Yuval Levental
IMGS-633

Near Field Propagation Exam Project

In this project, I am given a hyper-Gaussian field, which is concentrated at a certain waist size. As the field propagates above the surface, the amplitude of the electric field generally dissipates. This simulation displays the amplitudes and phases of the field at certain *z* distances.

The initial electric field is equal to the following:

$$E(x, y, 0) = \exp(-(r^2/w_0^2)^\eta) \text{ where } r^2 = x^2 + y^2 \text{ and } \eta > 30.$$

In this equation, $w_0 = 100\mu m$, and I set the value $\eta = 50$. This field has a hyper-Gaussian profile, which means that on the surface, the amplitude of the field is concentrated in a certain diameter, but rapidly drops off to zero after the diameter is exceeded.

I created a mesh grid from 256 x-points and 256 y-points, as to perform the Fast Fourier transform, 2^n grid points are needed. One important principle is to make the waist size pixels small enough to reduce artifacts at the edges of the array due to the periodic extension properties of the FFT. The equation for the waist size is $w_0 = \sqrt{N}/\pi$, where N is 256 pixels. W_0 was calculated to be around 9.03 pixels.

To calculate the amplitudes and phases at a distance above the surface, several steps must be followed. The Fourier transform of the initial electric field is calculated:

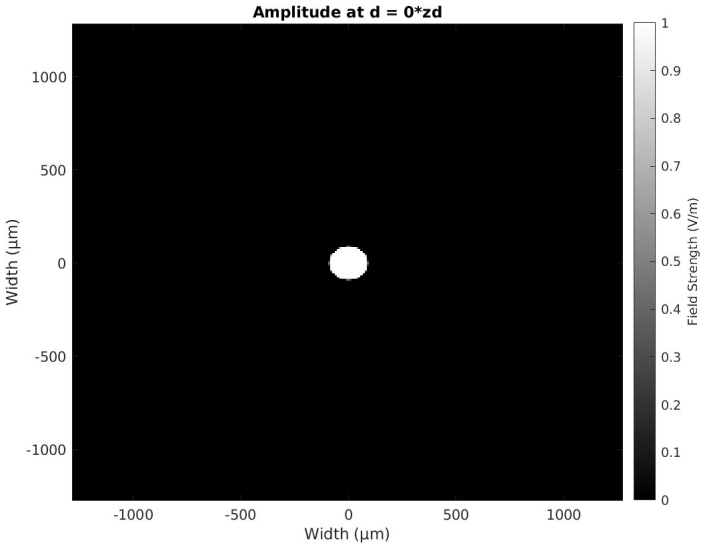
$$G(k_x, k_y) = \frac{1}{2\pi} \iint E(x, y, 0) e^{-ik_x x} e^{-ik_y y} dxdy = FT\left[E(x, y, 0)\right]$$

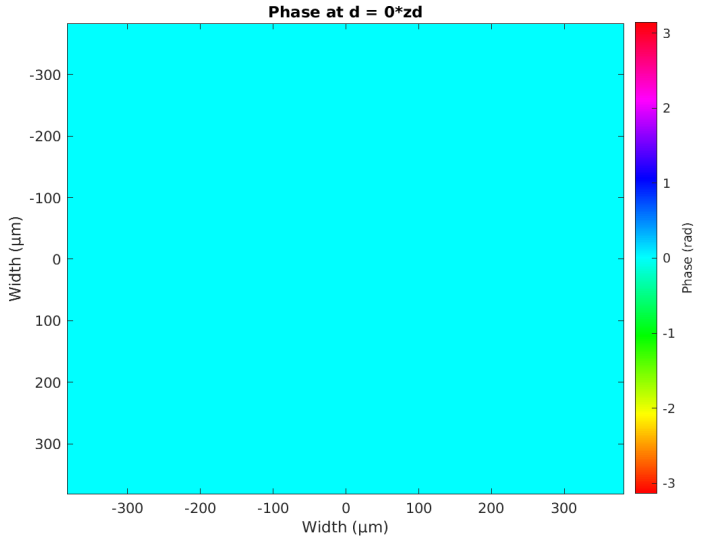
The electric field is assumed to be composed of Huygen wavelets, so $G(k_x,k_y)$ is multiplied by the Fourier transform of the Huygen distance equation. Finally, the inverse Fourier transform is taken to find the real and imaginary components of the field at the chosen z-distance:

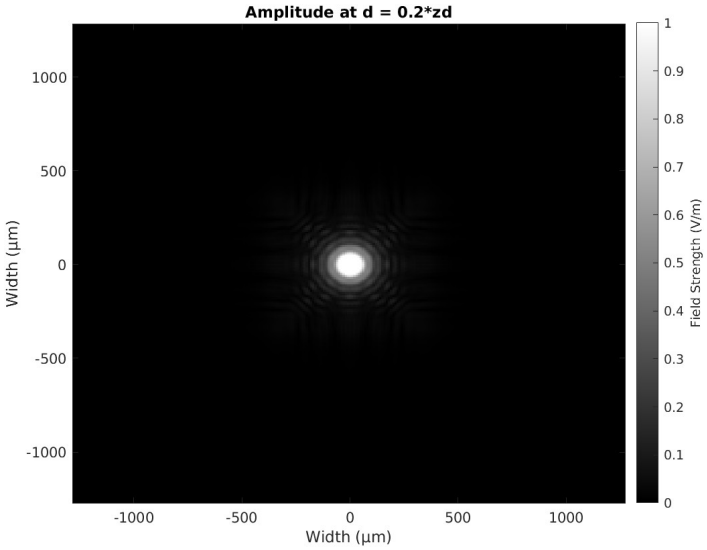
$$E(x', y', z) = FT^{-1}\left\{G(k_x, k_y) \exp\left[iz(k_x^2 + k_y^2)/2k\right]\right\}$$

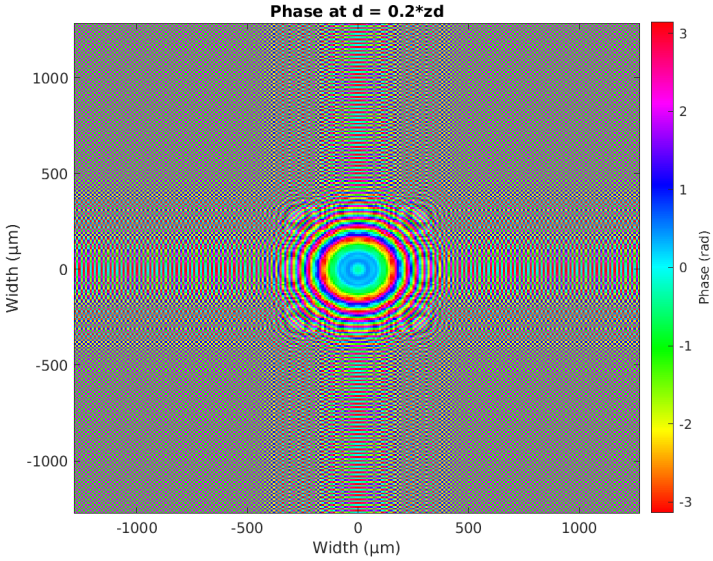
In this case I decided to set $k = 1/(2\lambda)$, where $\lambda = 1\ \mu m$. The distances are equal to:

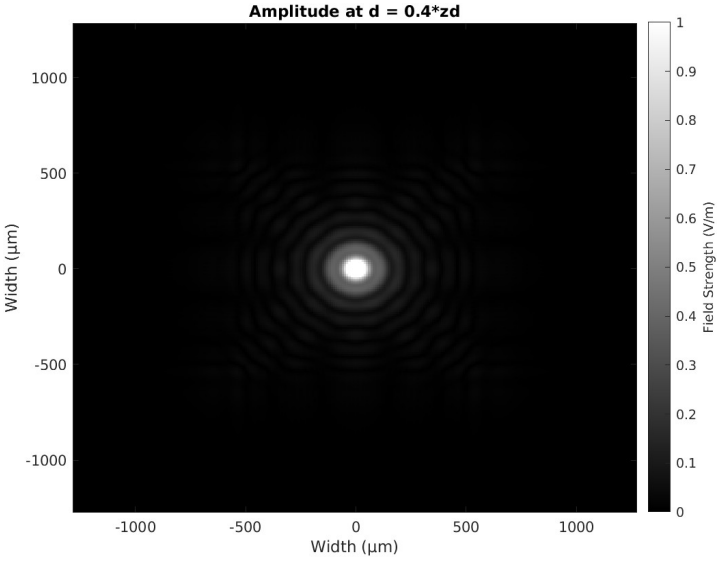
$$z = 0.2z_d,\ z = 0.4z_d,\ z = 0.6z_d,\ z = 0.8z_d,\ z = 1.0z_d \text{ where } z_d = \pi w_0^2/\lambda.$$

