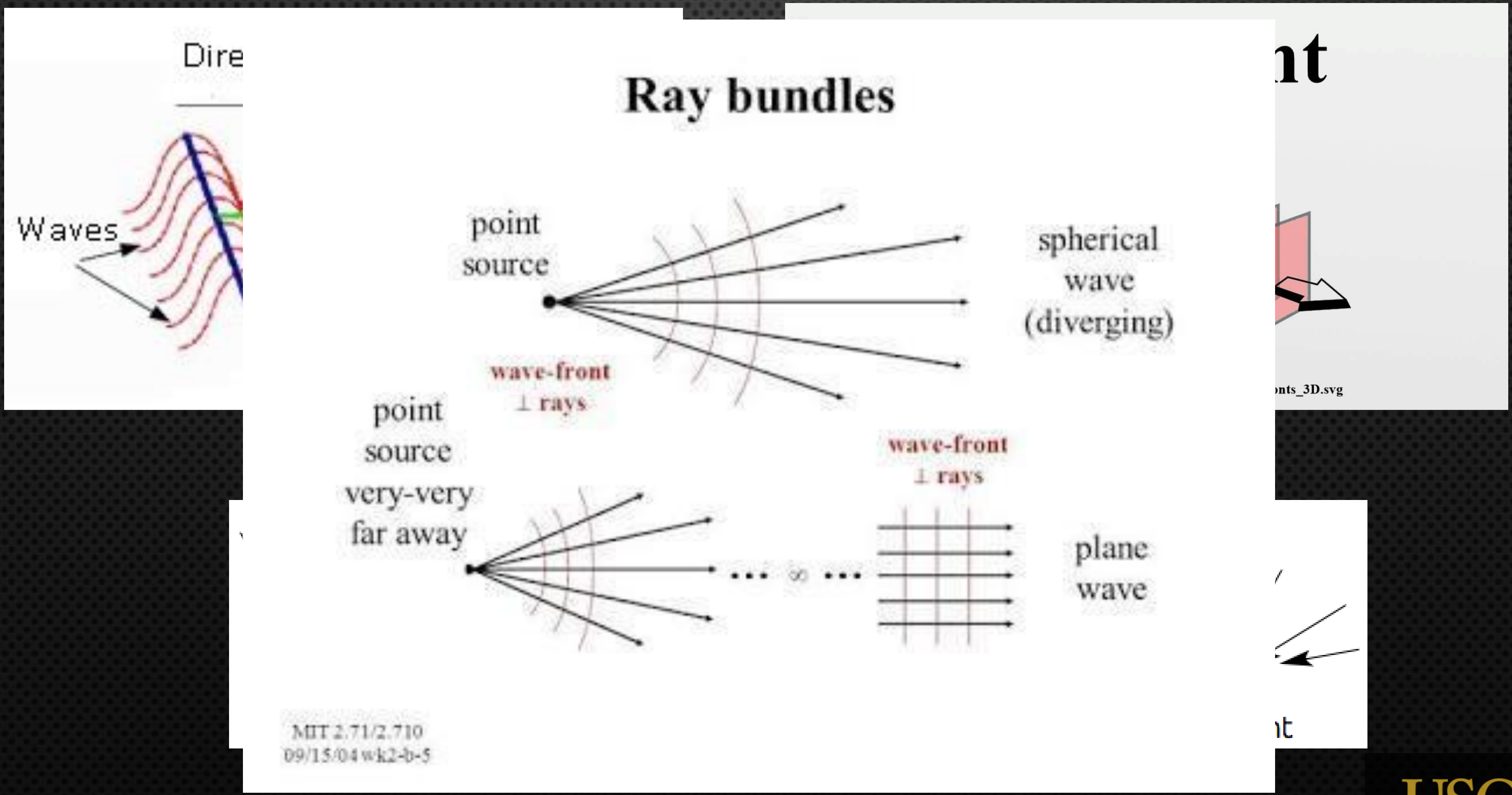
IMPLEMENTATION





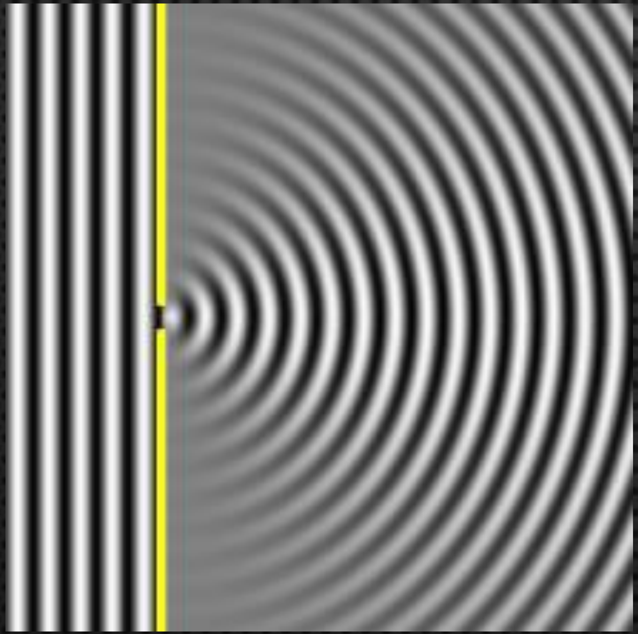


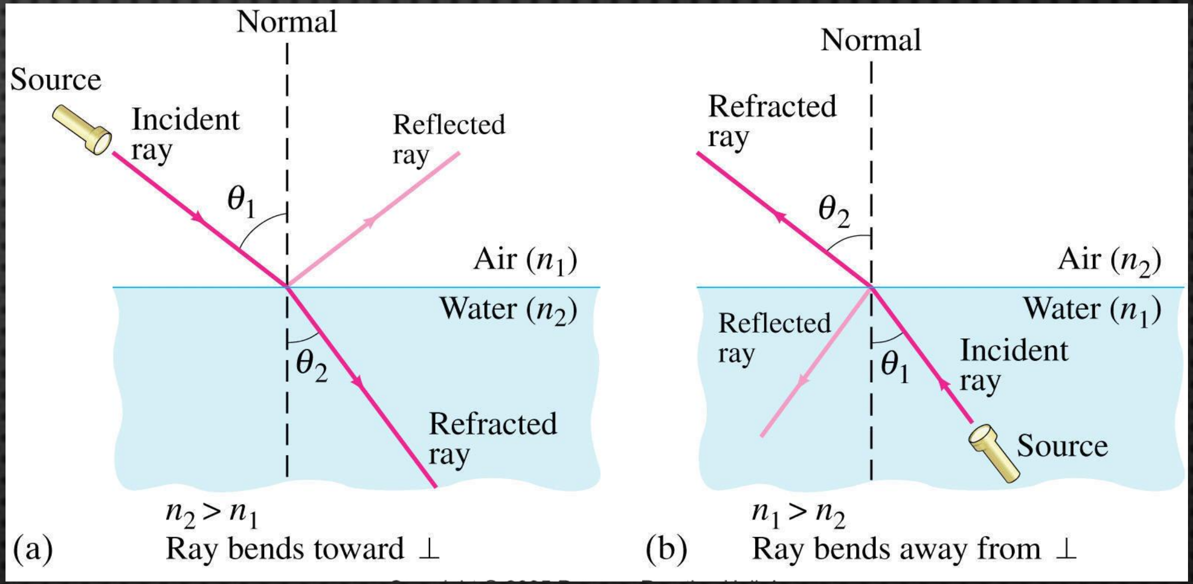
IMPLEMENTATION



USC Viterbi
School of Engineering

IMPLEMENTATION



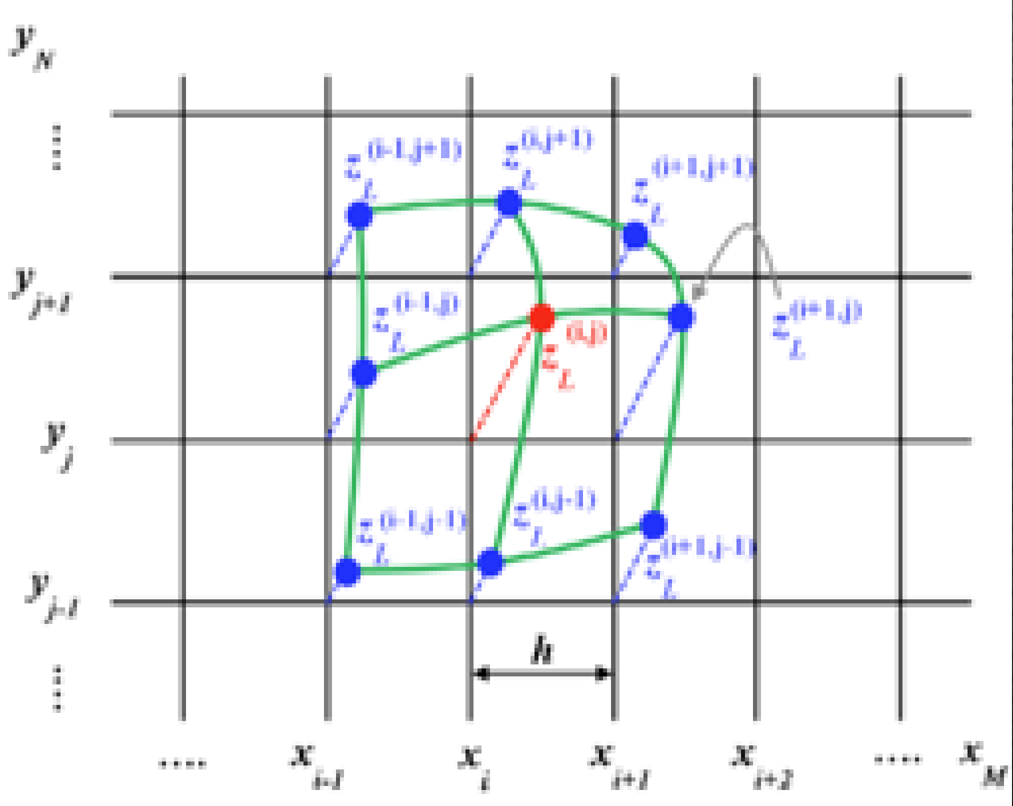


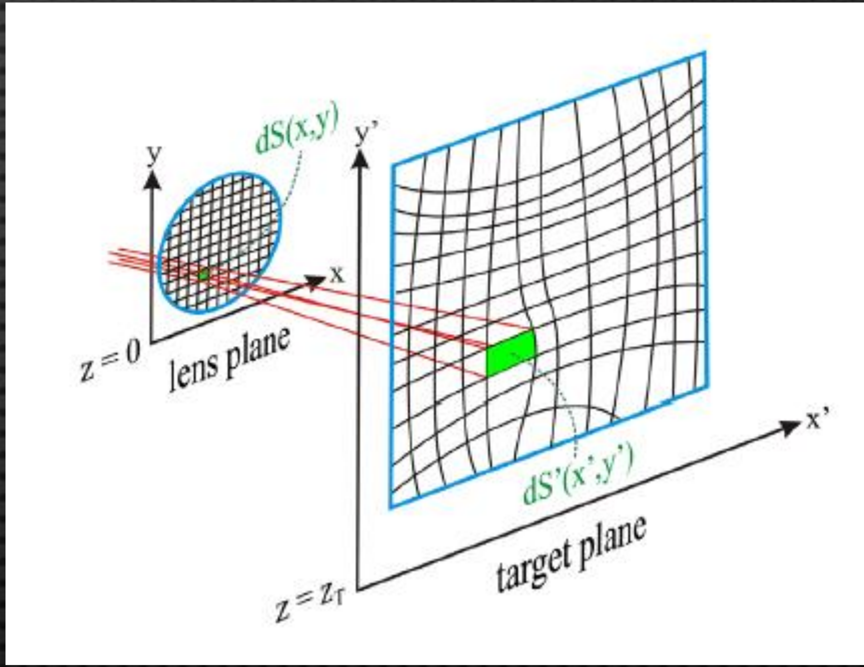
Snell's Law:

$$n_1 \sin\theta_1 = n_2 \sin\theta_2$$

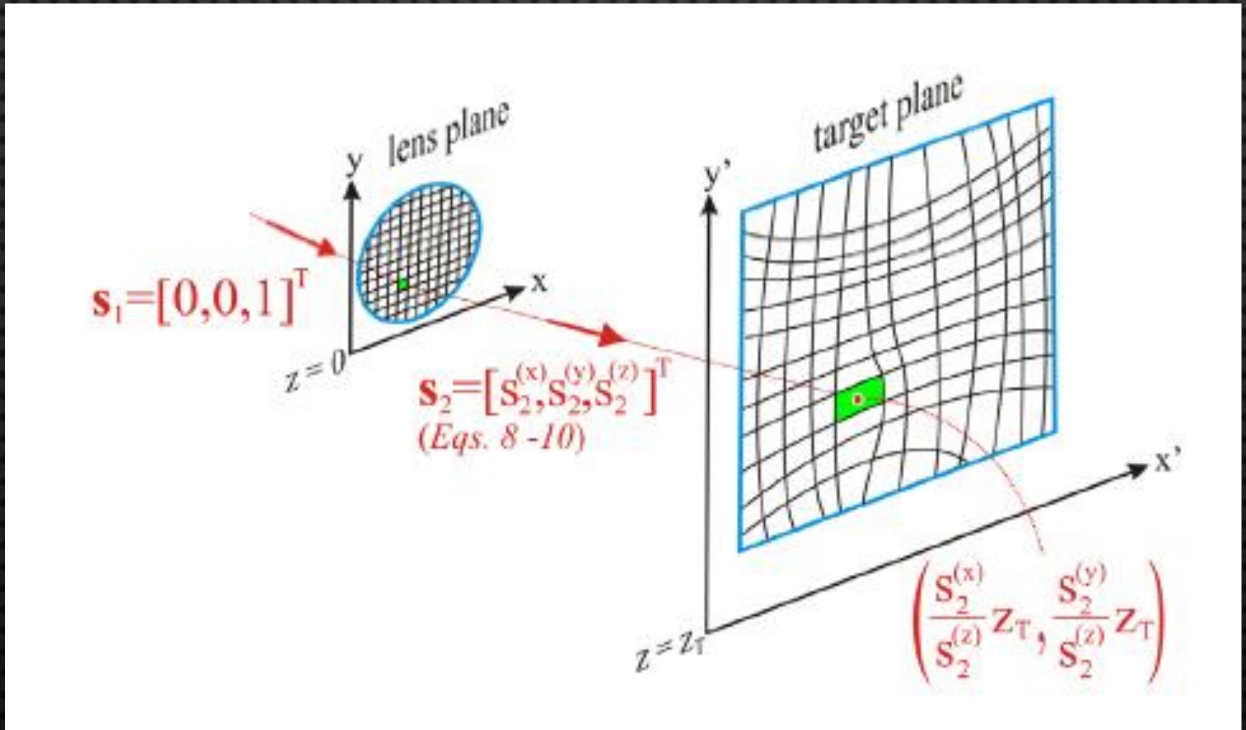
USC Viterbi School of Engineering

IMPLEMENTATION





IMPLEMENTATION



USC Viterbi School of Engineering

IMPLEMENTATION


```
clf;
clear all;

img = imread("USC_logo2.jpg");
imshow(img);
```

IMPLEMENTATION

NxN x 3

IMPLEMENTATION

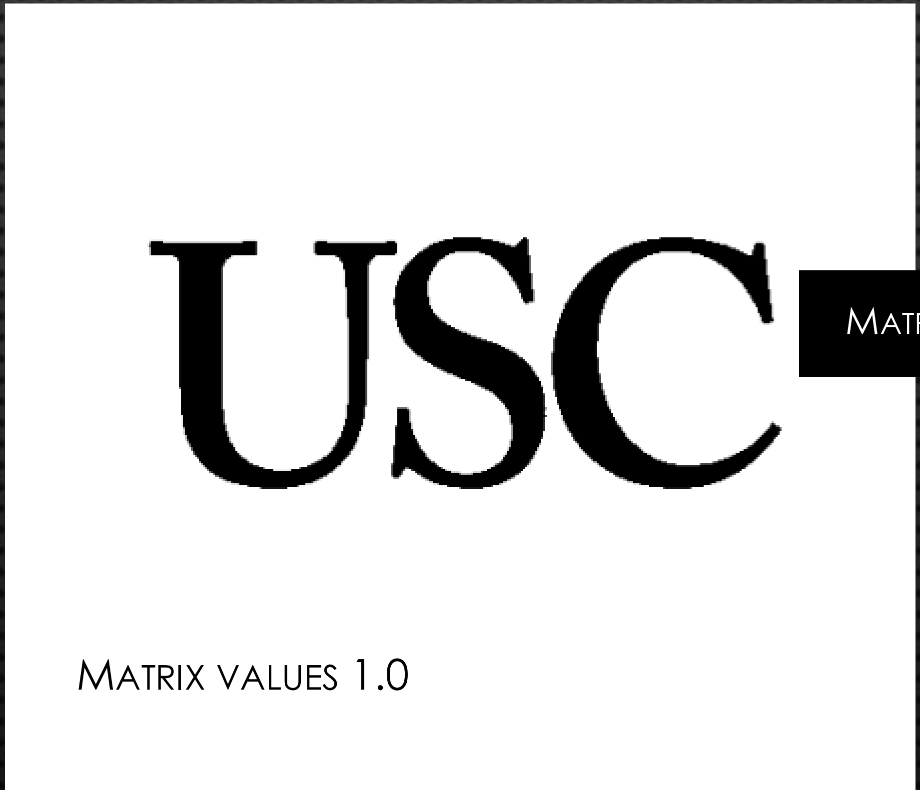
IMPLEMENTATION


```
img_bw = im2bw(img);

global target_img;
target_img = imcomplement(img_bw);


%Turn the following section on to lower
%resolution of target image
downSampleScalar = 100;
target_img = imresize(target_img, [floor(size(target_img, 1)/downSampleScalar), floor(size(target_img, 2)/downSampleScalar)]);

target_img = imcomplement(target_img);
imshow(target_img);
```

