
Multiple rays per point %% clear; close all; clc;

Table of Contents

System	1
Object 0	1
Objects at 1 and -1	1
Objects at 2 and -2	2
Move lens	4
Object 0	4
Object 1 and -1	4
Object 2 and -2	5
Propogation of shifted lens	5
Up by 0.6	5
Up by 1.2	6
Up by 1.8	8
Plotting blurred images for each instance	9
Refocusing (back tracing)	13

System

```
F1 = 0.6; %cm
d2 = F1;
```

Object 0

```
d0 = 800; %cm
lens_height = 1.8; %3.6/2 cm (half above orgin half below)
halfTheta0 = atan(lens_height/d0);
theta0 = linspace(-1*halfTheta0, halfTheta0, 50);
zeroPoints = zeros(1,50);
```

Objects at 1 and -1

```
d1 = 700; %cm
dNeg1 = 900; %cm
lens_PosHeight = 0.8; %3.6/2 cm (1.8-1 above orgin, rest below zero)
lens_NegHeight = 2.8;
postTheta1 = atan(lens_PosHeight/d1);
negTheta1 = atan(lens_NegHeight/d1);
theta1 = linspace(-negTheta1, postTheta1, 50); %valid angles for (1,1)
onePoints = ones(1,50);
%negative one
postThetan1 = atan(lens_PosHeight/dNeg1);
negThetan1 = atan(lens_NegHeight/dNeg1);
```

```
thetaNeg1 = linspace(-postThetan1, -negThetan1, 50); %valid angles for (-1,-1)
NegOnePoints = -1*ones(1,50);
```

Objects at 2 and -2

```
d3 = 600; %cm
dNeg3 = 1000; %cm
%theta calculations
postTheta2 = atan((1.8-2)/d3);
negTheta2 = atan((-1.8-2)/d3);
theta2 = linspace(-negTheta2, postTheta2, 50); %valid angles for (1,1)
twoPoints = 2 * ones(1,50);
%negative two
postThetan2 = atan((1.8-2)/dNeg3);
negThetan2 = atan((-1.8-2)/dNeg3);
thetaNeg2 = linspace(-postThetan2, -negThetan2, 50); %valid angles for (-2,-2)
NegTwoPoints = -2*ones(1,50);

%Holy Matricies
%--
%(x, y) = (0,0) while theta changes -- simulates cone on rays

%Object 0's x and thetaX
point0X = [zeroPoints;
           theta0];
%Object 0's y and thetaY
point0Y = [zeroPoints;
           theta0];
%--

%--
%(x, y) = (1,1) while theta changes -- simulates cone on rays
%Object 1's x and thetaX
point1X = [onePoints;
           theta1];
%Object 1's y and thetaY
point1Y = [onePoints;
           theta1];
%--

%Object 2
point2X = [twoPoints;
           theta2];

point2Y = [twoPoints;
           theta2];

%--
%(x, y) = (-1,-1) while theta changes -- simulates cone on rays
%Object -1's x and thetaX
pointNeg1X = [NegOnePoints;
              thetaNeg1];
```

Multiple rays per point %% clear;
close all; clc;

```
%Object -1's y and thetaY
pointNeg1Y = [NegOnePoints;
              thetaNeg1];
%--

%Object -2
%--
pointNeg2X = [NegTwoPoints;
              thetaNeg2];

pointNeg2Y = [NegTwoPoints;
              thetaNeg2];

%Positions matrix
point0 = [point0X; point0Y]; %all rays for point (0,0)
point1 = [point1X; point1Y]; %all rays for point (1,1)
point2 = [point2X; point2Y]; %all rays (2,2)
pointNeg1 = [pointNeg1X; pointNeg1Y]; %all rays for point (-1,-1)
pointNeg2 = [pointNeg2X; pointNeg2Y]; %all ray for (-2,-2)
Point1 = [point1, pointNeg1];
Point2 = [point2, pointNeg2];
%PositionsMatrix = [point2, point1, point0, pointNeg1, pointNeg2]; %All rays
for all points

%Propagation Matrix
M0d1 = [1, d0; 0, 1]; %obj 0 -> lens
M1d1 = [1, d1; 0, 1]; %obj 1's-> lens
M2d1 = [1, d3; 0, 1]; %obj 2's-> lens
M1dNeg1 = [1, dNeg1; 0, 1]; %obj 1's-> lens
M2dNeg1 = [1, dNeg3; 0, 1]; %obj 2's-> lens
M_Lens = [1, 0; -1/F1, 1]; %lens shift
Md2 = [1, d2; 0, 1]; %lens -> sensor
M0 = Md2 * M_Lens * M0d1; %d0 propogation matrix
M1 = Md2 * M_Lens * M1d1; %d1 propogation matrix
M2 = Md2 * M_Lens * M2d1; %d2 propogation matrix
M0d1_padded = blkdiag(M0d1, M0d1); % object 0 -> lens
M1d1_padded = blkdiag(M1d1, M1d1); % object 1 -> lens
M2d1_padded = blkdiag(M2d1, M2d1); % object 2 -> lens
M1dNeg1_padded = blkdiag(M1dNeg1, M1dNeg1); % object 1 -> lens
M2dNeg1_padded = blkdiag(M2dNeg1, M2dNeg1); % object 2 -> lens
M_Lens_padded = blkdiag(M_Lens, M_Lens); % lens matrix
M0_padded = blkdiag(M0, M0); % final propagation object 0
M1_padded = blkdiag(M1, M1); % final propagation object 1
M2_padded = blkdiag(M2, M2); % final propagation object 2

%interinent positions (for plotting)
Ray0_b4 = M0d1_padded * point0;
Ray1_b4 = M1d1_padded * Point1;
Ray2_b4 = M2d1_padded * Point2;
Ray0_lens = M_Lens_padded * point0;
Ray1_lens = M_Lens_padded * Point1;
Ray2_lens = M_Lens_padded * Point2;
```