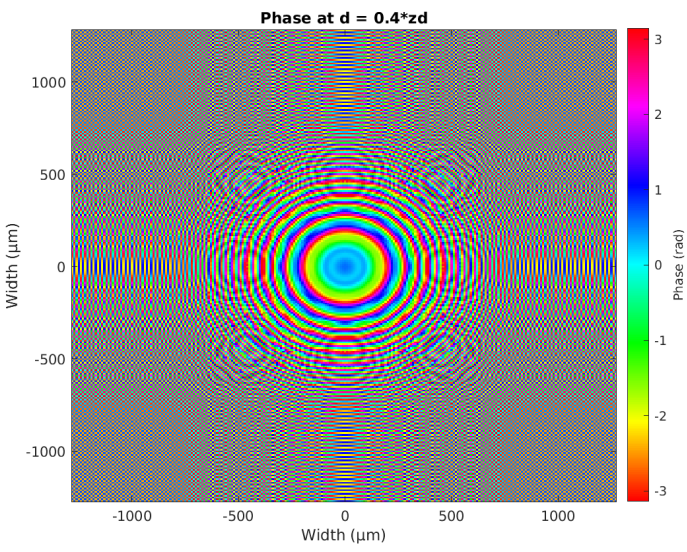

Amplitude at d = 0*zd
Width (µm)
Width (µm)
Field Strength (V/m)

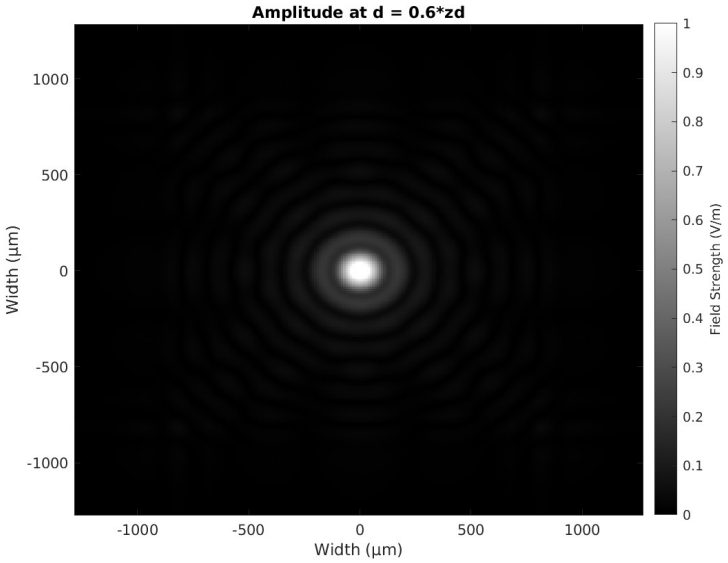

Phase at d = 0*zd
Width (µm)
Width (µm)
Phase (rad)

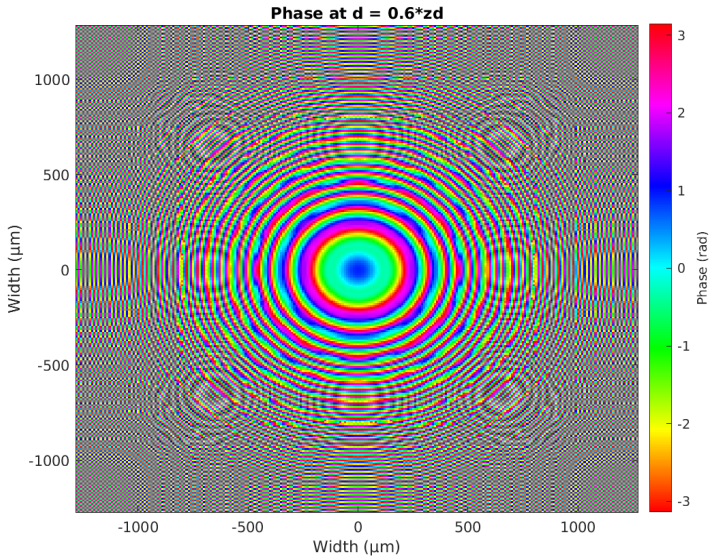

Amplitude at d = 0.2*zd
Width (µm)
Width (µm)
Field Strength (V/m)