USC Viterbi
School of Engineering

IMPLEMENTATION

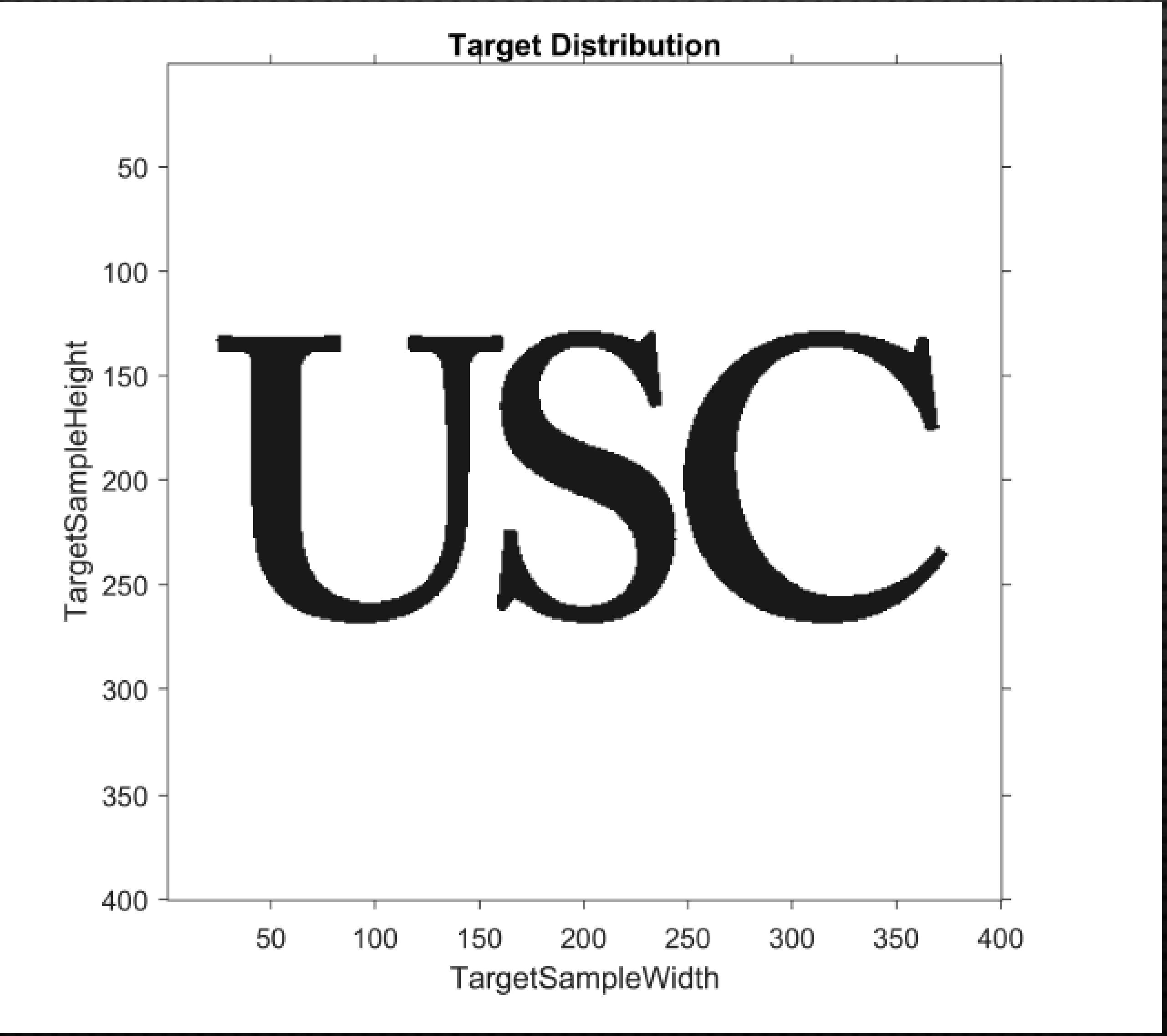
IMPLEMENTATION



MATRIX VALUES 0.10

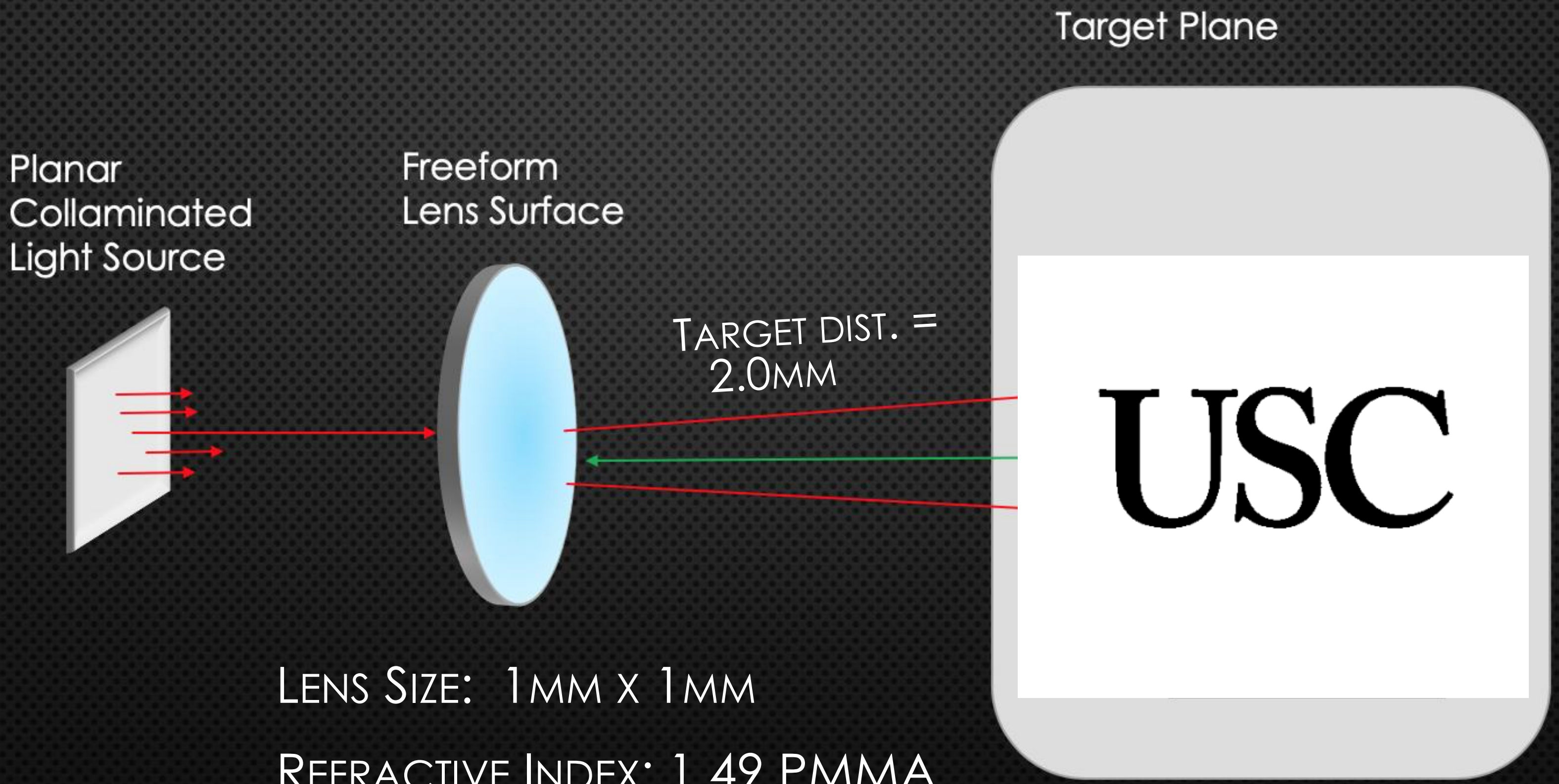
```
%Turn this on if you wish to add noise to the target image:
target_img = target_img + 0.1;
target_img(target_img>1) = 1.0;
imshow(target_img)
```

IMPLEMENTATION



IMPLEMENTATION

RESULTS:



LENS SIZE: 1MM X 1MM

REFRACTIVE INDEX: 1.49 PMMA

LENS SIZE: 20MM X 20MM

Next we define our hyperparameters:

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;
```

IMPLEMENTATION

```
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
```

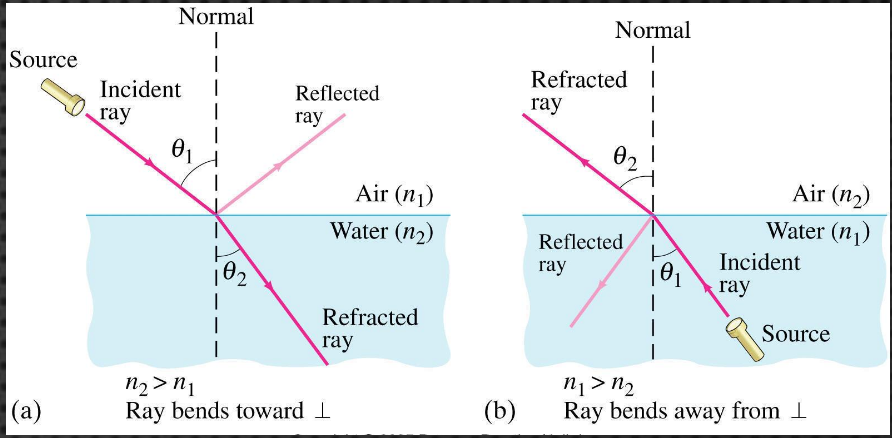
USC Viterbi School of Engineering

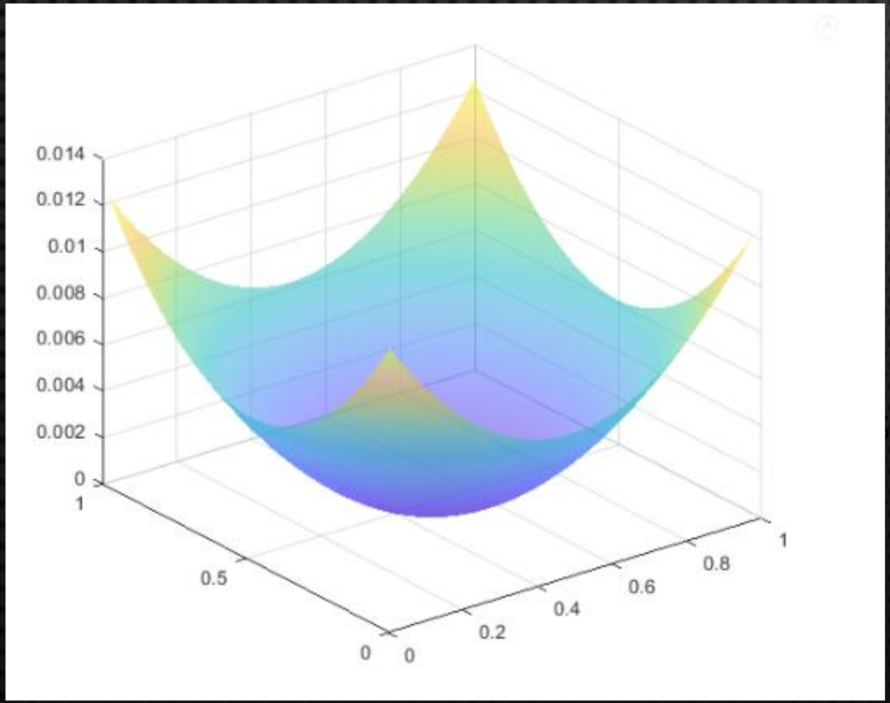
IMPLEMENTATION

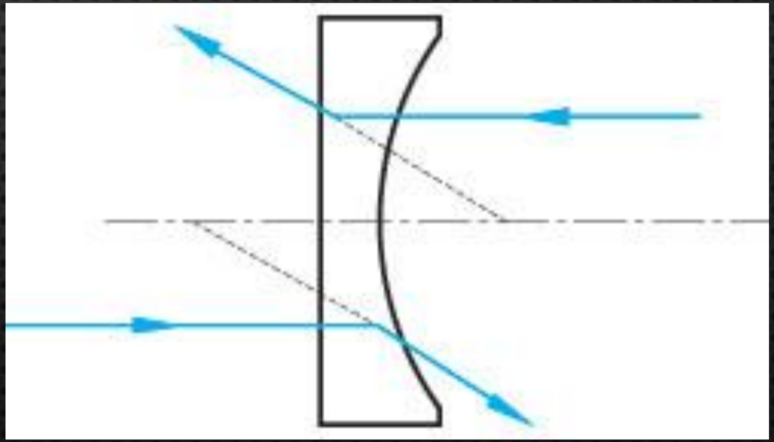
Lens Size = 1mm x 1mm | | | D0 1mm

Target distance Zt= 2.0mm

Target max dimensions = 20mm x 20mm | Tmax = 10mm







Snell's Law:

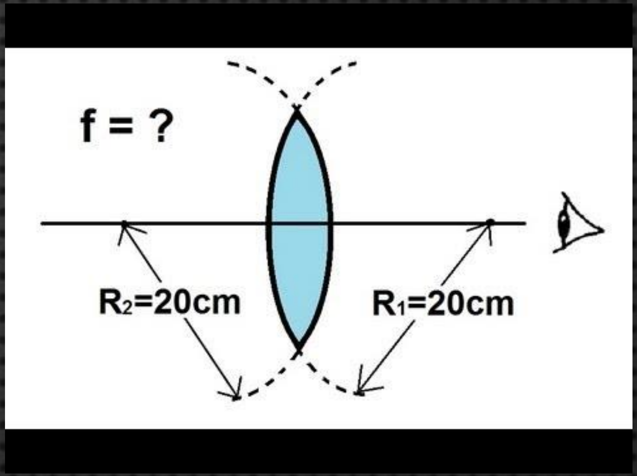
$$n_1 \sin\theta_1 = n_2 \sin\theta_2$$

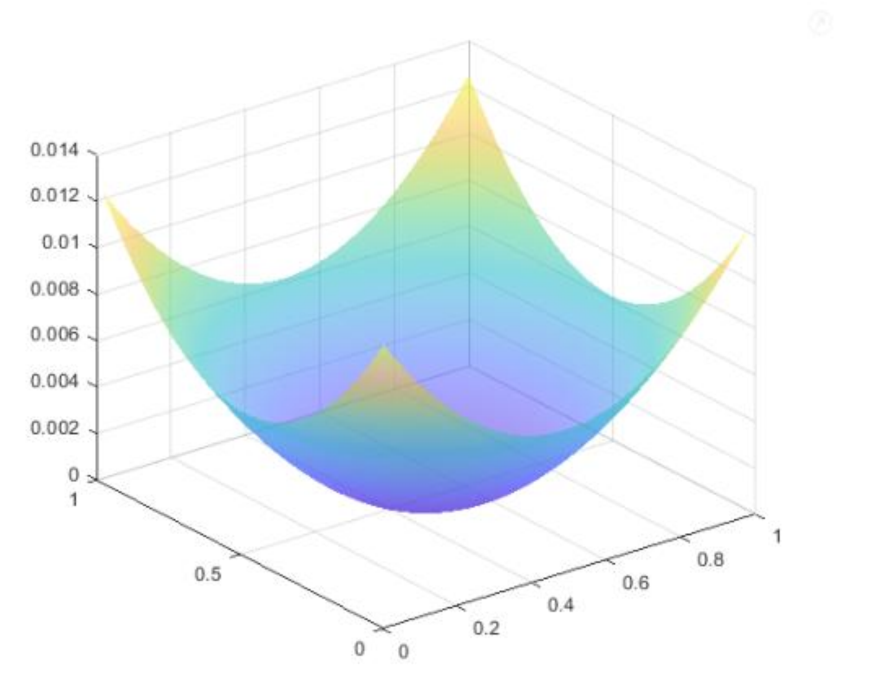
IMPLEMENTATION

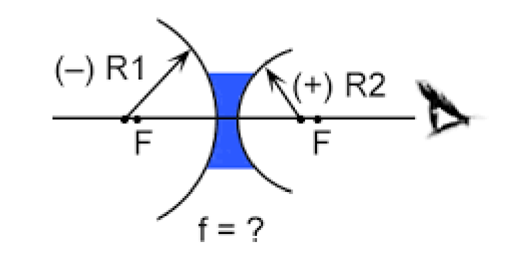
LENS SIZE = 1MM X 1MM | | | D0 1MM

TARGET DISTANCE ZT= 2.0MM

TARGET MAX DIMENSIONS = 20MM X 20MM | TMAX = 10MM







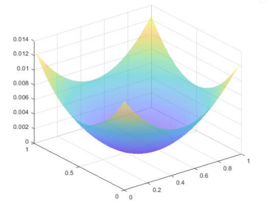
$$\frac{1}{f} \approx (n-1)\left[\frac{1}{R_1} - \frac{1}{R_2}\right]$$

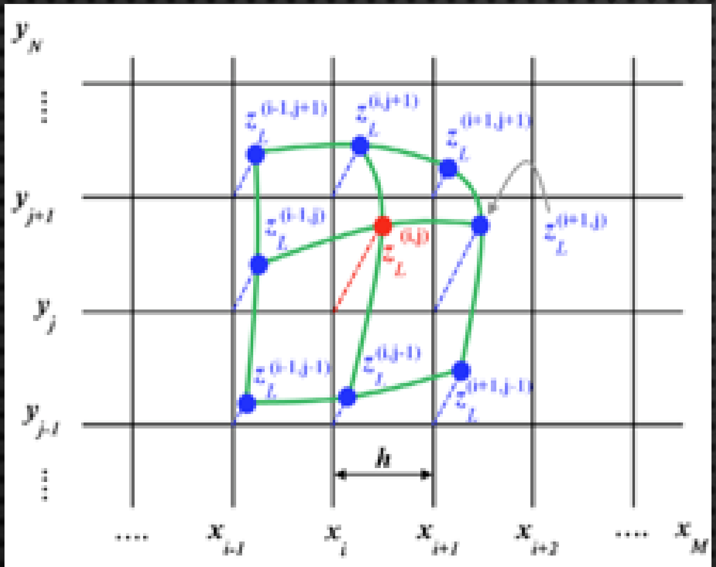
$$R_c = \frac{D_0(n-1)z_T}{2T_{\max}}$$

$$z_{L0}^{(i,j)} = \frac{x_{i,j}^2 + y_{i,j}^2}{\frac{D_0(n-1)z_T}{2T_{\max}} + \sqrt{\left(\frac{D_0(n-1)z_T}{2T_{\max}}\right)^2 - (x_{i,j}^2 + y_{i,j}^2)}}$$

IMPLEMENTATION

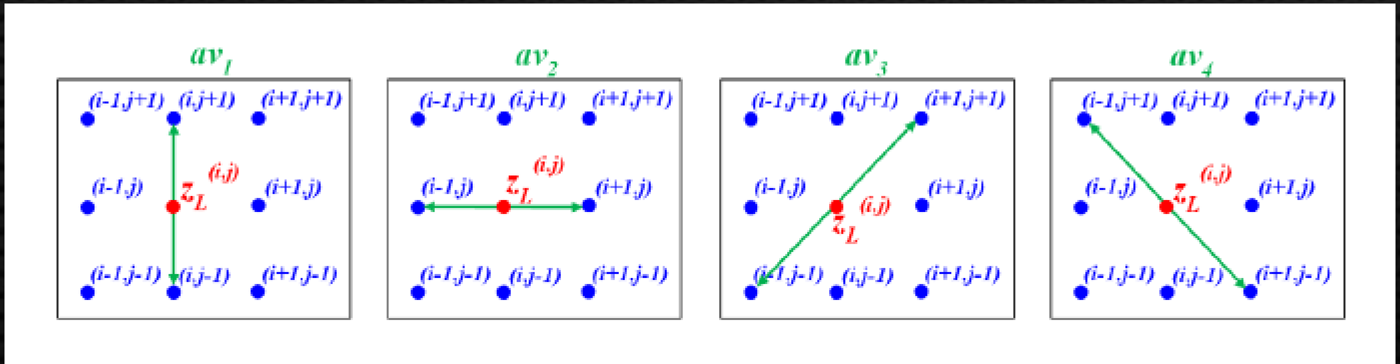
$$z_L^{(i,j)} = \frac{av_1^{(i,j)} + av_2^{(i,j)} - \sqrt{\left(av_1^{(i,j)} - av_2^{(i,j)}\right)^2 + \left(\frac{av_3^{(i,j)} - av_4^{(i,j)}}{2}\right)^2 + h^4 g^{(i,j)}}}{2}$$





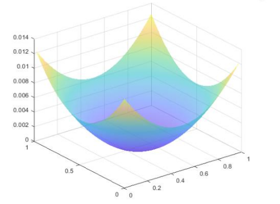
Average:

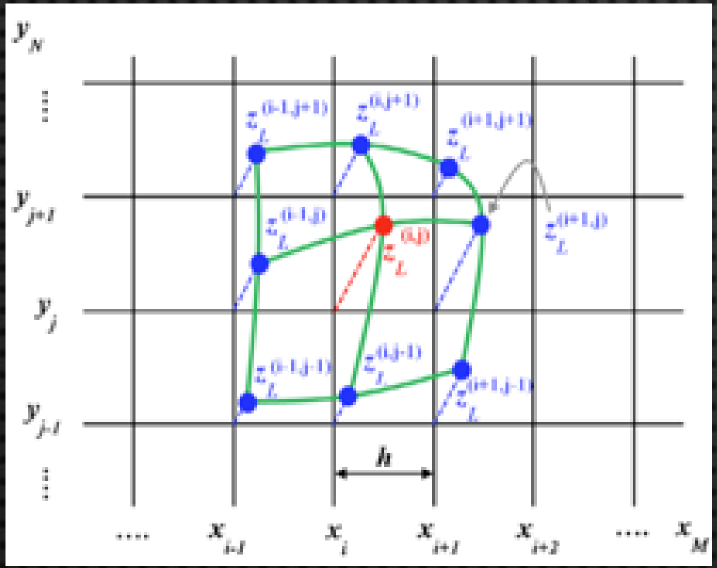
```
function avCalc= avfunc(input1, input2)
avCalc = double(input1 + input2)/2;
end
```



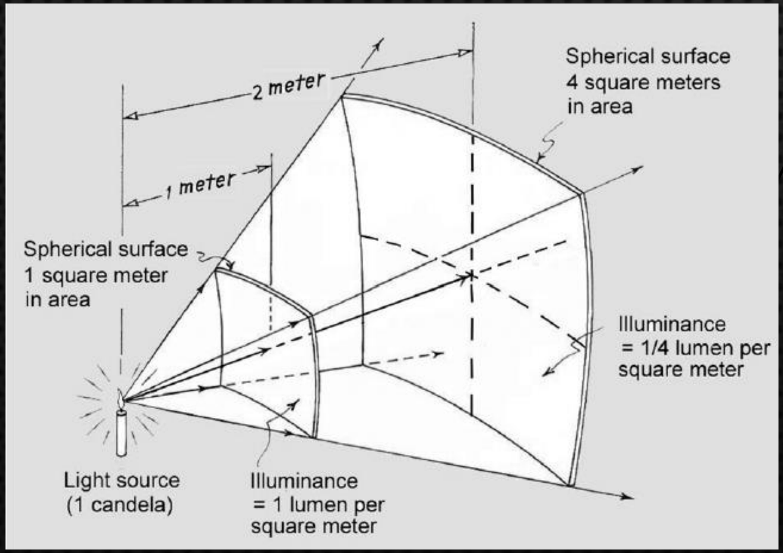
IMPLEMENTATION