Multiple rays per point %% clear;
close all; clc;

%Ray's final postion

```
RayNeg2_Final = M2dNeg1_padded * pointNeg2;  
RayNeg1_Final = M1dNeg1_padded * pointNeg1;  
Ray0_Final = M0_padded * point0; %Where each ray hits sensor  
Ray1_Final = M1_padded * point1; %Where each ray hits sensor  
Ray2_Final = M2_padded * point2; %Where each ray hits sensor
```

Move lens

%Shift entire lens up by 0.6 three times

Object 0

%1.8/-1.8 -> 2.4/-1.2 -> 3.0/-0.6 -> 3.6/0

%2.4/-1.2

```
d0 = 800; %cm  
lensPos_height = 1.8; %3.6/2 cm (half above orgin half below)  
lensNeg_height = -1.8;  
PosHalfTheta0 = atan((lensPos_height+0.6)/d0);  
NegHalfTheta0 = atan((lensNeg_height +0.6)/d0);  
Shifted1theta0 = linspace(-1*NegHalfTheta0, PosHalfTheta0, 50);
```

%3.0/-0.6

```
Shift2PosHalfTheta0 = atan((lensPos_height+1.2)/d0);  
Shift2NegHalfTheta0 = atan((lensNeg_height+1.2)/d0);  
Shifted2theta0 = linspace(-Shift2NegHalfTheta0, Shift2PosHalfTheta0, 50);
```

%3.6/0

```
Shift3PosHalfTheta0 = atan((lensPos_height+1.8)/d0);  
Shift3NegHalfTheta0 = atan((lensNeg_height+1.8)/d0);  
Shifted3theta0 = linspace(-Shift3NegHalfTheta0, Shift3PosHalfTheta0, 50);
```

Object 1 and -1

%0.8/-2.8 -> 1.4/-2.2 -> 1.8/-1.8 -> 2.4/-1.2

%1.4/-2.2

```
d1 = 700;  
dNeg1 = 900; %cm  
lens_PosHeight1 = 0.8;  
lens_NegHeight1 = 2.8;  
postTheta1Shifted = atan((lens_PosHeight1 + 0.6) / d1);  
negTheta1Shifted = atan((lens_NegHeight1 + 0.6) / d1);  
postTheta1nShifted = atan((lens_PosHeight1 + 0.6) / dNeg1);  
negTheta1nShifted = atan((lens_NegHeight1 + 0.6) / dNeg1);  
theta1Shifted1 = linspace(-negTheta1Shifted, postTheta1Shifted, 50);  
thetaNeg1Shifted1 = linspace(-postTheta1nShifted, -negTheta1nShifted, 50);
```

%1.8/-1.8

```
postTheta2Shifted = atan((lens_PosHeight1 + 1.2) / d1);  
negTheta2Shifted = atan((lens_NegHeight1 + 1.2) / d1);
```

Multiple rays per point %% clear;
close all; clc;

```
postTheta2nShifted = atan((lens_PosHeight1 + 1.2) / dNeg1);  
negTheta2nShifted = atan((lens_NegHeight1 + 1.2) / dNeg1);  
theta1Shifted2 = linspace(-negTheta2Shifted, postTheta2Shifted, 50);  
thetaNeg1Shifted2 = linspace(-postTheta2nShifted, -negTheta2nShifted, 50);  
  
%2.4/-1.2  
postTheta3Shifted = atan((lens_PosHeight1 + 1.8) / d1);  
negTheta3Shifted = atan((lens_NegHeight1 + 1.8) / d1);  
postTheta3nShifted = atan((lens_PosHeight1 + 1.8) / dNeg1);  
negTheta3nShifted = atan((lens_NegHeight1 + 1.8) / dNeg1);  
theta1Shifted3 = linspace(-negTheta3Shifted, postTheta3Shifted, 50);  
thetaNeg1Shifted3 = linspace(-postTheta3nShifted, -negTheta3nShifted, 50);
```

Object 2 and -2

```
d3 = 600;  
dNeg3 = 1000; %cm  
%Up by 0.6  
postTheta2Shifted1 = atan(((1.8-2)+0.6)/d3);  
negTheta2Shifted1 = atan(((1.8-2)+0.6)/d3);  
postTheta2nShifted1 = atan(((1.8-2)+0.6)/dNeg3);  
negTheta2nShifted1 = atan(((1.8-2)+0.6)/dNeg3);  
theta2Shifted1 = linspace(-negTheta2Shifted1, postTheta2Shifted1, 50);  
thetaNeg2Shifted1 = linspace(-postTheta2nShifted1, -negTheta2nShifted1, 50);  
  
%Up by 1.2  
postTheta2Shifted2 = atan(((1.8-2)+1.2)/d3);  
negTheta2Shifted2 = atan(((1.8-2)+1.2)/d3);  
postTheta2nShifted2 = atan(((1.8-2)+1.2)/dNeg3);  
negTheta2nShifted2 = atan(((1.8-2)+1.2)/dNeg3);  
theta2Shifted2 = linspace(-negTheta2Shifted2, postTheta2Shifted2, 50);  
thetaNeg2Shifted2 = linspace(-postTheta2nShifted2, -negTheta2nShifted2, 50);  
  
%Up by 1.8  
postTheta2Shifted3 = atan(((1.8-2)+1.8)/d3);  
negTheta2Shifted3 = atan(((1.8-2)+1.8)/d3);  
postTheta2nShifted3 = atan(((1.8-2)+1.8)/dNeg3);  
negTheta2nShifted3 = atan(((1.8-2)+1.8)/dNeg3);  
theta2Shifted3 = linspace(-negTheta2Shifted3, postTheta2Shifted3, 50);  
thetaNeg2Shifted3 = linspace(-postTheta2nShifted3, -negTheta2nShifted3, 50);
```