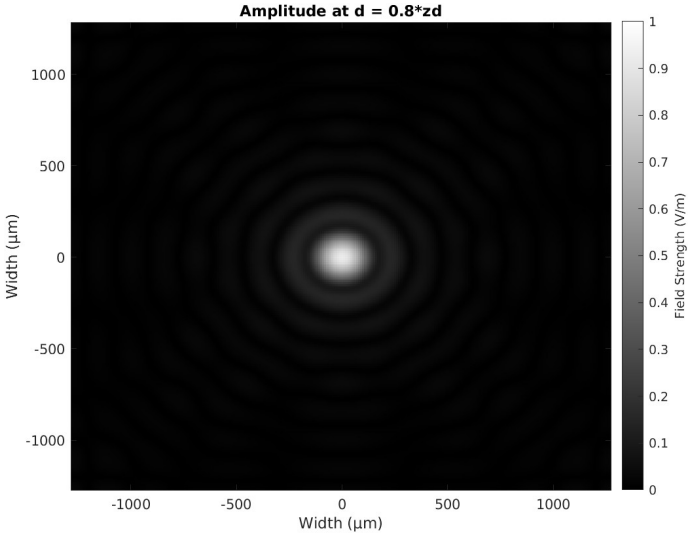

Phase at d = 0.2*zd
Width (µm)
Width (µm)
Phase (rad)

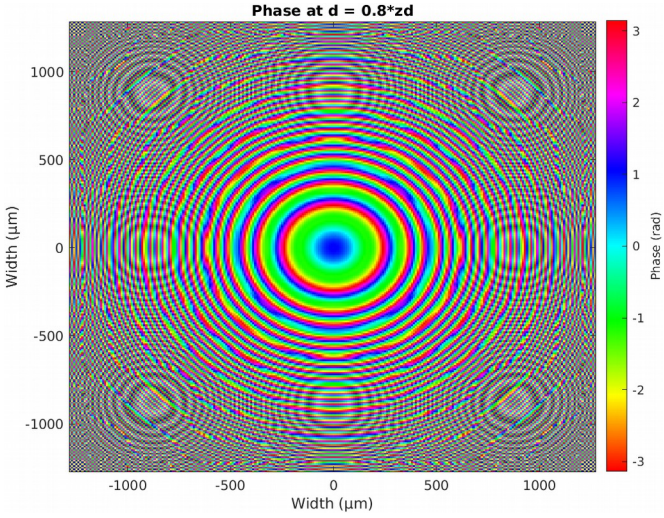

Amplitude at d = 0.4*zd
Width (µm)
Width (µm)
Field Strength (V/m)

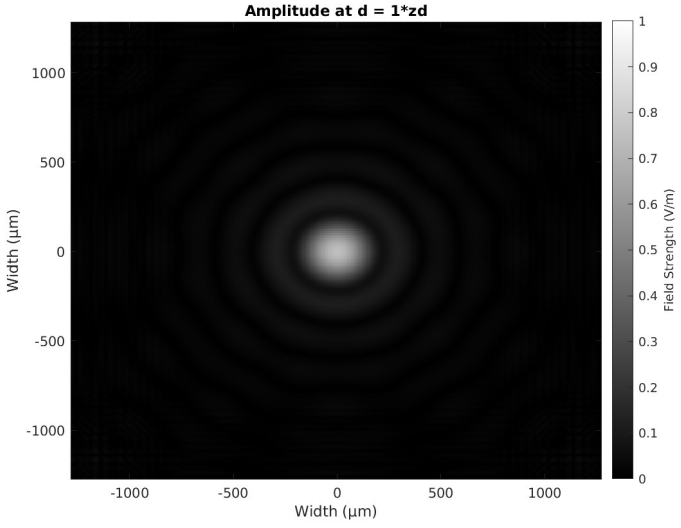

Phase at d = 0.4*zd
Width (µm)
Width (µm)
Phase (rad)