$$z_L^{(i,j)} = \frac{av_1^{(i,j)} + av_2^{(i,j)} - \sqrt{\left(av_1^{(i,j)} - av_2^{(i,j)}\right)^2 + \left(\frac{av_3^{(i,j)} - av_4^{(i,j)}}{2}\right)^2 + h^4 g^{(i,j)}}}{2}$$

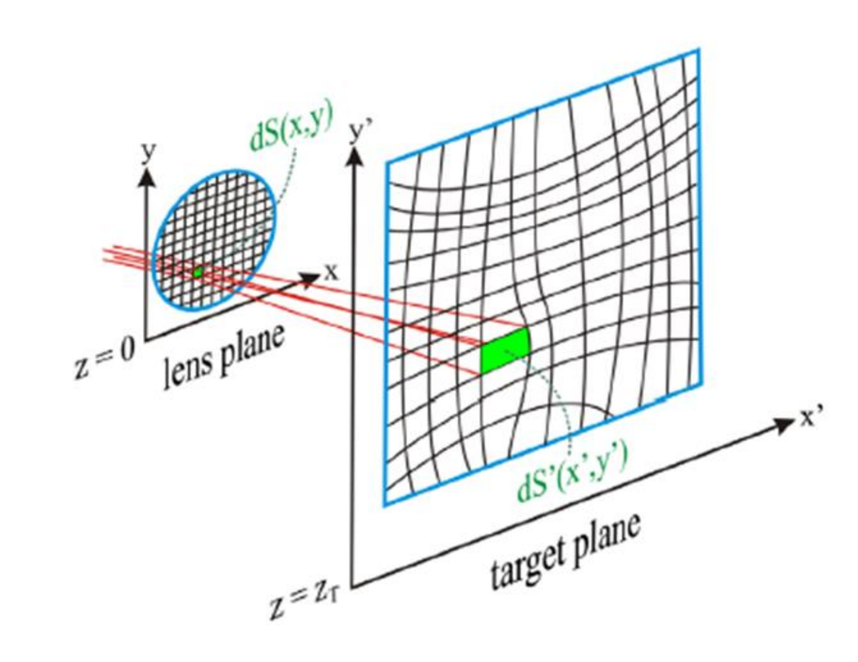




$$g^{(i,j)} = \frac{I_0(x_i, y_i)}{I_T(f_1(x_i, y_i), f_2(x_i, y_i))} \frac{\sqrt{1 + \left(z_{Lx}^{(i,j)2} + z_{Ly}^{(i,j)2}\right)\left(1 - n^2\right)}}{K(z_{Lx}^{(i,j)}, z_{Ly}^{(i,j)})^2}$$

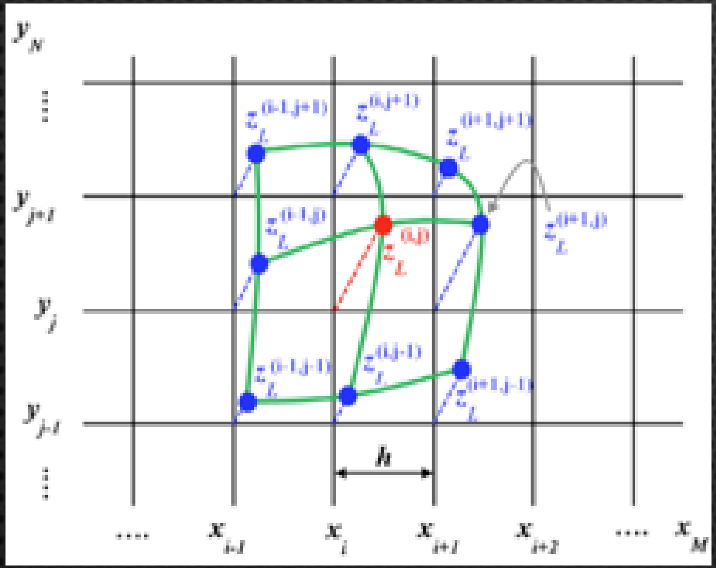


MATRIX

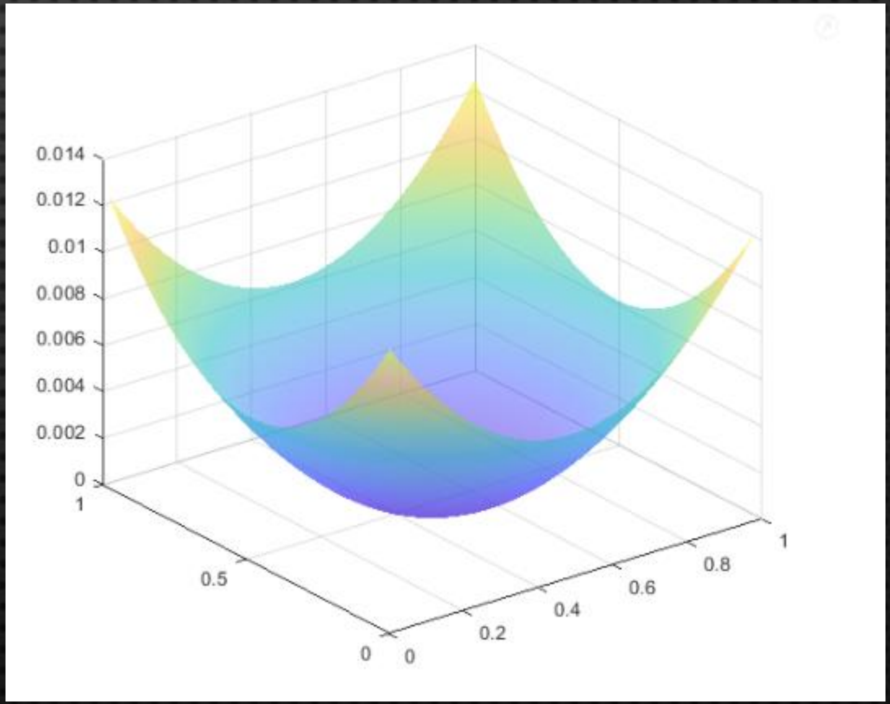


USC Viterbi
School of Engineering

IMPLEMENTATION



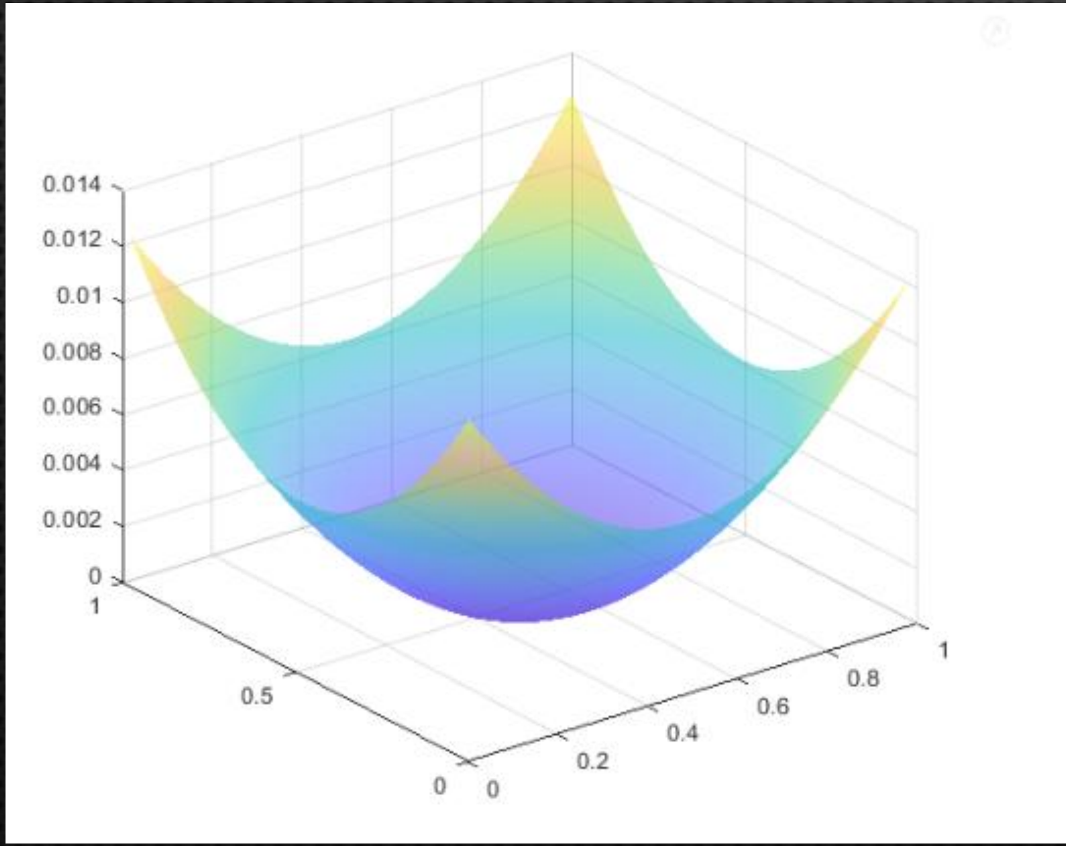
$$z_L^{(i,j)} = \frac{av_1^{(i,j)} + av_2^{(i,j)} - \sqrt{\left(av_1^{(i,j)} - av_2^{(i,j)}\right)^2 + \left(\frac{av_3^{(i,j)} - av_4^{(i,j)}}{2}\right)^2 + h^4 g^{(i,j)}}}{2}$$



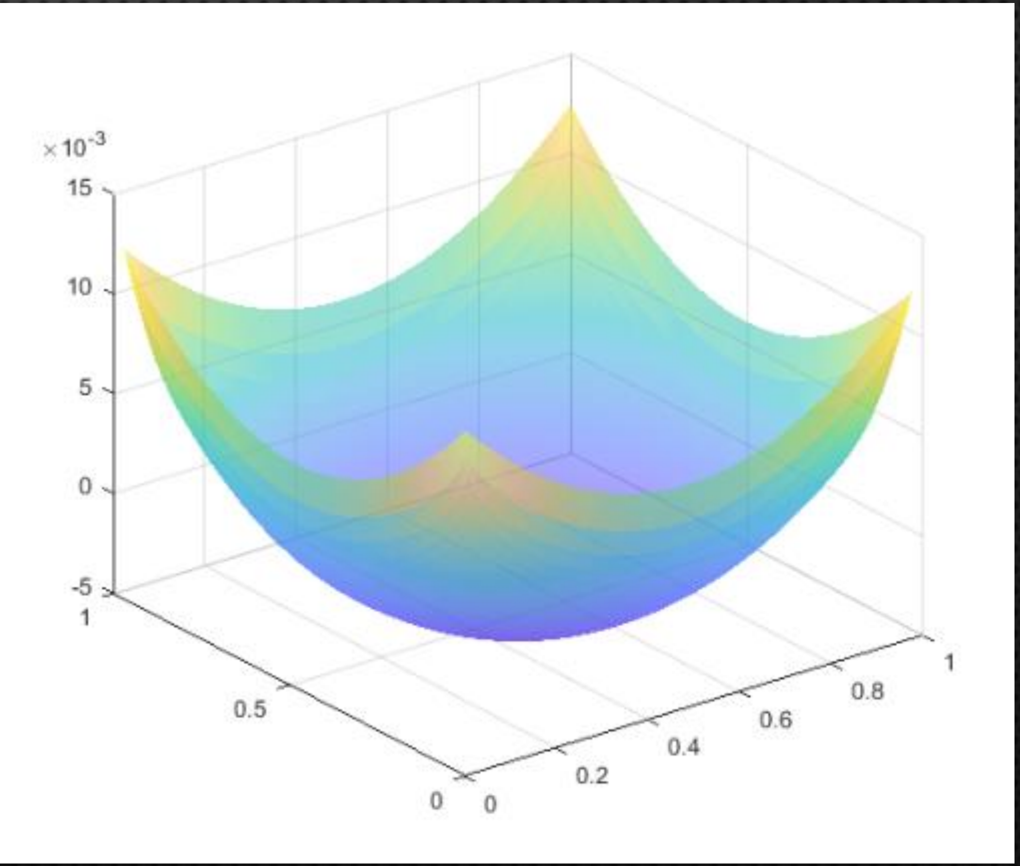
USC Viterbi
School of Engineering

IMPLEMENTATION

Initial Lens:

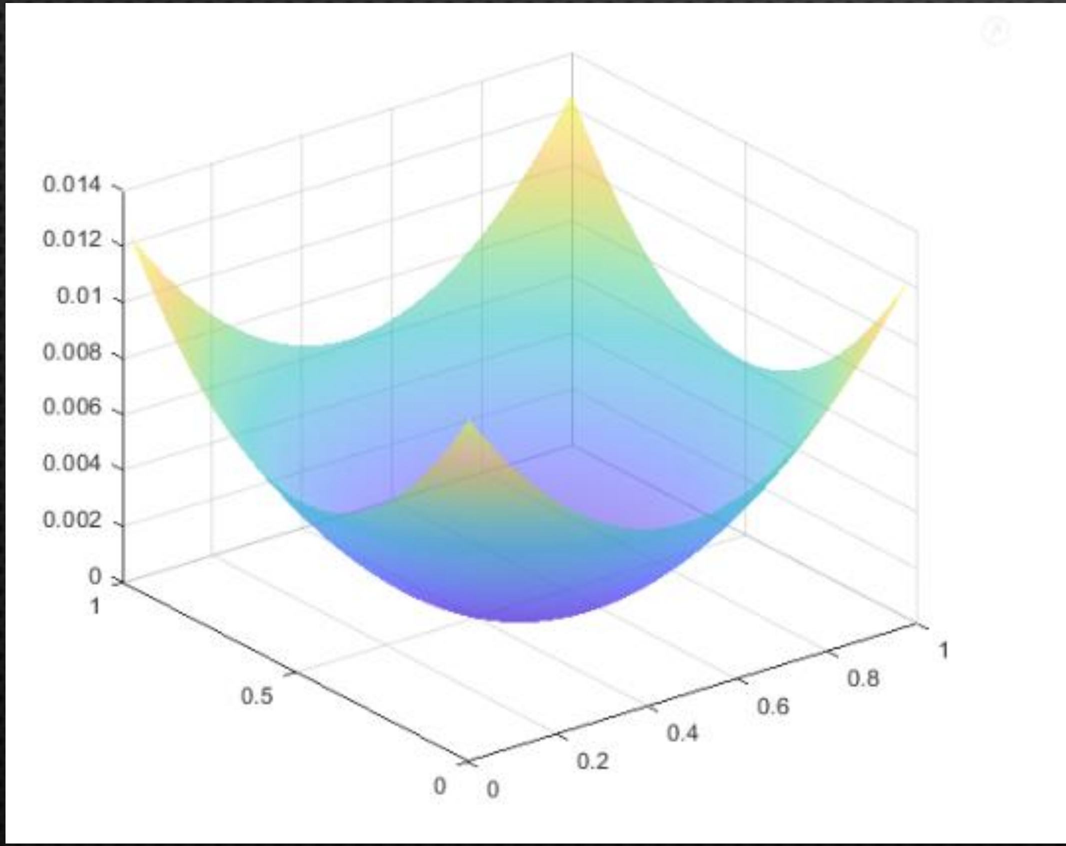


Lens after 25 Iterations:

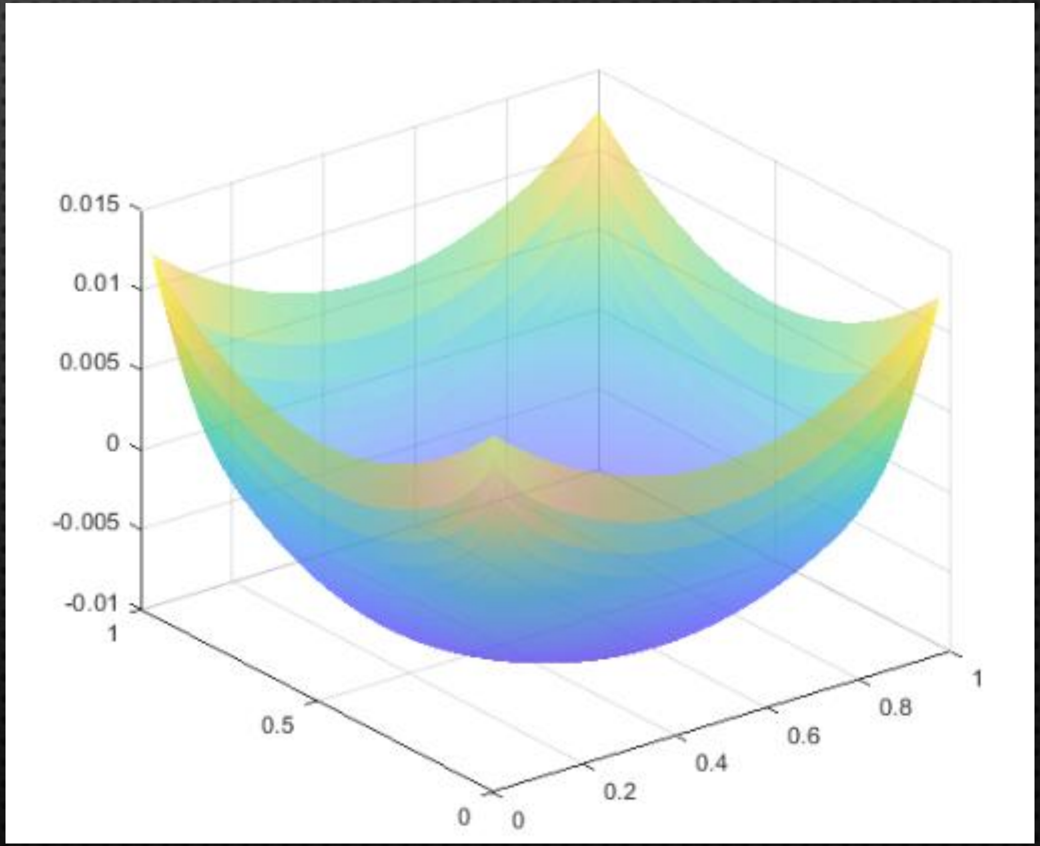


IMPLEMENTATION

Initial Lens:

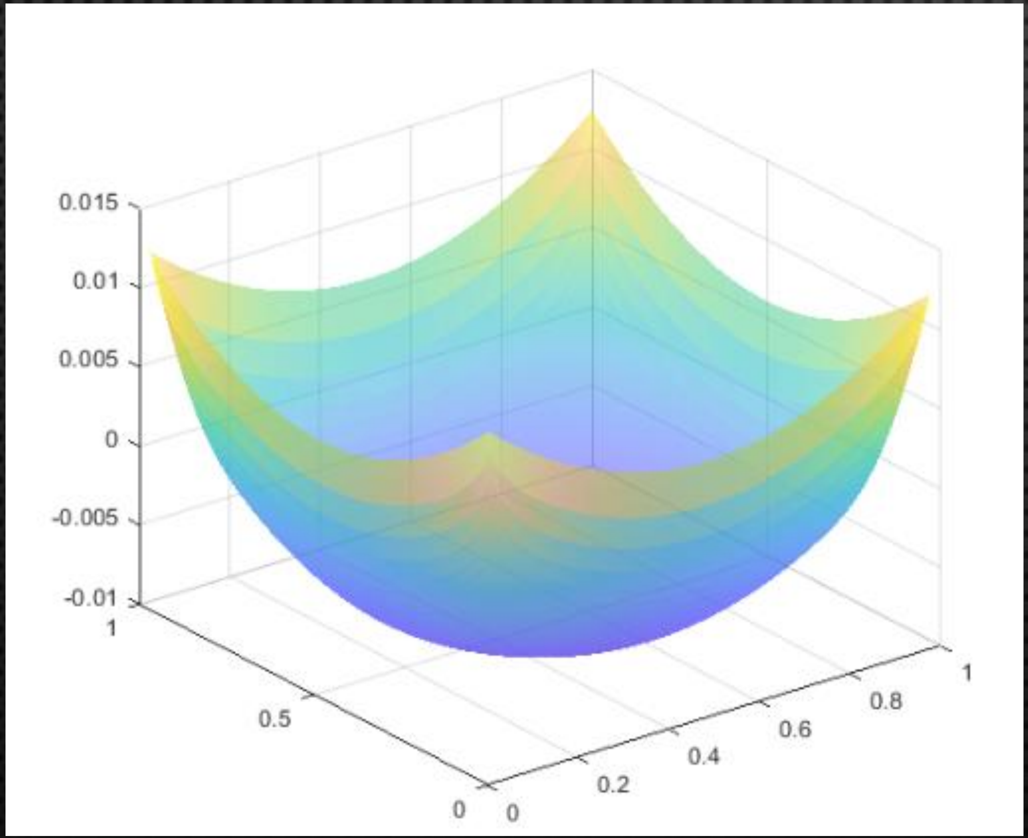


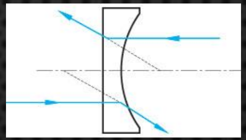
Lens after 50 Iterations:



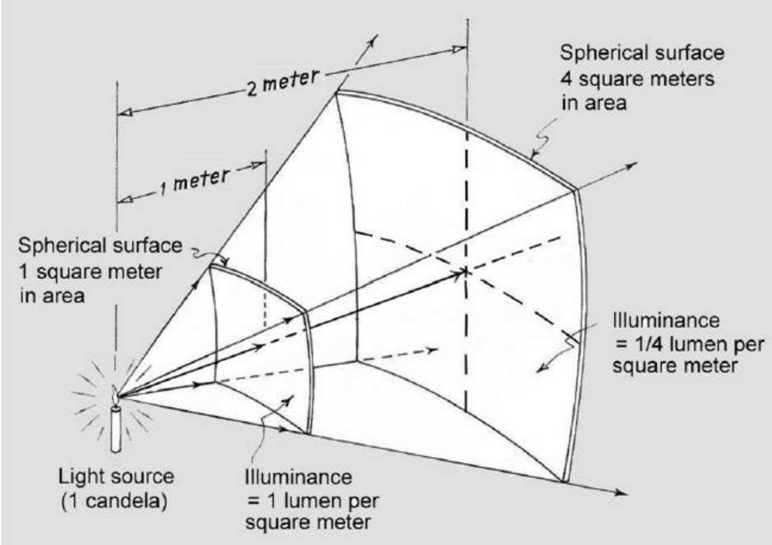
USC Viterbi School of Engineering

IMPLEMENTATION





IMPLEMENTATION - Challenges



$$z_L^{(i,j)} = \frac{av_1^{(i,j)} + av_2^{(i,j)} - \sqrt{\left(av_1^{(i,j)} - av_2^{(i,j)}\right)^2 + \left(\frac{av_3^{(i,j)} - av_4^{(i,j)}}{2}\right)^2 + h^4 g^{(i,j)}}}{2}$$

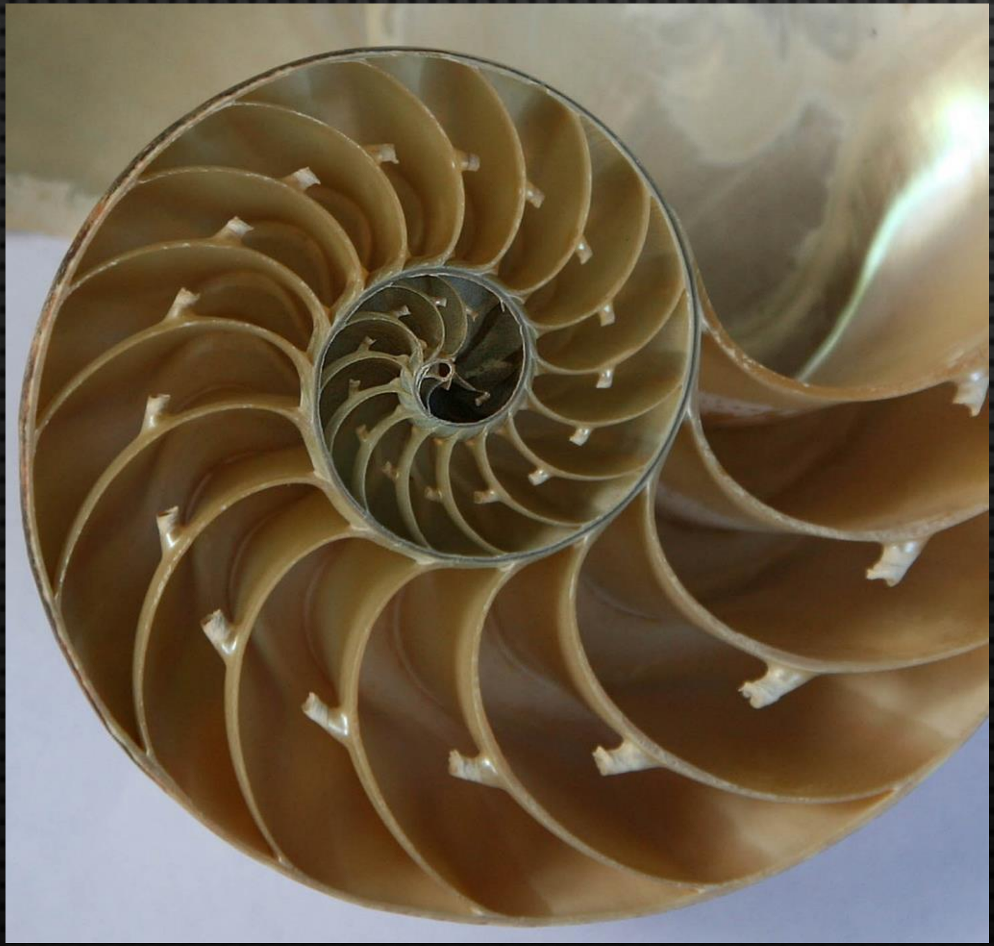
$$g^{(i,j)} = \frac{I_0(x_i, y_i)}{I_T(f_1(x_i, y_i), f_2(x_i, y_i))} \frac{\sqrt{1 + \left(z_{Lx}^{(i,j)2} + z_{Ly}^{(i,j)2}\right)(1 - n^2)}}{K(z_{Lx}^{(i,j)}, z_{Ly}^{(i,j)2})}$$