Propogation of shifted lens

Up by 0.6

```
%Object 0's x and thetaX  
point0Xshift1 = [zeroPoints;  
                 Shifted1theta0];  
%Object 0's y and thetaY  
point0Yshift1 = [zeroPoints;  
                 Shifted1theta0];  
%--
```

Multiple rays per point %% clear;
close all; clc;

```
%--
%Object 1's x and thetaX
point1Xshift1 = [onePoints;
                 theta1Shifted1];
%Object 1's y and thetaY
point1Yshift1 = [onePoints;
                 theta1Shifted1];
%--
%Object 2
point2Xshift1 = [twoPoints;
                 theta2Shifted1];

point2Yshift1 = [twoPoints;
                 theta2Shifted1];

%Object -1's x and thetaX
pointNeg1Xshift1 = [NegOnePoints;
                   thetaNeg1Shifted1];

%Object -1's y and thetaY
pointNeg1Yshift1 = [NegOnePoints;
                   thetaNeg1Shifted1];
%--

%Object -2
%--
pointNeg2Xshift1 = [NegTwoPoints;
                   thetaNeg2Shifted1];

pointNeg2Yshift1 = [NegTwoPoints;
                   thetaNeg2Shifted1];

%Positions shifted by 0.6
point0shift1 = [point0Xshift1; point0Yshift1];
point1shift1 = [point1Xshift1; point1Yshift1];
point2shift1 = [point2Xshift1; point2Yshift1];
pointNeg1shift1 = [pointNeg1Xshift1; pointNeg1Yshift1];
pointNeg2shift1 = [pointNeg2Xshift1; pointNeg2Yshift1];
Point1shift1 = [point1shift1, pointNeg1shift1];
Point2shift1 = [point2shift1, pointNeg2shift1];

%Ray's final postion
RayNeg2_Finalshift1 = M2dNeg1_padded * pointNeg2shift1;
RayNeg1_Finalshift1 = M1dNeg1_padded * pointNeg1shift1;
Ray0_Finalshift1 = M0_padded * point0shift1; %Where each ray hits sensor
Ray1_Finalshift1 = M1_padded * point1shift1; %Where each ray hits sensor
Ray2_Finalshift1 = M2_padded * point2shift1; %Where each ray hits sensor
```

Up by 1.2

```
%Object 0's x and thetaX
point0Xshift2 = [zeroPoints;
```

Multiple rays per point %% clear;
close all; clc;

```
        Shifted2theta0];
%Object 0's y and thetaY
point0Yshift2 = [zeroPoints;
        Shifted2theta0];
%--

%--
%Object 1's x and thetaX
point1Xshift2 = [onePoints;
        theta1Shifted2];
%Object 1's y and thetaY
point1Yshift2 = [onePoints;
        theta1Shifted2];
%--
%Object 2
point2Xshift2 = [twoPoints;
        theta2Shifted2];

point2Yshift2 = [twoPoints;
        theta2Shifted2];

%--
%Object -1's x and thetaX
pointNeg1Xshift2 = [NegOnePoints;
        thetaNeg1Shifted2];

%Object -1's y and thetaY
pointNeg1Yshift2 = [NegOnePoints;
        thetaNeg1Shifted2];
%--

%Object -2
%--
pointNeg2Xshift2 = [NegTwoPoints;
        thetaNeg2Shifted2];

pointNeg2Yshift2 = [NegTwoPoints;
        thetaNeg2Shifted2];

%Positions shifted by 1.2
point0shift2 = [point0Xshift2; point0Yshift2];
point1shift2 = [point1Xshift2; point1Yshift2];
point2shift2 = [point2Xshift2; point2Yshift2];
pointNeg1shift2 = [pointNeg1Xshift2; pointNeg1Yshift2];
pointNeg2shift2 = [pointNeg2Xshift2; pointNeg2Yshift2];
Point1shift2 = [point1shift2, pointNeg1shift2];
Point2shift2 = [point2shift2, pointNeg2shift2];

RayNeg2_Finalshift2 = M2dNeg1_padded * pointNeg2shift2;
RayNeg1_Finalshift2 = M1dNeg1_padded * pointNeg1shift2;
Ray0_Finalshift2 = M0_padded * point0shift2; %Where each ray hits sensor
Ray1_Finalshift2 = M1_padded * point1shift2; %Where each ray hits sensor
Ray2_Finalshift2 = M2_padded * point2shift2; %Where each ray hits sensor
```

Up by 1.8

```
%Object 0's x and thetaX
point0Xshift3 = [zeroPoints;
                 Shifted3theta0];
%Object 0's y and thetaY
point0Yshift3 = [zeroPoints;
                 Shifted3theta0];
%--

%--
%Object 1's x and thetaX
point1Xshift3 = [onePoints;
                 theta1Shifted3];
%Object 1's y and thetaY
point1Yshift3 = [onePoints;
                 theta1Shifted3];
%--

%Object 2
point2Xshift3 = [twoPoints;
                 theta2Shifted3];

point2Yshift3 = [twoPoints;
                 theta2Shifted3];

%--

%Object -1's x and thetaX
pointNeg1Xshift3 = [NegOnePoints;
                   thetaNeg1Shifted3];

%Object -1's y and thetaY
pointNeg1Yshift3 = [NegOnePoints;
                   thetaNeg1Shifted3];
%--

%Object -2
%--
pointNeg2Xshift3 = [NegTwoPoints;
                   thetaNeg2Shifted3];

pointNeg2Yshift3 = [NegTwoPoints;
                   thetaNeg2Shifted3];

%Positions shifted by 1.2
point0shift3 = [point0Xshift3; point0Yshift3];
point1shift3 = [point1Xshift3; point1Yshift3];
point2shift3 = [point2Xshift3; point2Yshift3];
pointNeg1shift3 = [pointNeg1Xshift3; pointNeg1Yshift3];
pointNeg2shift3 = [pointNeg2Xshift3; pointNeg2Yshift3];
Point1shift3 = [point1shift3, pointNeg1shift3];
Point2shift3 = [point2shift3, pointNeg2shift3];
```