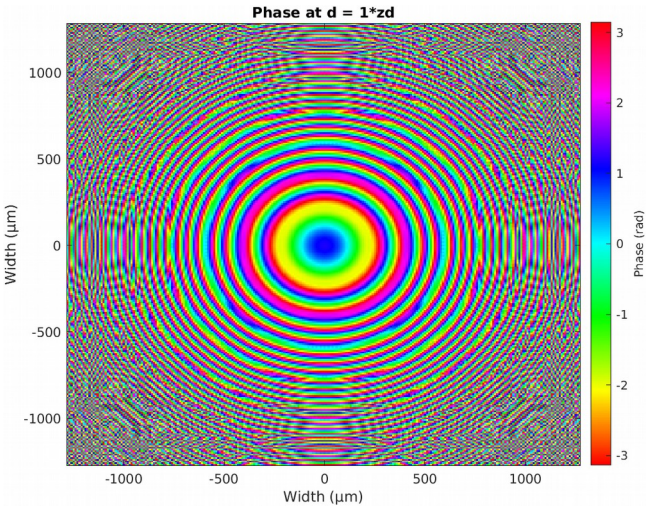

Amplitude at d = 0.6*zd
Width (µm)
Width (µm)
Field Strength (V/m)


Phase at d = 0.6*zd
Width (µm)
Width (µm)
Phase (rad)


Amplitude at d = 0.8*zd
Width (µm)
Width (µm)
Field Strength (V/m)


Phase at d = 0.8*zd
Width (µm)
Width (µm)
Phase (rad)


Amplitude at d = 1*zd
Width (µm)
Width (µm)
Field Strength (V/m)


Phase at d = 1*zd
Width (µm)
Width (µm)
Phase (rad)

Methods

This code was programmed in MATLAB.

When I put in the mesh grid values into E(x,y,0), I realized that I needed to perform dot matrix multiplication for each power value. After inputting the values into E(x,y,0), I took the fftshift of the results so that the zero-frequency component could be where the spatial frequencies are equal to zero. I then took the 2-D Fast Fourier Transform of the data.

Because the image was transformed from the space domain into the spatial frequency domain, I had to create a mesh grid for the Fourier transform of the Huygen distance equation. The original values were -128:1:128-1, but the new values were now -1:1/(128):1-(1/(128)). This is because spatial frequency is the inverse of space, so the limits and spacing were rearranged. This spacing made sure that the grid size was exactly the same as the space domain grid size, so the Fourier transform was dot multiplied by the fftshift of the Huygen distance equation.

After this step, the inverse fourier transformation was taken, along with the ifftshift, negating the Fourier shift of the previous calculations. The final results were displayed with a colorbar.

One extremely helpful resource was the textbook Computational Fourier Optics: A MATLAB Tutorial by David G. Voelz. Chapter 5, *Propagation Simulation*, explained MATLAB FFT programming very well. This resource was suggested to me by a fellow student.

Results and Conclusion

As the distance from the beam increases, the electric field amplitude at the center gradually decreases and spreads outward. This is because all light is subject to diffraction. Since light is a periodic wave, there are alternating light and dark bands at each distance.

The frequency of light propagation for a hyper-gaussian beam at $\eta = 50$, is much faster than for a regular gaussian beam at $\eta = 1$. This is because there is significantly more power at the center of the distribution, which drops off much faster.

The alternating bands are also represented by the phase diagrams, which indicate that the phases change constantly as the light waves propagate. At the center of the graphs, the phase is initially equal to zero, but later is equal to $\pi/2$ as the phase gradually changes.