$$I_T(x', y') \cdot |J(x, y)| = I(x, y)$$

Matrix values 1.0

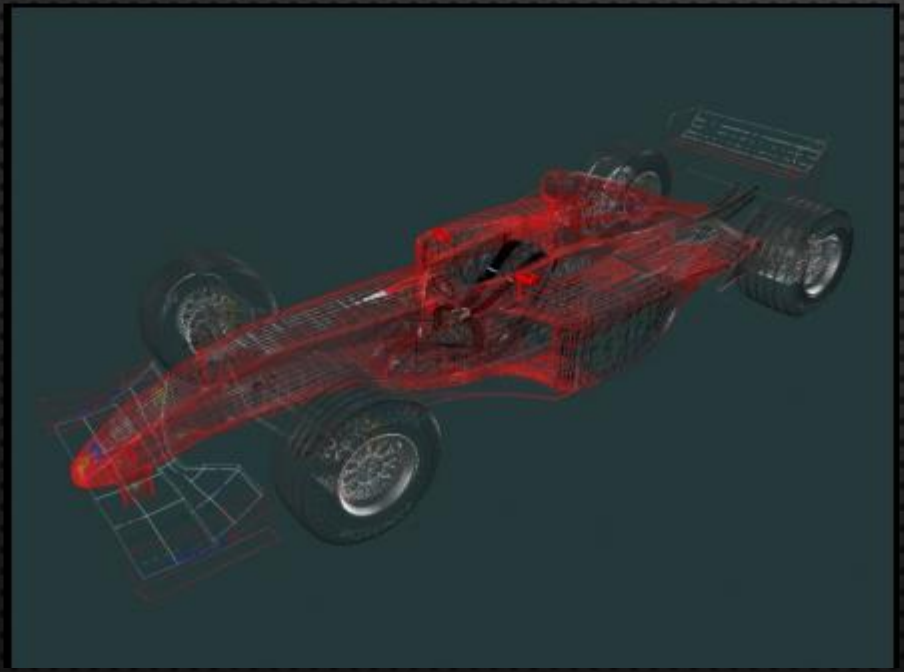
IMPLEMENTATION

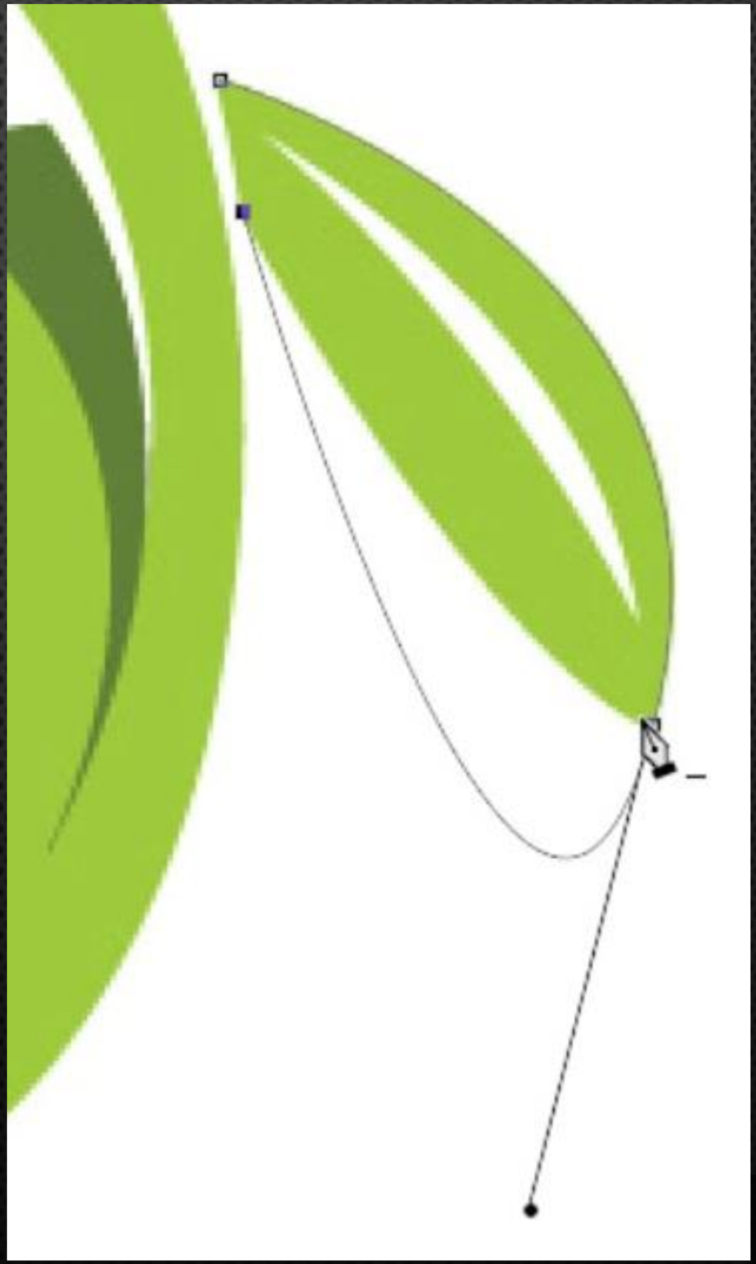
IMPLEMENTATION



USC Viterbi
School of Engineering

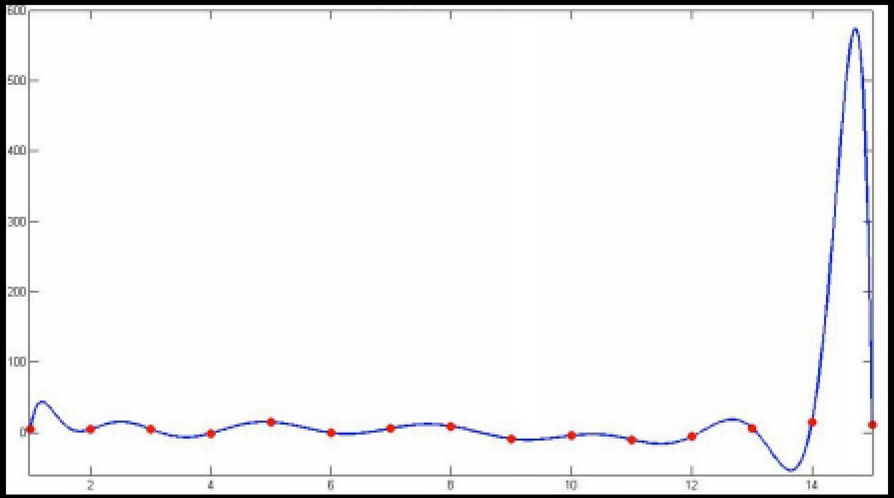
IMPLEMENTATION SPLINES





IMPLEMENTATION SPLINES

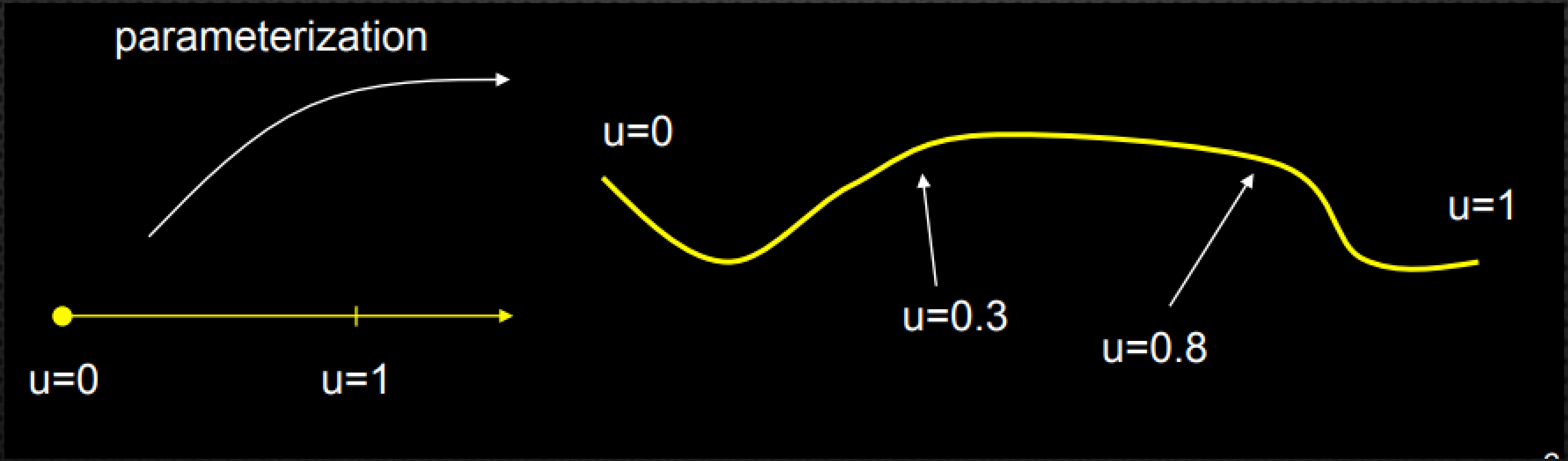
- An *n*-th degree polynomial fits a curve to n+1 points
  - called Lagrange Interpolation
  - result is a curve that is too wiggly, change to any control point affects entire curve (non-local)
  - *this method is poor*
- We usually want the curve to be as smooth as possible
  - minimize the wiggles
  - high-degree polynomials are bad



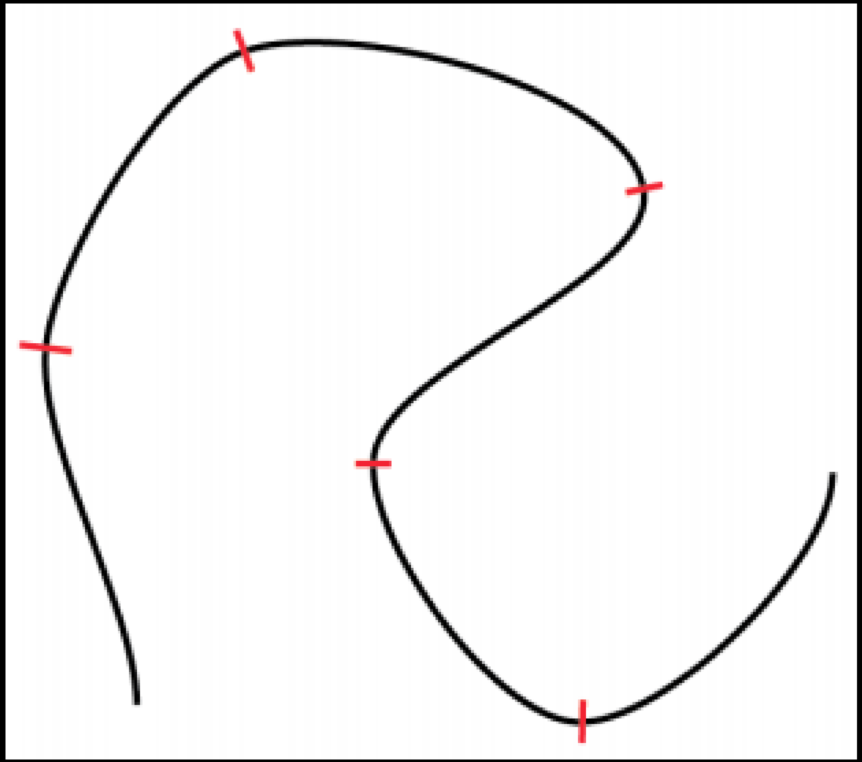
source: Wikipedia

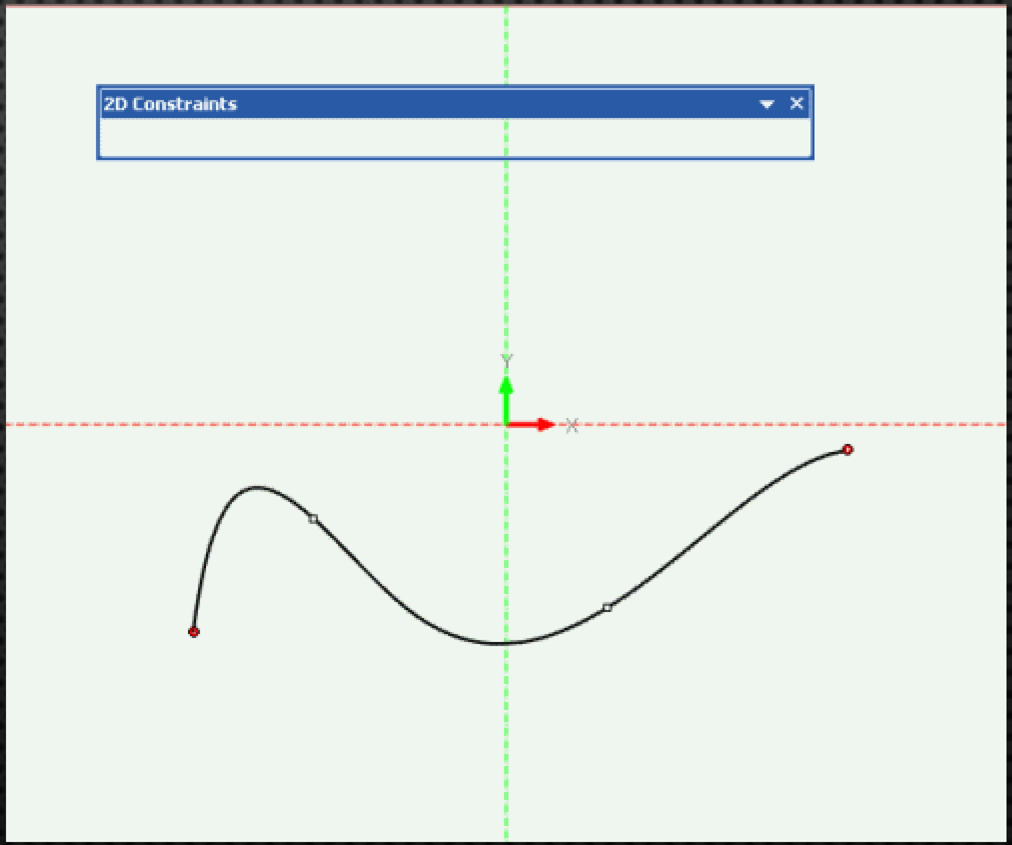
Lagrange interpolation, degree=15

IMPLEMENTATION SPLINES



IMPLEMENTATION SPLINES


a spline



USC Viterbi School of Engineering

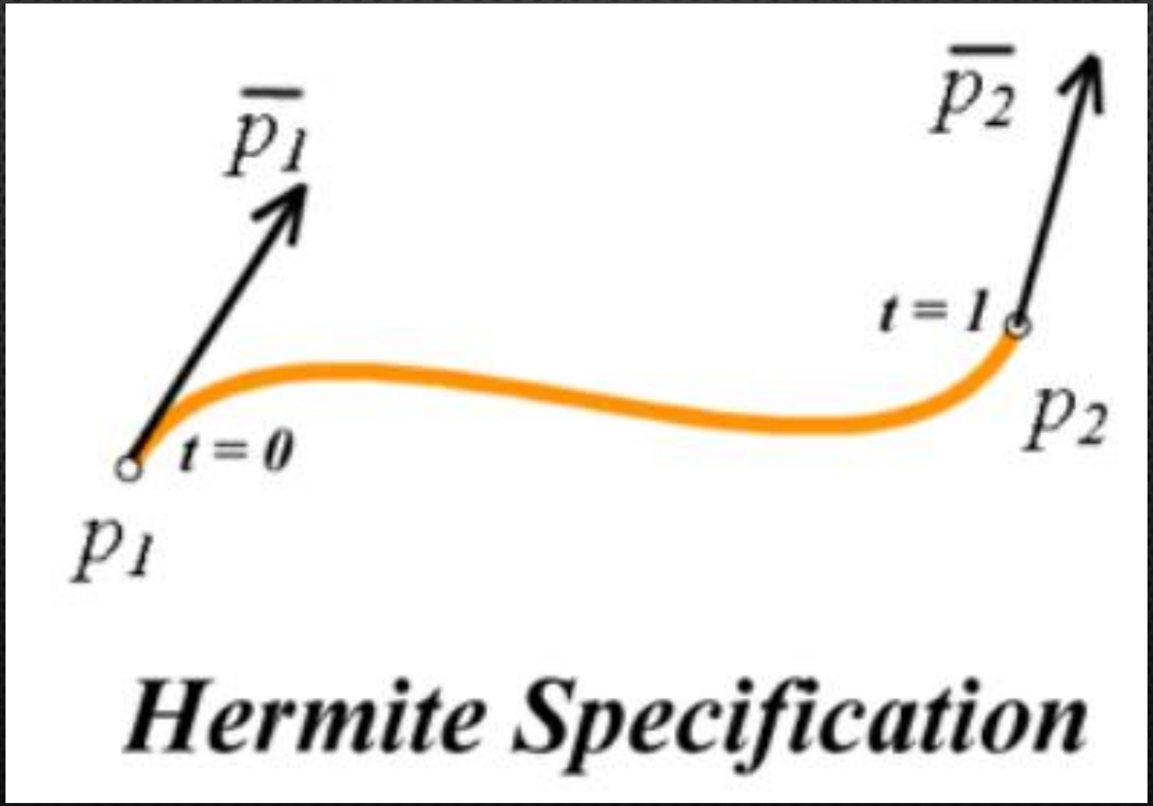
IMPLEMENTATION SPLINES

- Cubic polynomial:
  - $p(u) = au^3+bu^2+cu+d = [u^3 \;\; u^2 \;\; u \;\; 1]\,[a \;\; b \;\; c \;\; d]^T$
  - a,b,c,d are 3-vectors, u is a scalar
- Three cubic polynomials, one for each coordinate:
  - $x(u) = a_x u^3 + b_x u^2 + c_x u + d_x$
  - $y(u) = a_y u^3 + b_y u^2 + c_y u + d_y$
  - $z(u) = a_z u^3 + b_z u^2 + c_z u + d_z$
- In matrix notation:

$$\begin{bmatrix} x(u) & y(u) & z(u) \end{bmatrix} = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \begin{bmatrix} a_x & a_y & a_z \\ b_x & b_y & b_z \\ c_x & c_y & c_z \\ d_x & d_y & d_z \end{bmatrix}$$