Multiple rays per point %% clear;
close all; clc;

```
RayNeg2_Finalshift3 = M2dNeg1_padded * pointNeg2shift3;  
RayNeg1_Finalshift3 = M1dNeg1_padded * pointNeg1shift3;  
Ray0_Finalshift3 = M0_padded * point0shift3; %Where each ray hits sensor  
Ray1_Finalshift3 = M1_padded * point1shift3; %Where each ray hits sensor  
Ray2_Finalshift3 = M2_padded * point2shift3; %Where each ray hits sensor
```

Plotting blurred images for each instance

```
%Sensor specifications  
pixel_size = 3.45e-6;  
sensor_H_px = 1080;  
sensor_W_px = 1440;  
sensor_H = sensor_H_px * pixel_size; %sensor height  
sensor_W = sensor_W_px * pixel_size; %sensor width  
d2 = f1;  
  
all_objects_rays = { ...  
    {RayNeg2_Final, RayNeg2_Finalshift1, RayNeg2_Finalshift2,  
RayNeg2_Finalshift3}, ... % -2  
    {RayNeg1_Final, RayNeg1_Finalshift1, RayNeg1_Finalshift2,  
RayNeg1_Finalshift3}, ... % -1  
    {Ray0_Final, Ray0_Finalshift1, Ray0_Finalshift2,  
Ray0_Finalshift3}, ... % 0  
    {Ray1_Final, Ray1_Finalshift1, Ray1_Finalshift2,  
Ray1_Finalshift3}, ... % 1  
    {Ray2_Final, Ray2_Finalshift1, Ray2_Finalshift2,  
Ray2_Finalshift3} ... % 2  
};  
  
object_names = {'-2', '-1', '0', '1', '2'};  
shift_labels = {'Original', 'Up 0.6 cm', 'Up 1.2 cm', 'Up 1.8 cm'};  
colors = lines(5);  
  
%Loop over each object  
for obj_idx = 1:length(all_objects_rays)  
    %Create a new figure for this object  
    figure('Name', ['Object ', object_names{obj_idx}], 'NumberTitle', 'off');  
  
    %Loop over each shift  
    for shift_idx = 1:4  
        %Extract rays for current object and shift  
        rays = all_objects_rays{obj_idx}{shift_idx};  
        if isempty(rays)  
            continue  
        end  
  
        %Sensor positions  
        x_sensor = rays(1,:); % [cm] X positions at sensor plane  
        y_sensor = rays(3,:); % [cm] Y positions at sensor plane  
  
        %Compute blur (standard deviation of rays in X and Y)  
        sigma_x = std(x_sensor);
```

Multiple rays per point %% clear;
close all; clc;

```
sigma_y = std(y_sensor);

%Print standard deviation and number of rays
fprintf("Object %s | Shift %d |  $\sigma_x$ =%.6f |  $\sigma_y$ =%.6f | Rays=%d\n", ...
    object_names{obj_idx}, shift_idx, sigma_x, sigma_y,
    numel(x_sensor));

%Map sensor positions to pixel coordinates
%Convert cm to pixels using sensor size and pixel count
x_pix = round((x_sensor + sensor_W/2) / sensor_W * sensor_W_px);
y_pix = round((y_sensor + sensor_H/2) / sensor_H * sensor_H_px);

%Clamp to valid pixel range
x_pix = min(max(x_pix,1), sensor_W_px);
y_pix = min(max(y_pix,1), sensor_H_px);

%Count number of rays per pixel using accumarray
counts = accumarray([y_pix', x_pix'], 1, [sensor_H_px, sensor_W_px]);

%Map counts back to rays
linear_idx = sub2ind([sensor_H_px, sensor_W_px], y_pix, x_pix);
intensities = counts(linear_idx);

%Normalize marker size for scatter
intensities = intensities ./ max(intensities); % scale 0-1

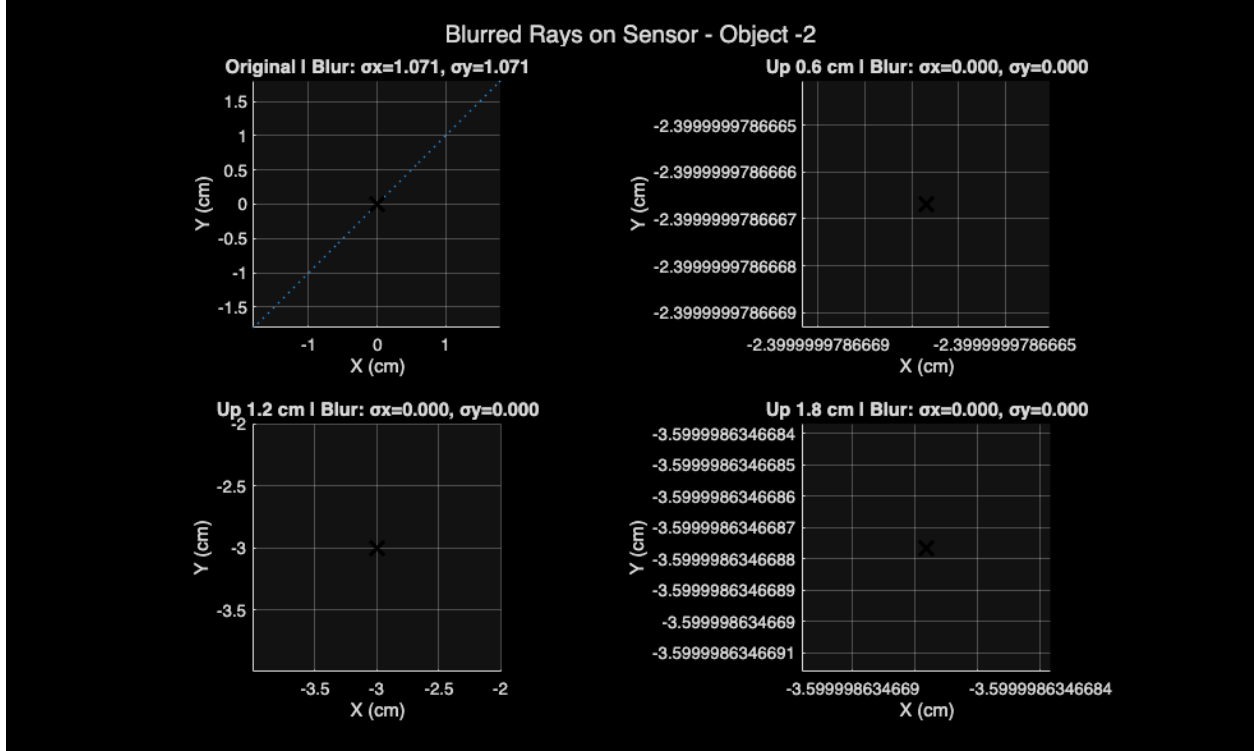
%Plot the blurred rays
subplot(2,2,shift_idx);
scatter(x_sensor, y_sensor, intensities, colors(obj_idx,:),
'filled'); hold on;
plot(mean(x_sensor), mean(y_sensor),
'kx','MarkerSize',12,'LineWidth',2);
axis equal tight;
xlabel('X (cm)'); ylabel('Y (cm)');
title([shift_labels{shift_idx}, sprintf(' | Blur:  $\sigma_x$ =%.3f,  $\sigma_y$ =%.3f',
sigma_x, sigma_y)]);
grid on;
end

%Title
sgtitle(['Blurred Rays on Sensor - Object ', object_names{obj_idx}]);
end
```