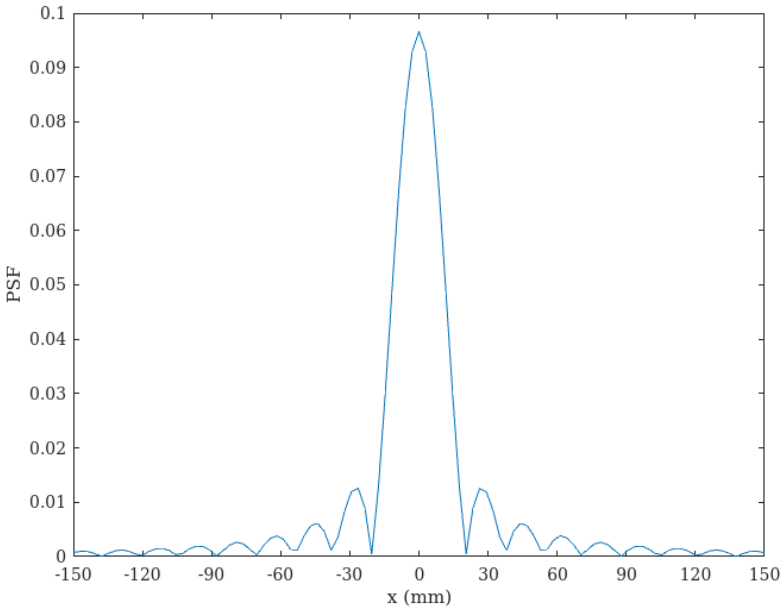
MATLAB Code

```
x = -128:1:128-1; %initial mesh grid for the space domain
y = -128:1:128-1;

[X,Y] = meshgrid(x,y);

E_1 = fft2(fftshift(exp(-((X.^2 + Y.^2)/((9.03)^2)).^50))); %The initial electric
field

x2 = -1:1/(128):1-(1/(128)); %initial mesh grid for the space domain
y2 = -1:1/(128):1-(1/(128));

[X2,Y2] = meshgrid(x2,y2);

E_2 = E_1.*fftshift(exp(1i*1*pi*(((9.03)^2)*(X2.^2 + Y2.^2)))); %The fourier
transform of the electric field and propagation multiplied
E_3 = ifftshift(ifft2(E_2)); %The final inverse fourier transform

imagesc(x,y,angle(E_3))
colormap hsv
c = colorbar;
title('Phase at d = 1*zd')
xlabel('Width (μm)')
ylabel('Width (μm)')
ylabel(c, 'Phase (rad)')
xticklabels({'-1000','-500','0','500','1000'})
yticklabels({'1000','500','0','-500','-1000'})
caxis([-pi,pi])
```