- Or simply: $p = [u^3 \; u^2 \; u \; 1]\, A$

IMPLEMENTATION – Hermite SPLINES



IMPLEMENTATION – Hermite SPLINES

- Four constraints: value and slope (in 3-D, position and tangent vector) at beginning and end of interval [0,1] :

$$p(0) = p_1 = (x_1, y_1, z_1)$$
$$p(1) = p_2 = (x_2, y_2, z_2)$$
$$p'(0) = \bar{p}_1 = (\bar{x}_1, \bar{y}_1, \bar{z}_1)$$
$$p'(1) = \bar{p}_2 = (\bar{x}_2, \bar{y}_2, \bar{z}_2)$$

the user constraints

- Assume cubic form: $p(u) = au^3 + bu^2 + cu + d$
- Four unknowns: a, b, c, d

USC Viterbi School of Engineering

IMPLEMENTATION – Hermite SPLINES

- Assume cubic form: $p(u) = au^3 + bu^2 + cu + d$

$$p_1 = p(0) = d$$

$$p_2 = p(1) = a + b + c + d$$

$$\overline{p_1} = p'(0) = c$$

$$\overline{p_2} = p'(1) = 3a + 2b + c$$

- Linear system: 12 equations for 12 unknowns
(however, can be simplified to 4 equations for 4 unknowns)
- Unknowns: *a, b, c, d* (each of *a*, *b*, *c*, *d* is a 3-vector)

IMPLEMENTATION – Hermite SPLINES

$$d = p_1$$
$$a + b + c + d = p_2$$
$$c = \overline{p}_1$$
$$3a + 2b + c = \overline{p}_2$$

Rewrite this 12x12 system as a 4x4 system:

$$\begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix} \begin{bmatrix} a_x & a_y & a_z \\ b_x & b_y & b_z \\ c_x & c_y & c_z \\ d_x & d_y & d_z \end{bmatrix} = \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ \overline{x}_1 & \overline{y}_1 & \overline{z}_1 \\ \overline{x}_2 & \overline{y}_2 & \overline{z}_2 \end{bmatrix}$$

USC Viterbi School of Engineering

IMPLEMENTATION – Hermite SPLINES

- After inverting the 4x4 matrix, we obtain:

$$\begin{bmatrix} x & y & z \end{bmatrix} = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ \bar{x}_1 & \bar{y}_1 & \bar{z}_1 \\ \bar{x}_2 & \bar{y}_2 & \bar{z}_2 \end{bmatrix}$$

point on the spline — parameter vector — basis — control matrix (what the user gets to pick)

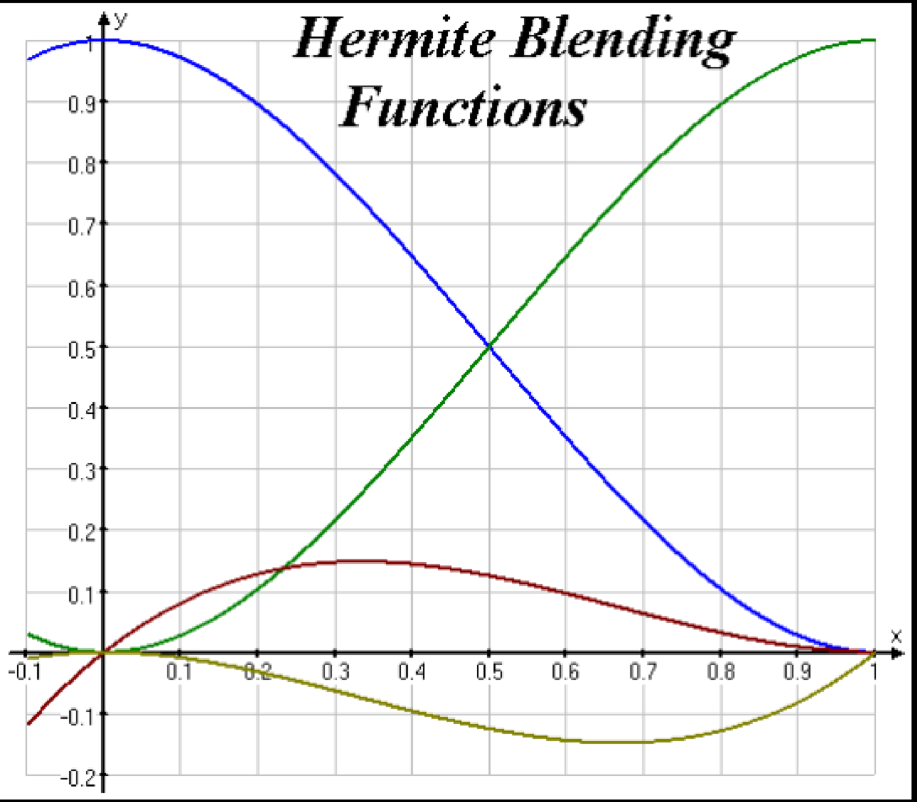
- This form is typical for splines
  - basis matrix and meaning of control matrix change with the spline type

IMPLEMENTATION – Hermite SPLINES

transpose

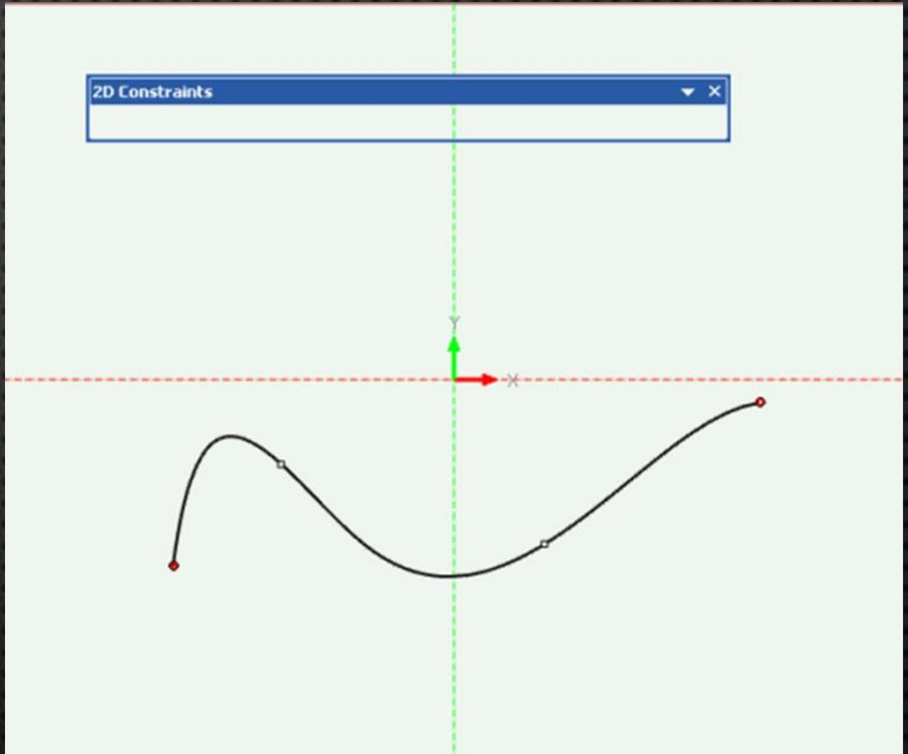
$$p(u) = \begin{bmatrix} 2u^3 - 3u^2 + 1 \\ -2u^3 + 3u^2 \\ u^3 - 2u^2 + u \\ u^3 - u^2 \end{bmatrix}^T \begin{bmatrix} p_1 \\ p_2 \\ \bar{p}_1 \\ \bar{p}_2 \end{bmatrix}$$

4 Basis Functions



Every cubic Hermite spline is a linear combination (blend) of these 4 functions.

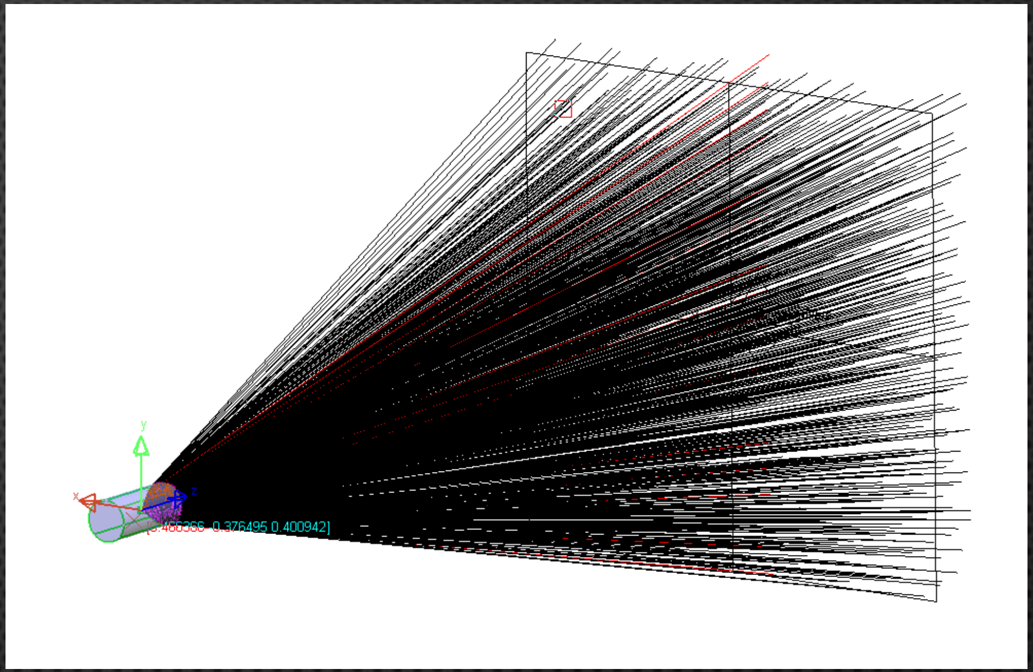
IMPLEMENTATION – B-Splines

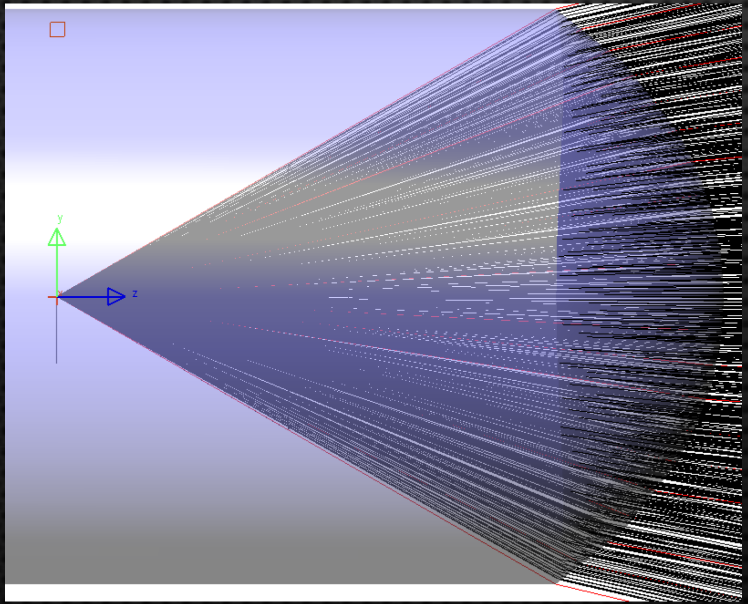


USC Viterbi
School of Engineering

IMPLEMENTATION

LIGHT TOOLS





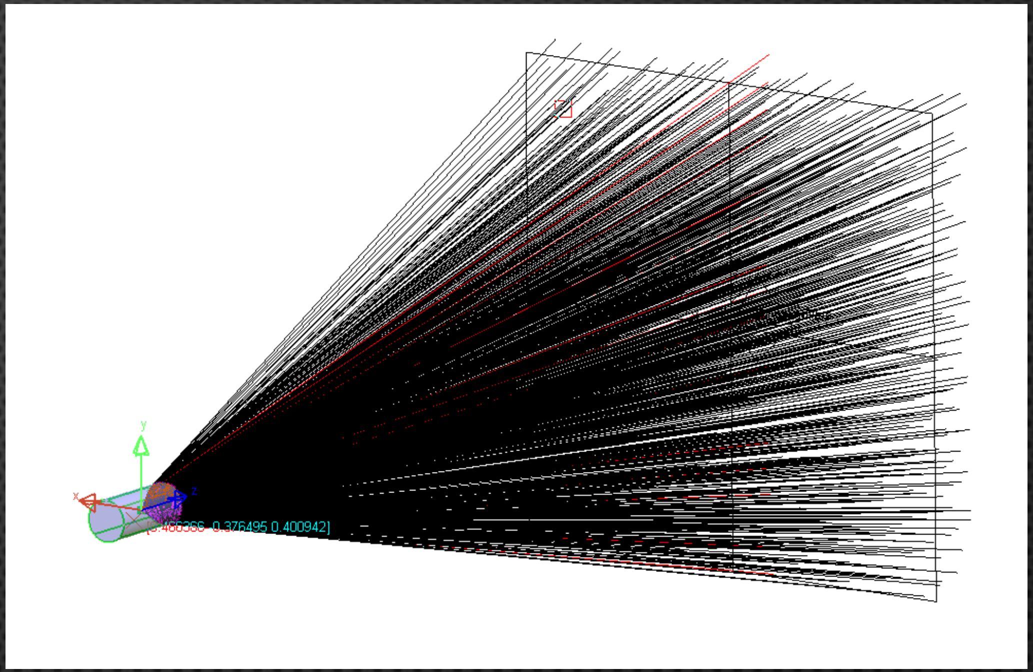
LENS SIZE = 1MM X 1MM | | | D0 1MM

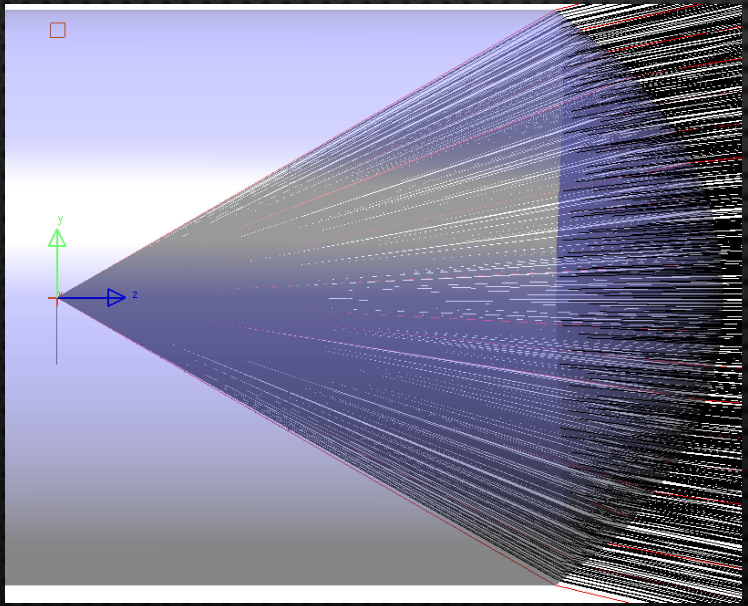
TARGET DISTANCE ZT= 2.0MM

TARGET MAX DIMENSIONS = 20MM X 20MM | TMAX = 10MM

IMPLEMENTATION

LIGHT TOOLS



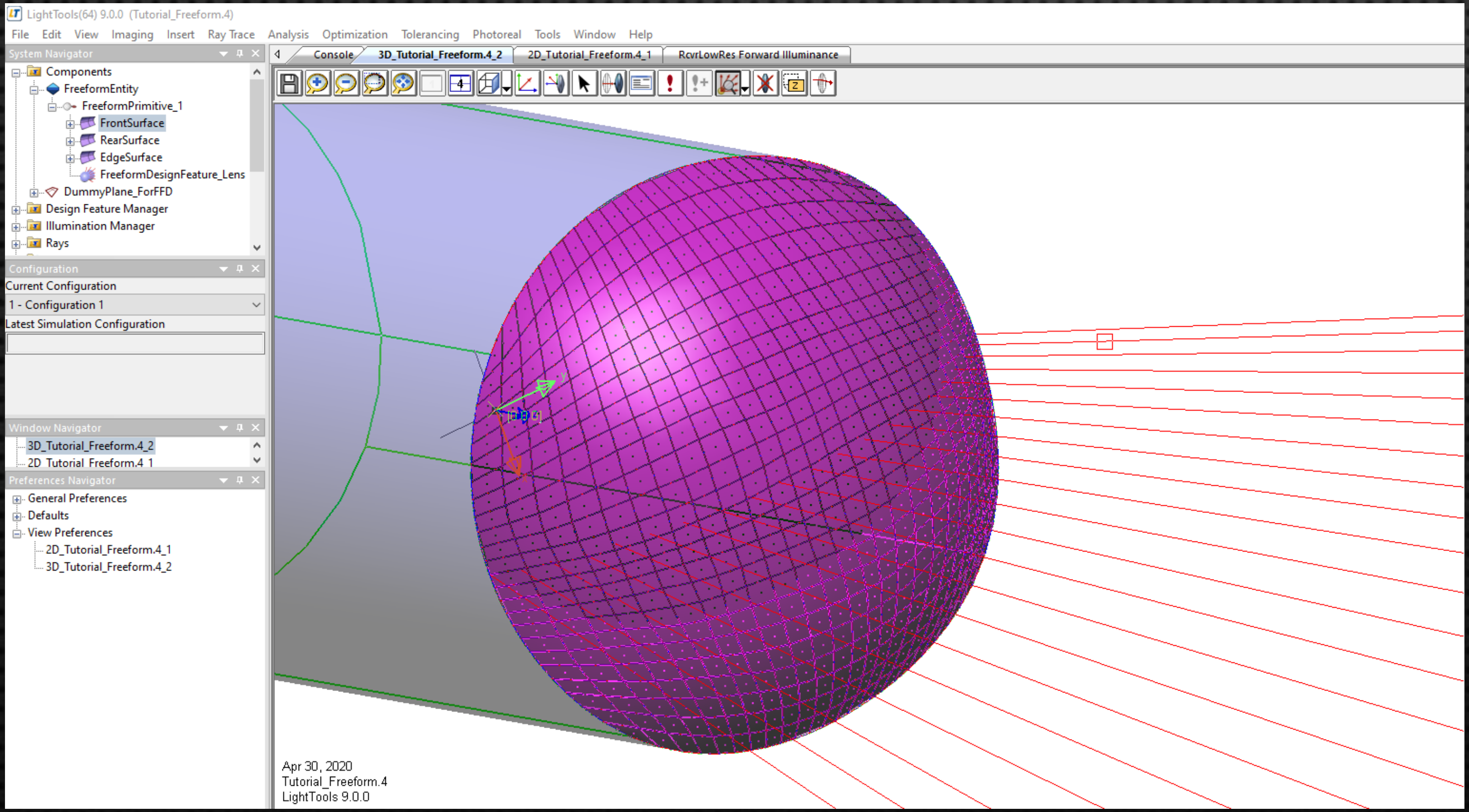


Lens Size = 1mm x 1mm | | | D0 1mm

Target distance Zt= 2.0mm

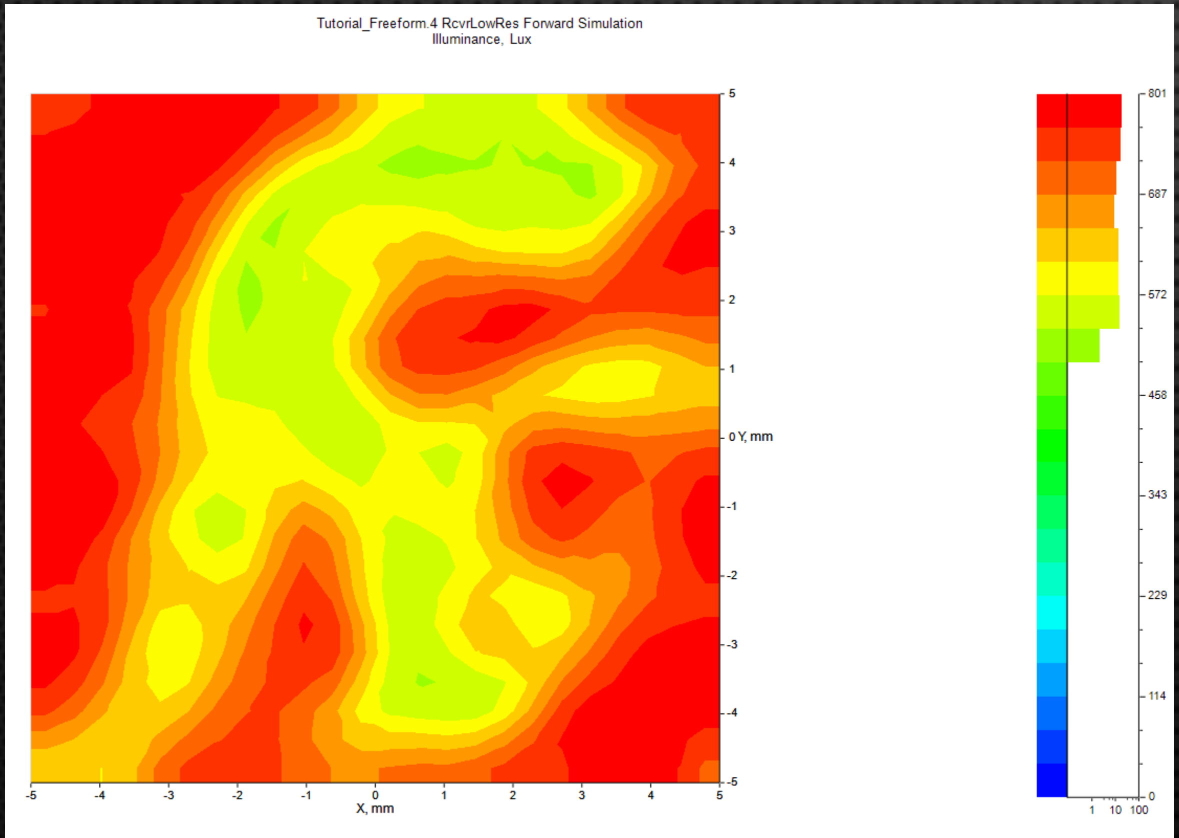
Target max dimensions = 20mm x 20mm | Tmax = 10mm

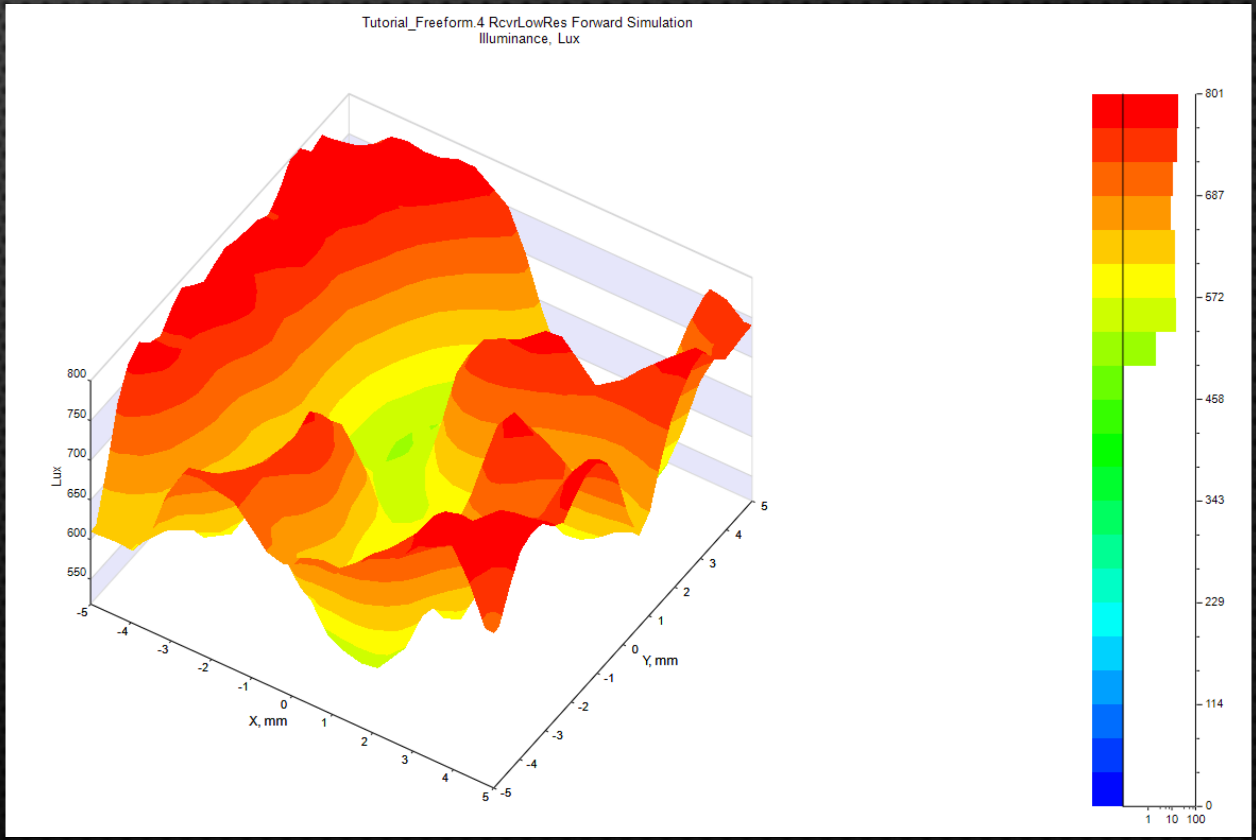
IMPLEMENTATION



USC Viterbi
School of Engineering

IMPLEMENTATION

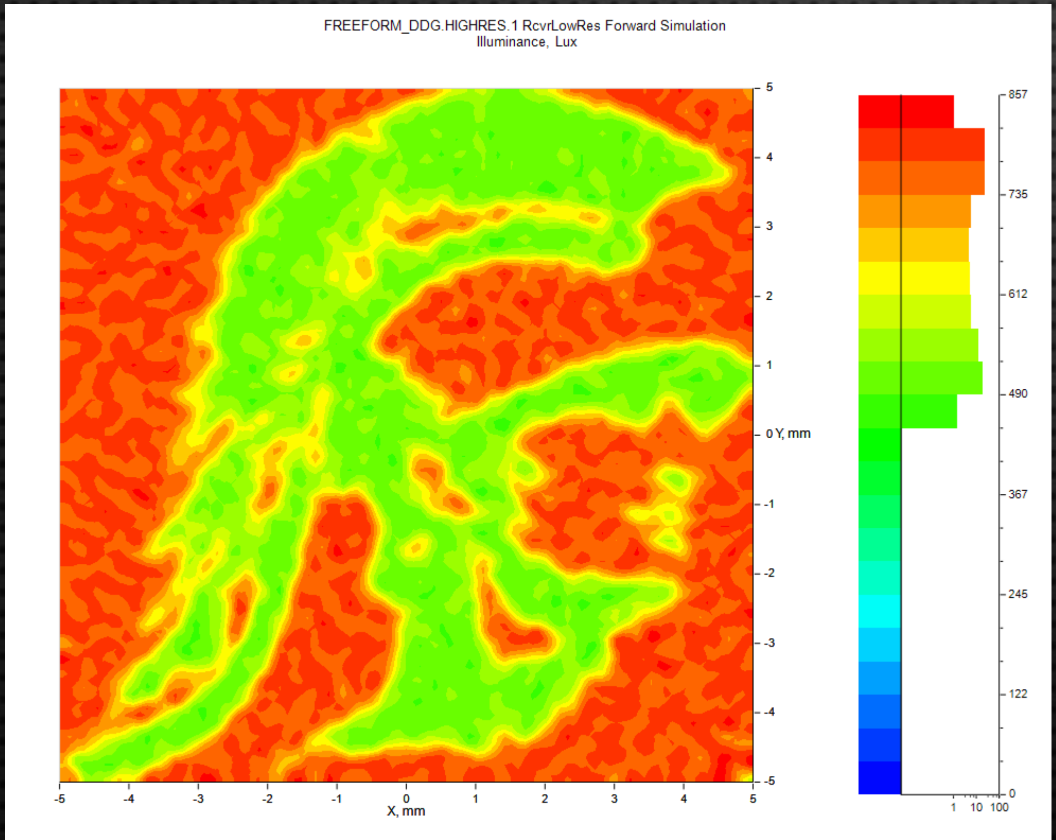


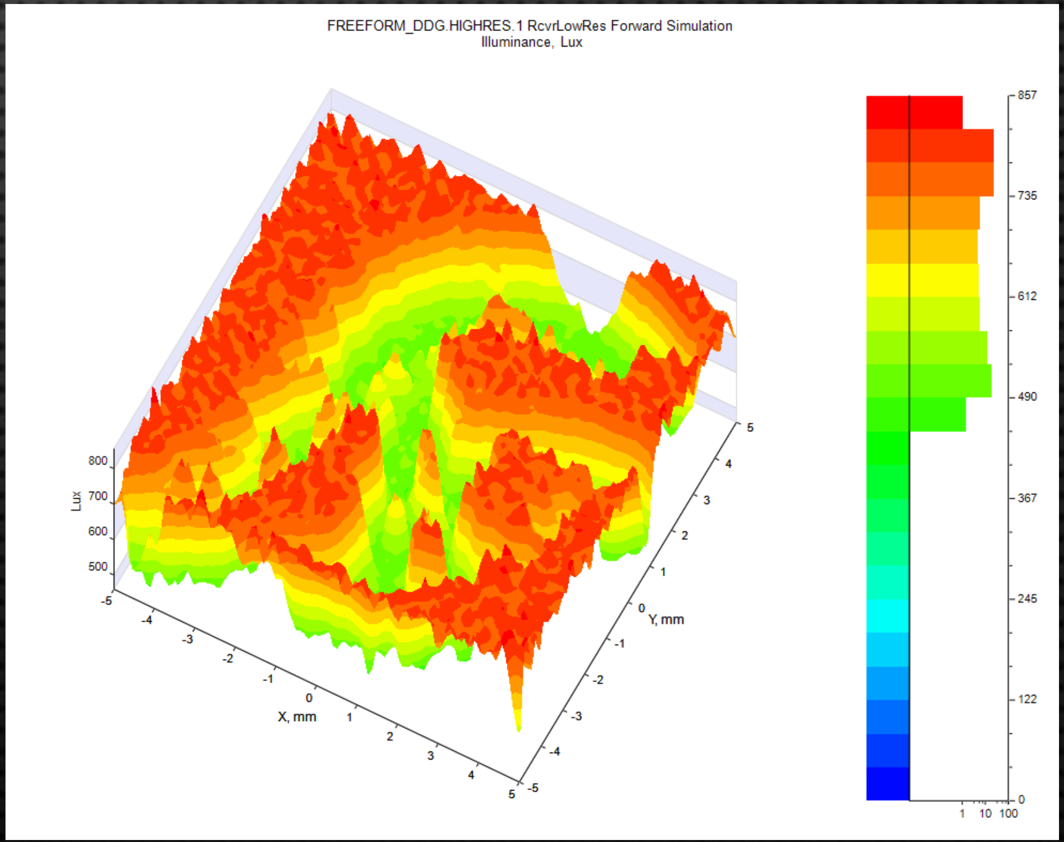


Discretizing the lens surface into 25x25 samples