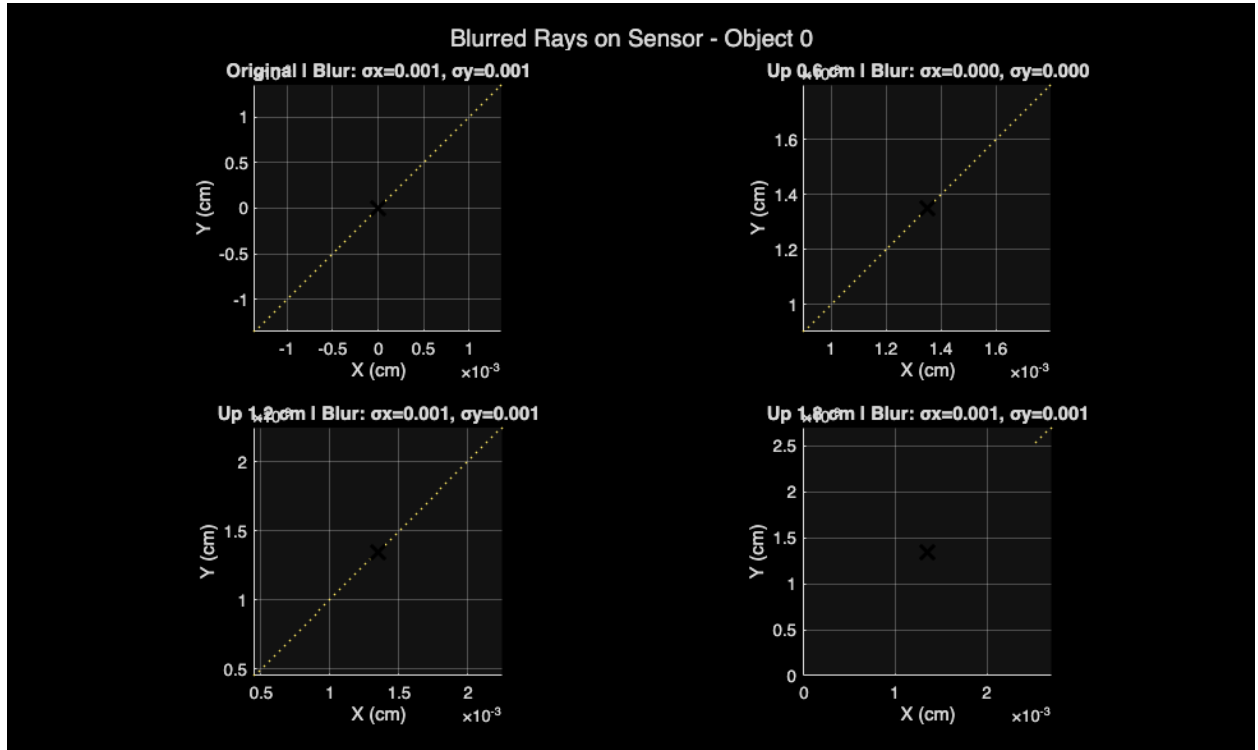
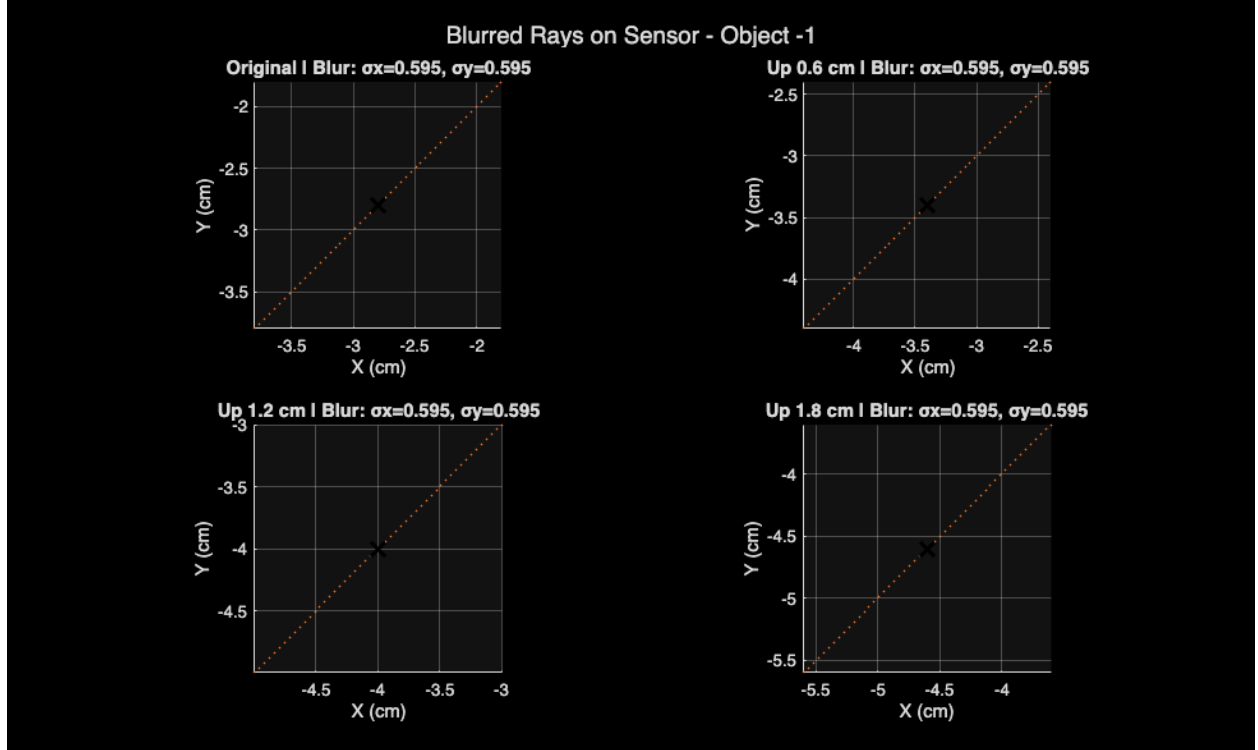
```
Object -2 | Shift 1 |  $\sigma_x$ =1.070986 |  $\sigma_y$ =1.070986 | Rays=50
Object -2 | Shift 2 |  $\sigma_x$ =0.000000 |  $\sigma_y$ =0.000000 | Rays=50
Object -2 | Shift 3 |  $\sigma_x$ =0.000000 |  $\sigma_y$ =0.000000 | Rays=50
Object -2 | Shift 4 |  $\sigma_x$ =0.000000 |  $\sigma_y$ =0.000000 | Rays=50
Object -1 | Shift 1 |  $\sigma_x$ =0.594992 |  $\sigma_y$ =0.594992 | Rays=50
Object -1 | Shift 2 |  $\sigma_x$ =0.594991 |  $\sigma_y$ =0.594991 | Rays=50
Object -1 | Shift 3 |  $\sigma_x$ =0.594988 |  $\sigma_y$ =0.594988 | Rays=50
Object -1 | Shift 4 |  $\sigma_x$ =0.594985 |  $\sigma_y$ =0.594985 | Rays=50
Object 0 | Shift 1 |  $\sigma_x$ =0.000803 |  $\sigma_y$ =0.000803 | Rays=50
Object 0 | Shift 2 |  $\sigma_x$ =0.000268 |  $\sigma_y$ =0.000268 | Rays=50
Object 0 | Shift 3 |  $\sigma_x$ =0.000535 |  $\sigma_y$ =0.000535 | Rays=50
```