Yuval Levental

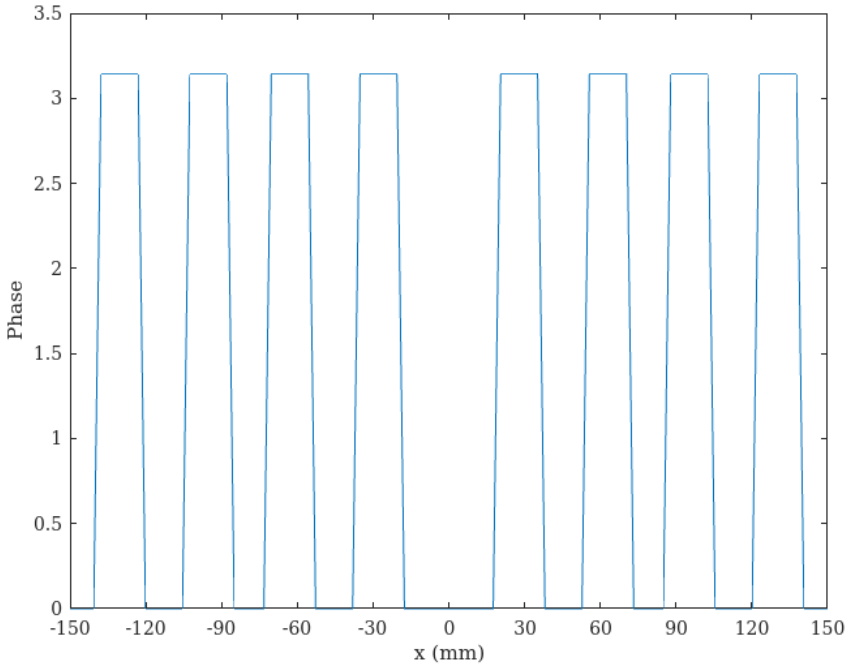
Part 2: Image Fields

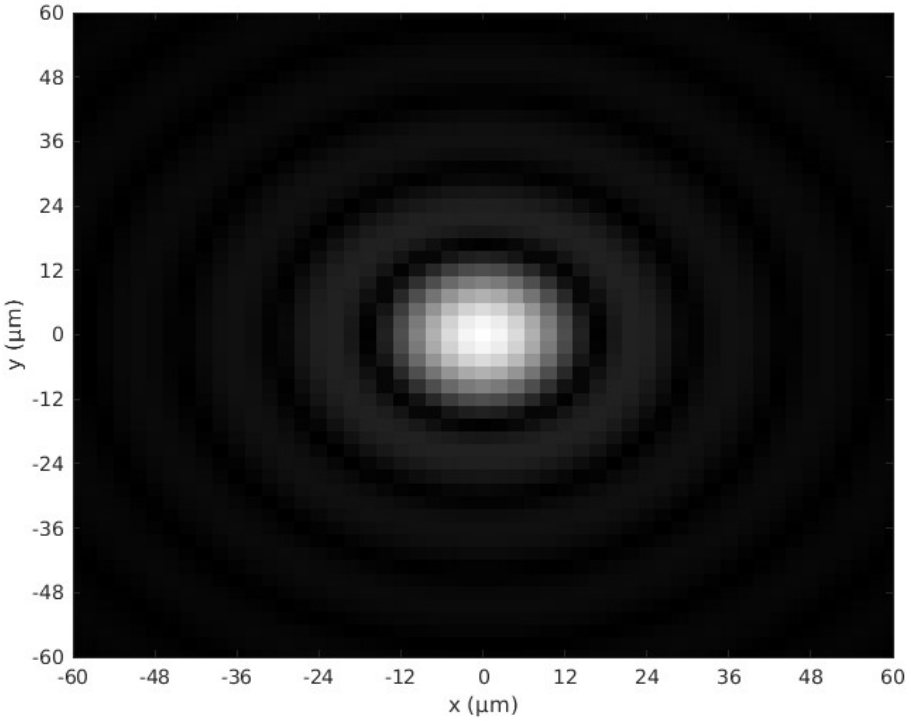
1.