USC Viterbi
School of Engineering

IMPLEMENTATION





Discretizing the lens surface into 100x100 samples

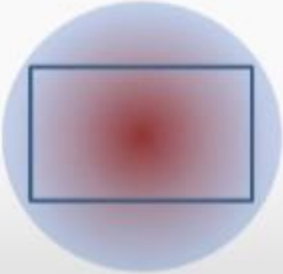
USC Viterbi School of Engineering

Concluding Remarks

Fact: Rotationally symmetrical lenses optimize image quality in a circle

The Opportunity: Optimize image quality almost <u>any way you want</u>

(within the laws of physics of course...)



Today: Image Quality from Conventional Rotationally Symmetric Lens

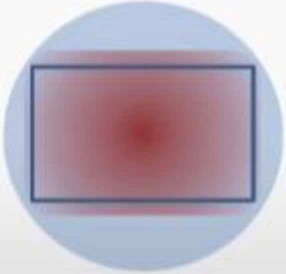
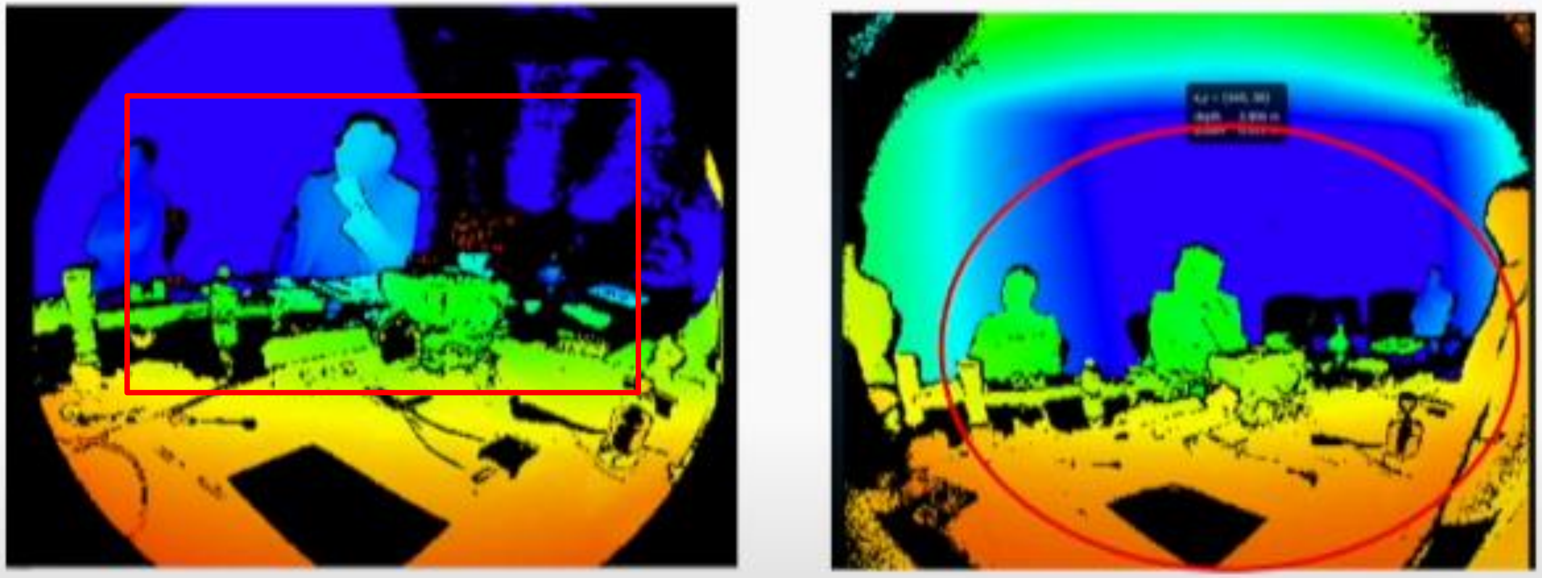


Image Quality Optimized with using free-form lenses

USC Viterbi
School of Engineering

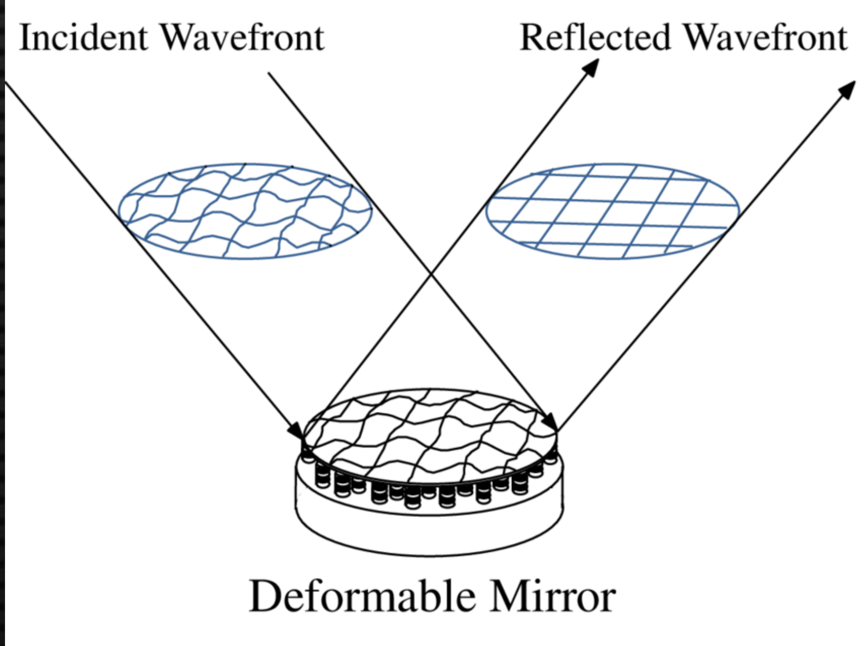
Concluding Remarks

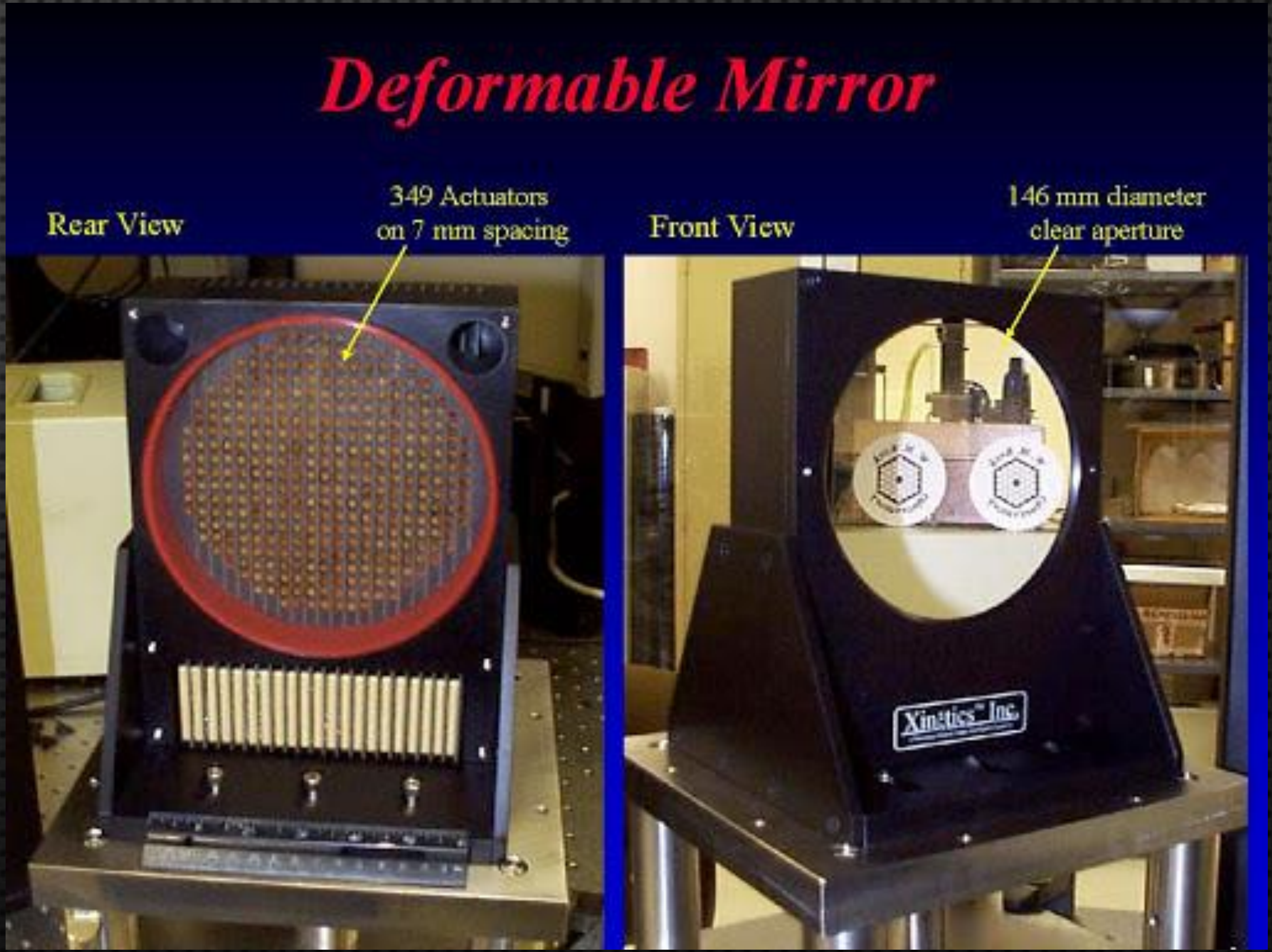


100 deg horizontal | 120 deg horizontal

Concluding Remarks

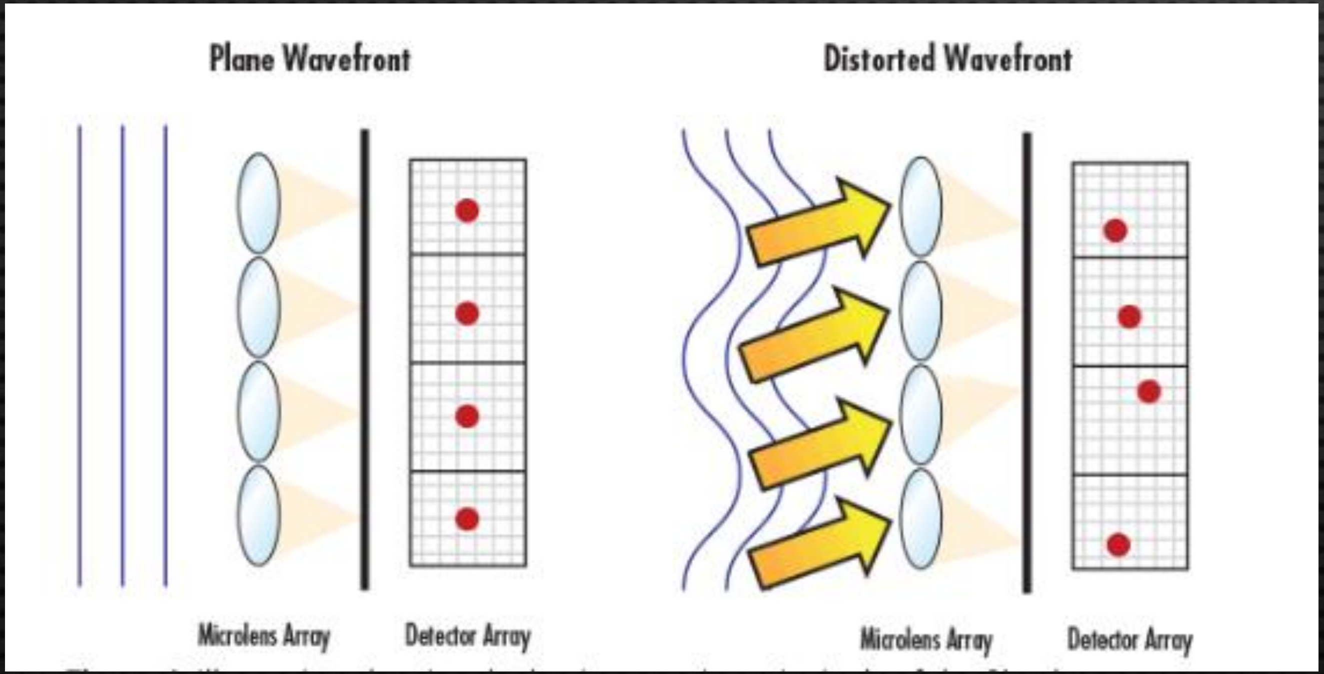
Concluding Remarks

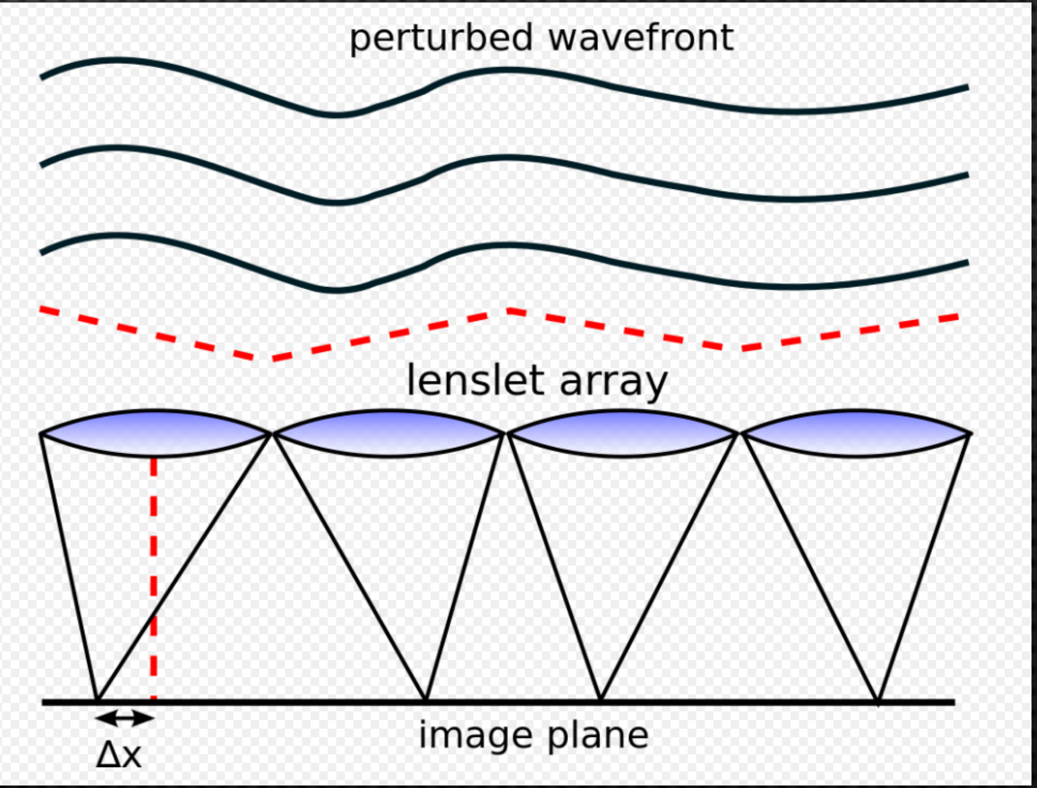




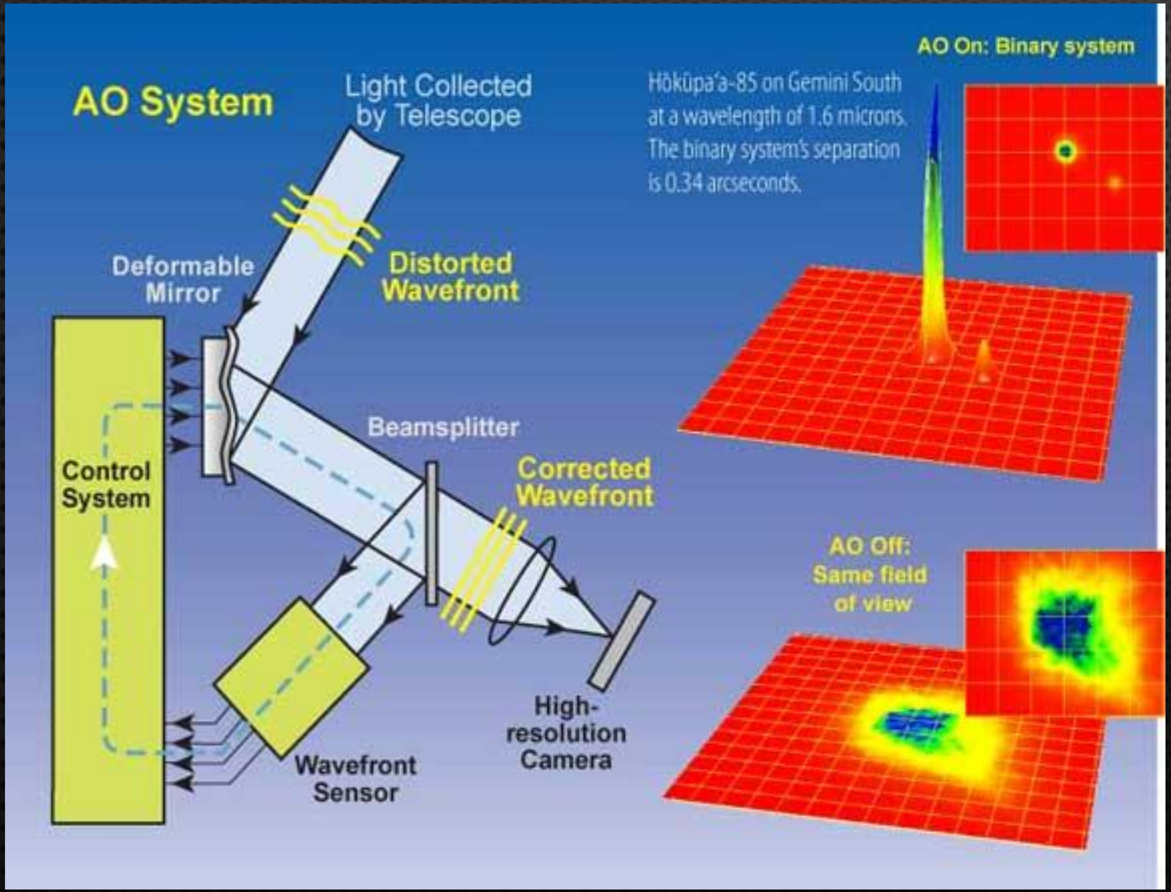
USC Viterbi
School of Engineering

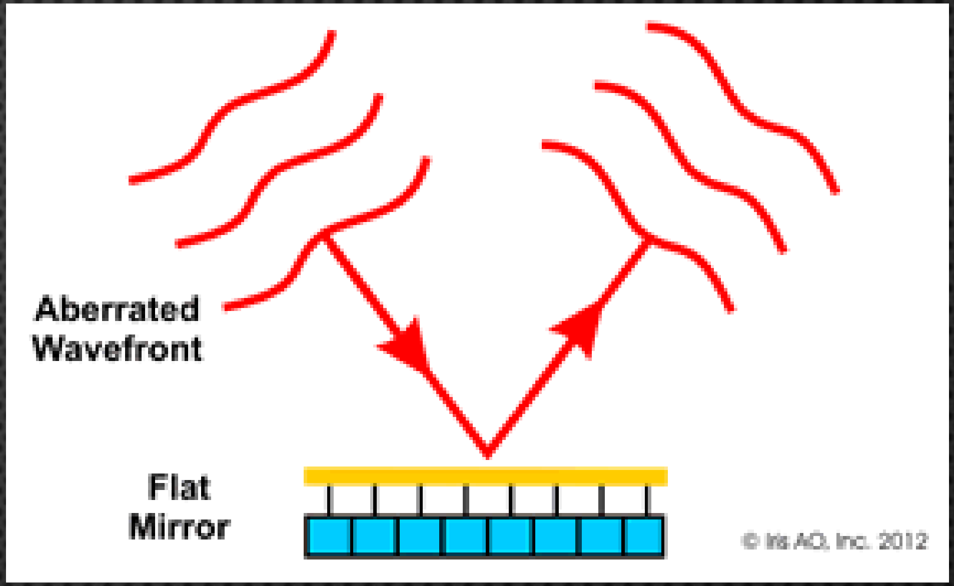
Concluding Remarks

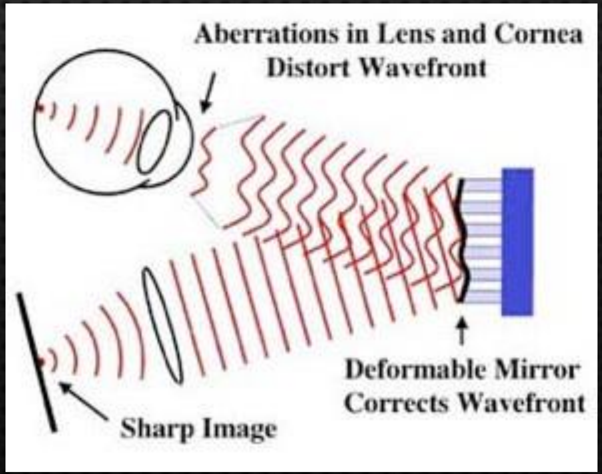




Concluding Remarks

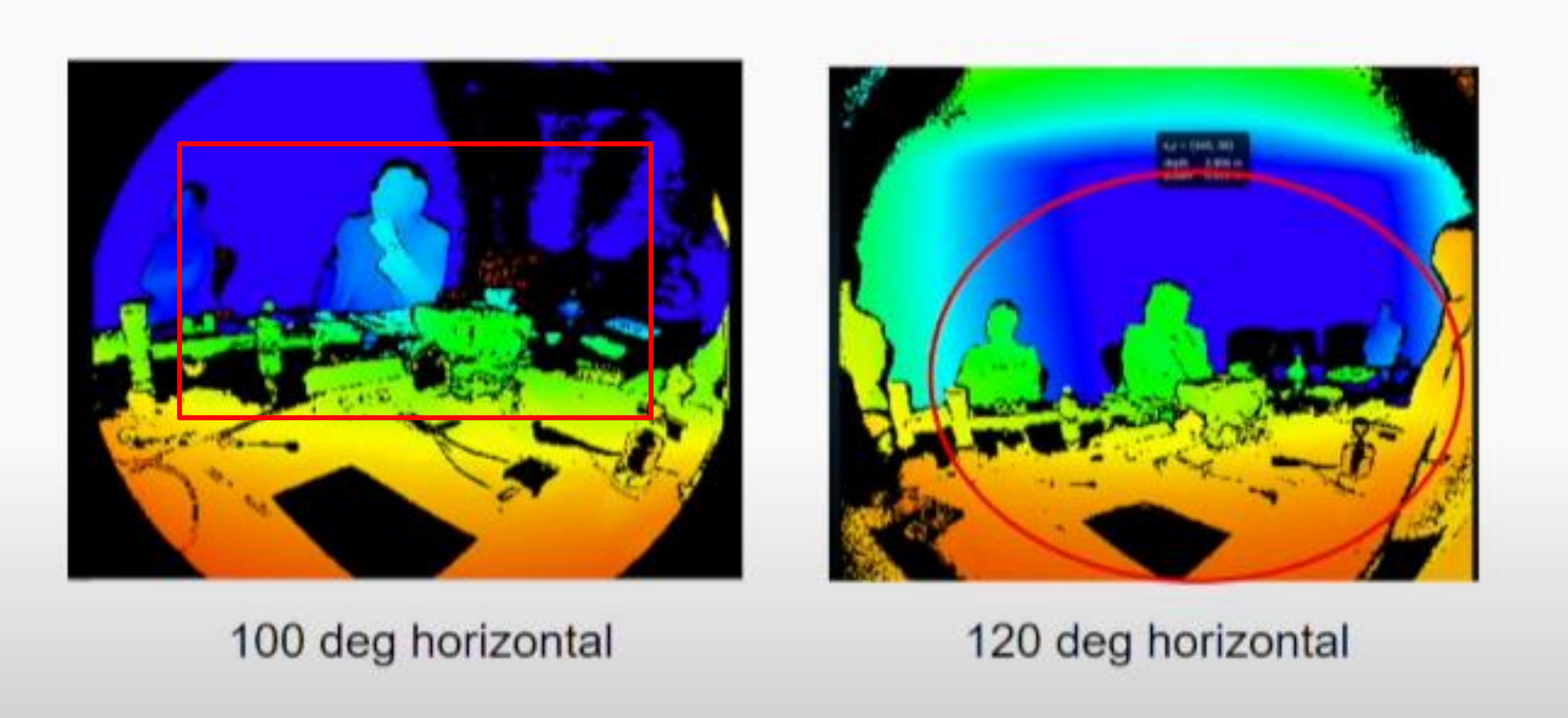






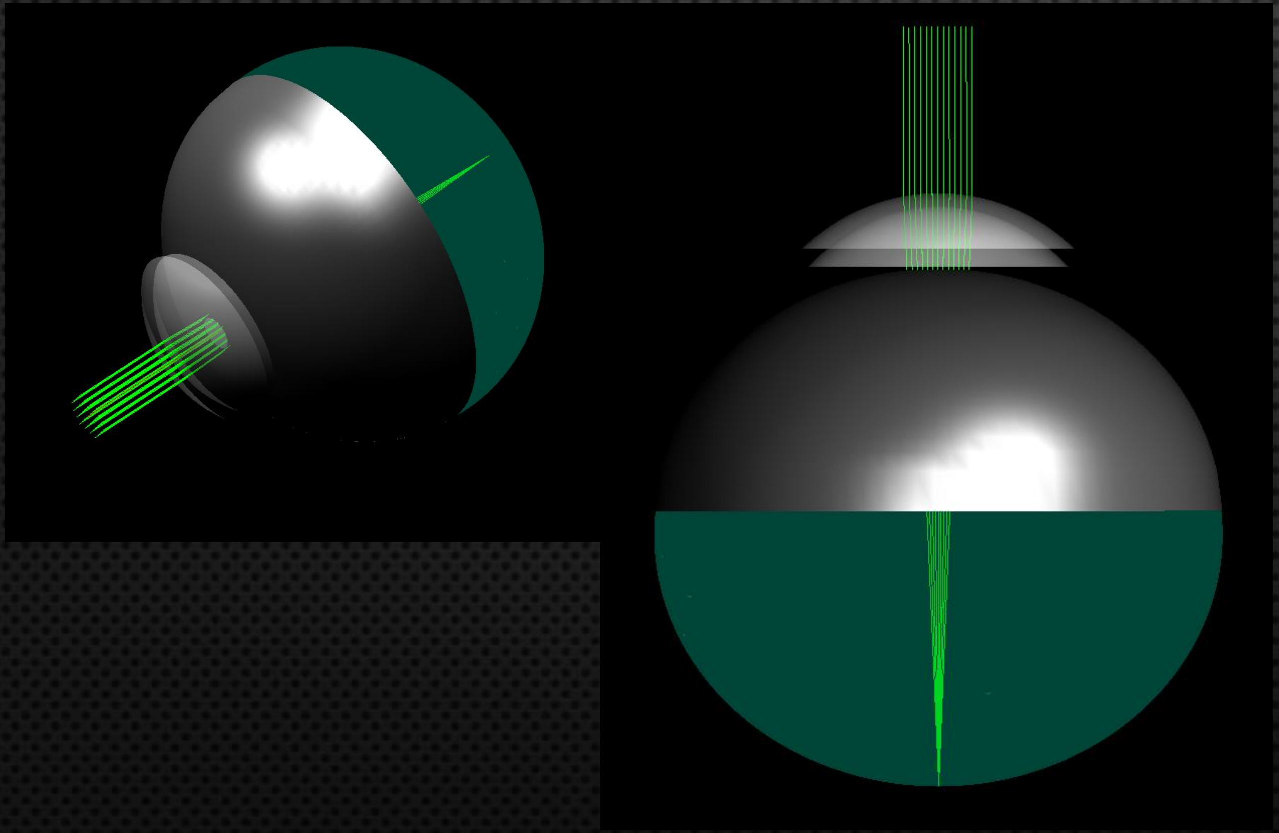
USC Viterbi School of Engineering

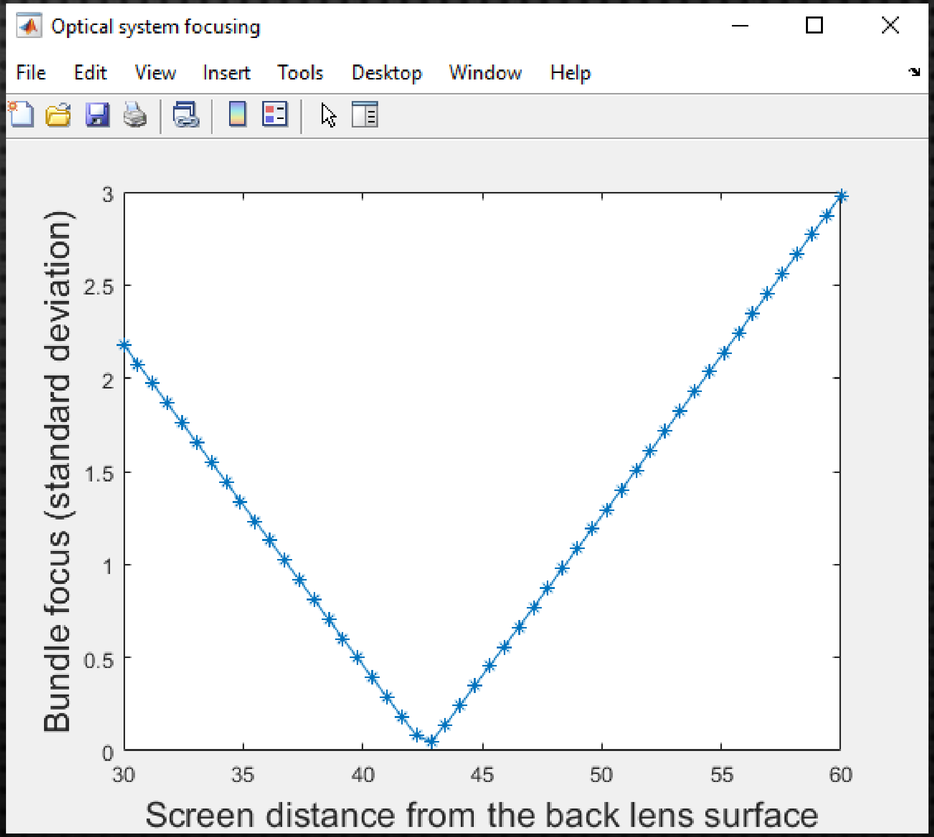
Concluding Remarks



USC Viterbi School of Engineering

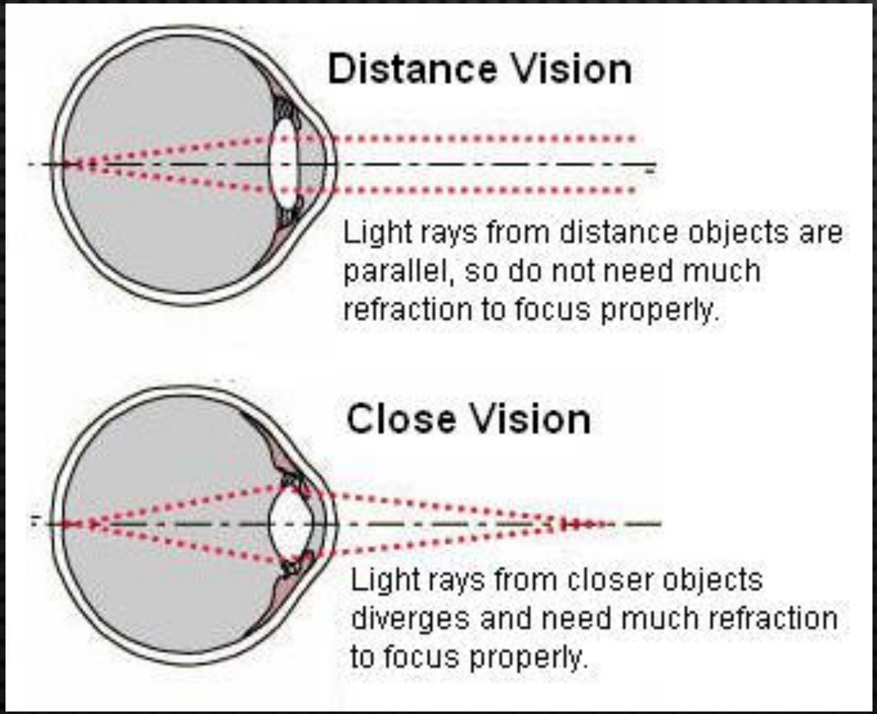
CONCLUSIONS

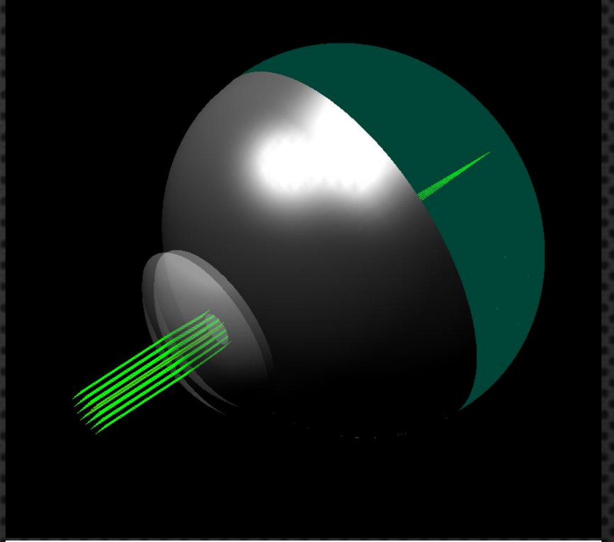


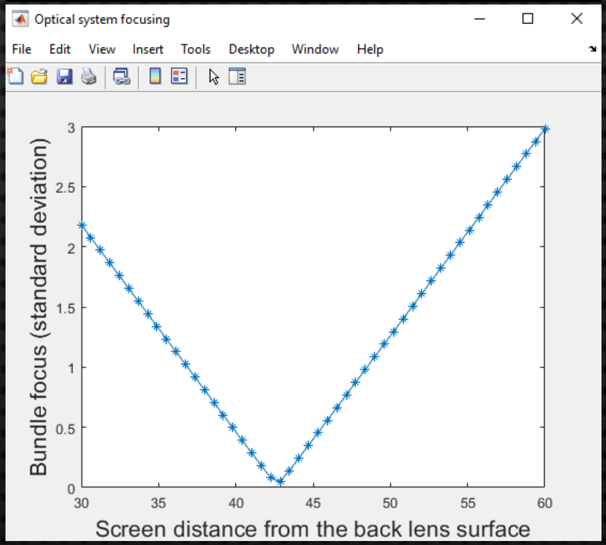


USC Viterbi
School of Engineering

CONCLUSIONS







USC Viterbi
School of Engineering

Q & A

END

USC Viterbi School of Engineering