Multiple rays per point %% clear;
close all; clc;

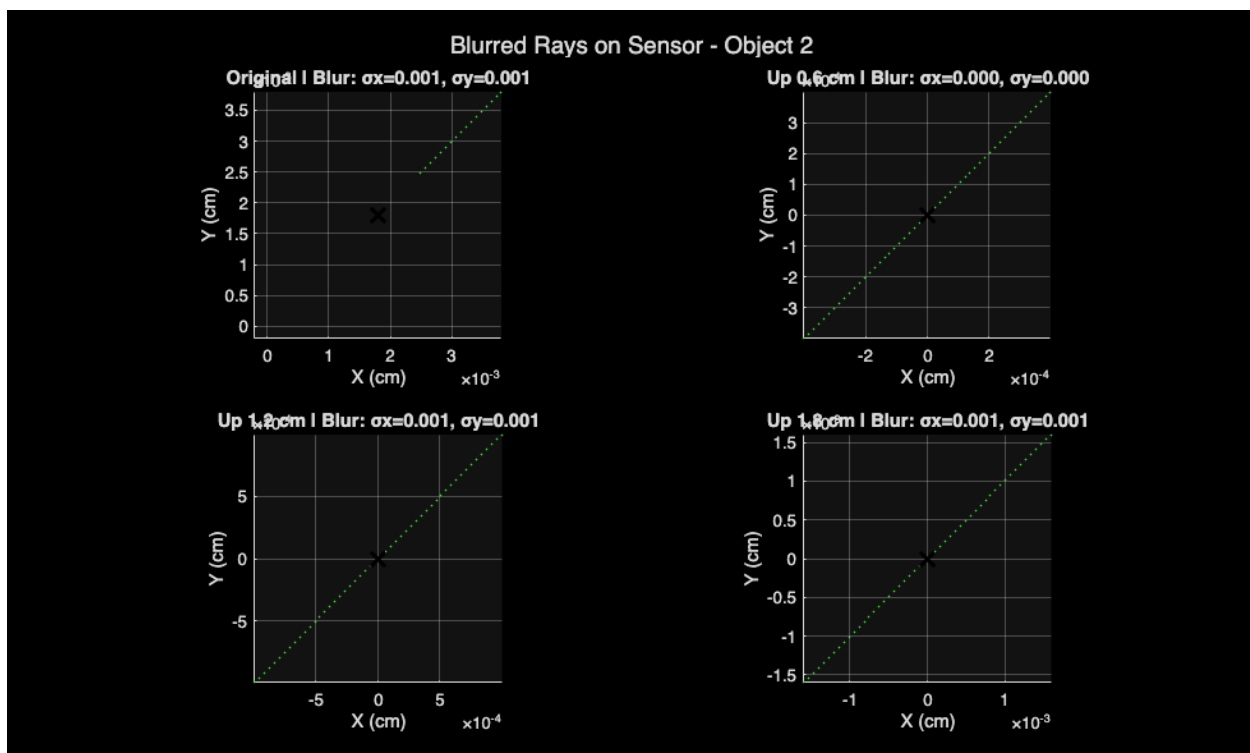
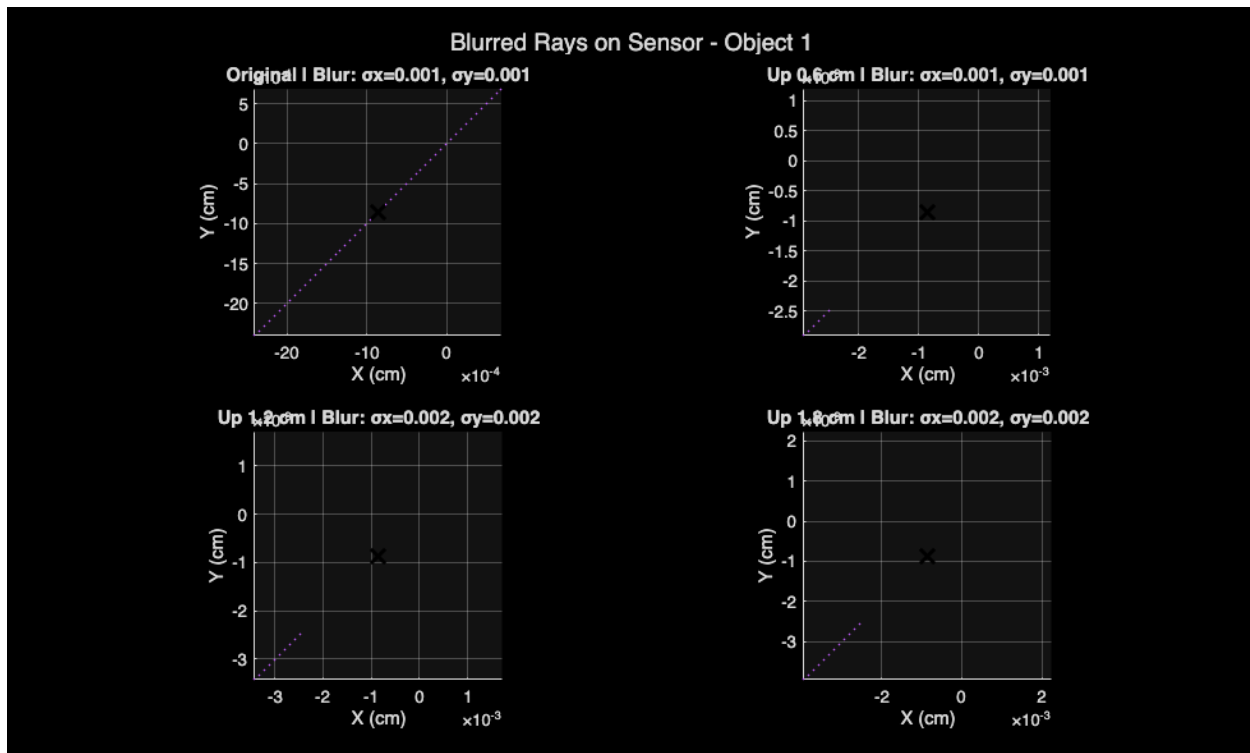
```
Object 0 / Shift 4 /  $\sigma_x=0.000803$  /  $\sigma_y=0.000803$  / Rays=50  
Object 1 / Shift 1 /  $\sigma_x=0.000918$  /  $\sigma_y=0.000918$  / Rays=50  
Object 1 / Shift 2 /  $\sigma_x=0.001224$  /  $\sigma_y=0.001224$  / Rays=50  
Object 1 / Shift 3 /  $\sigma_x=0.001530$  /  $\sigma_y=0.001530$  / Rays=50  
Object 1 / Shift 4 /  $\sigma_x=0.001836$  /  $\sigma_y=0.001836$  / Rays=50  
Object 2 / Shift 1 /  $\sigma_x=0.001190$  /  $\sigma_y=0.001190$  / Rays=50  
Object 2 / Shift 2 /  $\sigma_x=0.000238$  /  $\sigma_y=0.000238$  / Rays=50  
Object 2 / Shift 3 /  $\sigma_x=0.000595$  /  $\sigma_y=0.000595$  / Rays=50  
Object 2 / Shift 4 /  $\sigma_x=0.000952$  /  $\sigma_y=0.000952$  / Rays=50
```



Multiple rays per point %% clear;
close all; clc;



Multiple rays per point %% clear;
close all; clc;



Refocusing (back tracing)

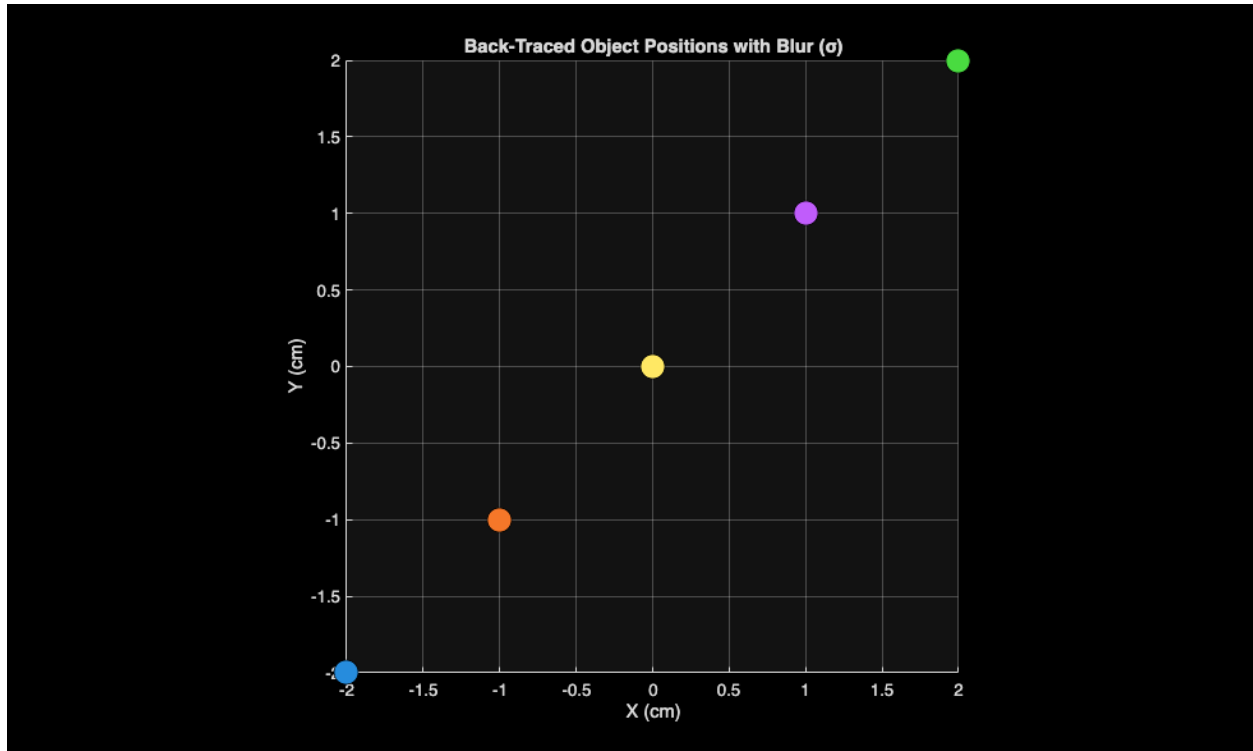
```
%Collect all final rays per object
all_objects = {
```

Multiple rays per point %% clear;
close all; clc;

```
{RayNeg2_Final, RayNeg2_Finalshift1, RayNeg2_Finalshift2,  
RayNeg2_Finalshift3}, ...  
{RayNeg1_Final, RayNeg1_Finalshift1, RayNeg1_Finalshift2,  
RayNeg1_Finalshift3}, ...  
{Ray0_Final, Ray0_Finalshift1, Ray0_Finalshift2,  
Ray0_Finalshift3}, ...  
{Ray1_Final, Ray1_Finalshift1, Ray1_Finalshift2,  
Ray1_Finalshift3}, ...  
{Ray2_Final, Ray2_Finalshift1, Ray2_Finalshift2,  
Ray2_Finalshift3}  
};  
  
object_names = {'-2', '-1', '0', '1', '2'};  
colors = lines(5);  
  
%Create figure  
figure('Name', 'Refocused Points', 'NumberTitle', 'off'); hold on;  
  
%Loop over each object  
for obj_idx = 1:length(all_objects)  
  
    all_ref_x = []; %store all X positions  
    all_ref_y = []; %store all Y positions  
  
    %Loop over shifts for this object  
    for shift_idx = 1:4  
        rays_final = all_objects{obj_idx}{shift_idx};  
        if isempty(rays_final)  
            continue  
        end  
  
        %Back ray trace to object plane  
        x_ref = rays_final(1,:) + rays_final(2,:) * d2; % X positions [cm]  
        y_ref = rays_final(3,:) + rays_final(4,:) * d2; % Y positions [cm]  
  
        %Append to all_ref arrays  
        all_ref_x = [all_ref_x, x_ref];  
        all_ref_y = [all_ref_y, y_ref];  
    end  
  
    %Object position  
    obj_pos_x = str2double(object_names{obj_idx}); %object X position  
    (-2,-1,...,2)  
    obj_pos_y = str2double(object_names{obj_idx}); %object Y position  
    (-2,-1,...,2)  
  
    intensity = length(all_ref_x); %number of rays  
  
    %Plot the object positions  
    scatter(obj_pos_x, obj_pos_y, intensity, colors(obj_idx,:), 'filled',  
    'MarkerEdgeColor', 'k'); hold on;  
end  
  
xlabel('X (cm)');
```

Multiple rays per point %% clear;
close all; clc;

```
ylabel('Y (cm)');  
title('Back-Traced Object Positions with Blur ( $\sigma$ )');  
axis equal tight;  
grid on;  
hold off;
```



Published with MATLAB® R2025a