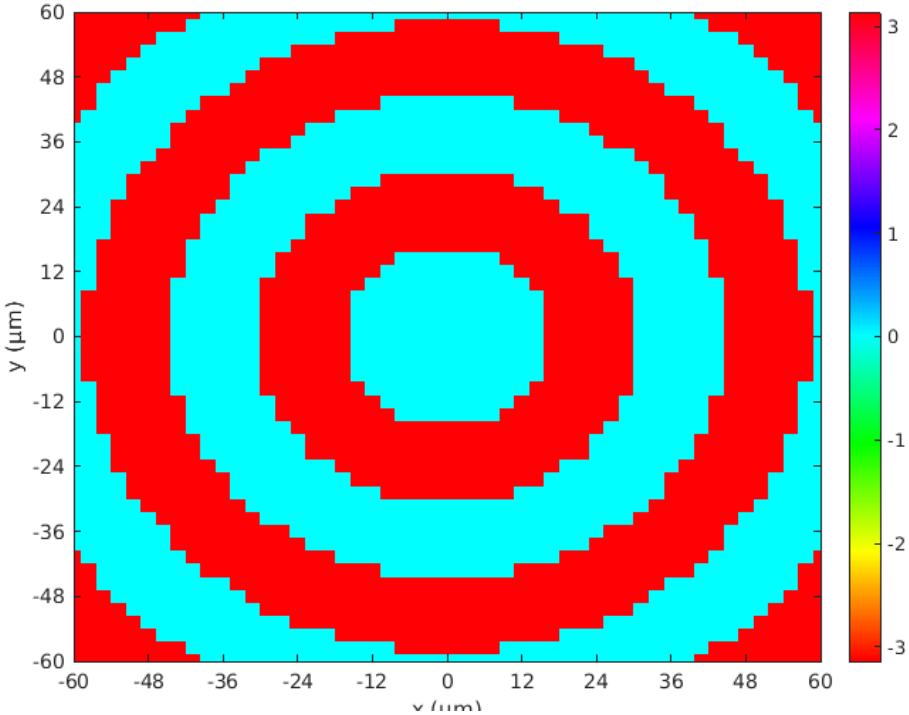


This was computed through a Fourier Transform. di = f#*2*D = 500 mm

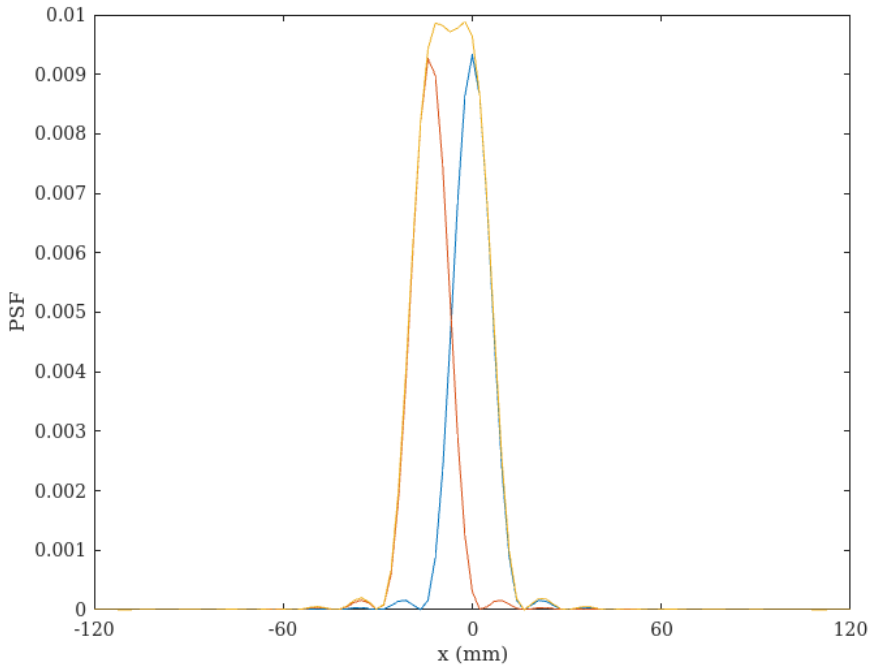
$$h(x_i, y_i) = \frac{M}{\lambda^2 d_i^2} \iint A_l(x_l, y_l) \exp\left[-ik(x_l/d_i)(x_i)\right] \exp\left[-ik(y_l/d_i)(y_i)\right] dx_l dy_l$$



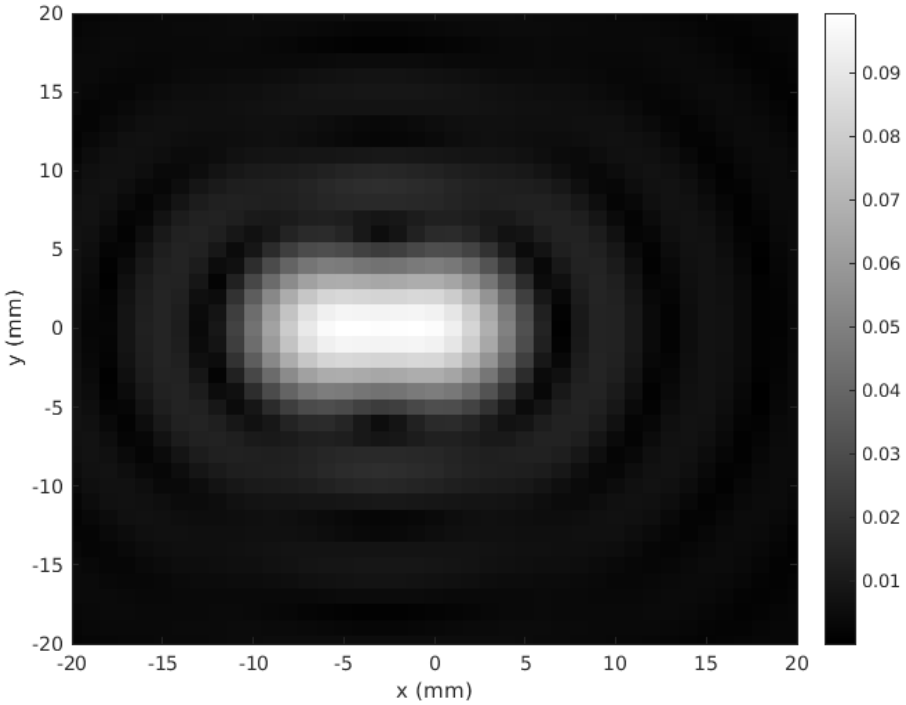

60
48
36
24
12
0
-12
-24
-36
-48
-60
y (µm)
-60 -48 -36 -24 -12 0 12 24 36 48 60
x (µm)


60
48
36
24
12
0
-12
-24
-36
-48
-60
y (µm)
-60 -48 -36 -24 -12 0 12 24 36 48 60
3
2
1
0
-1
-2
-3

2.

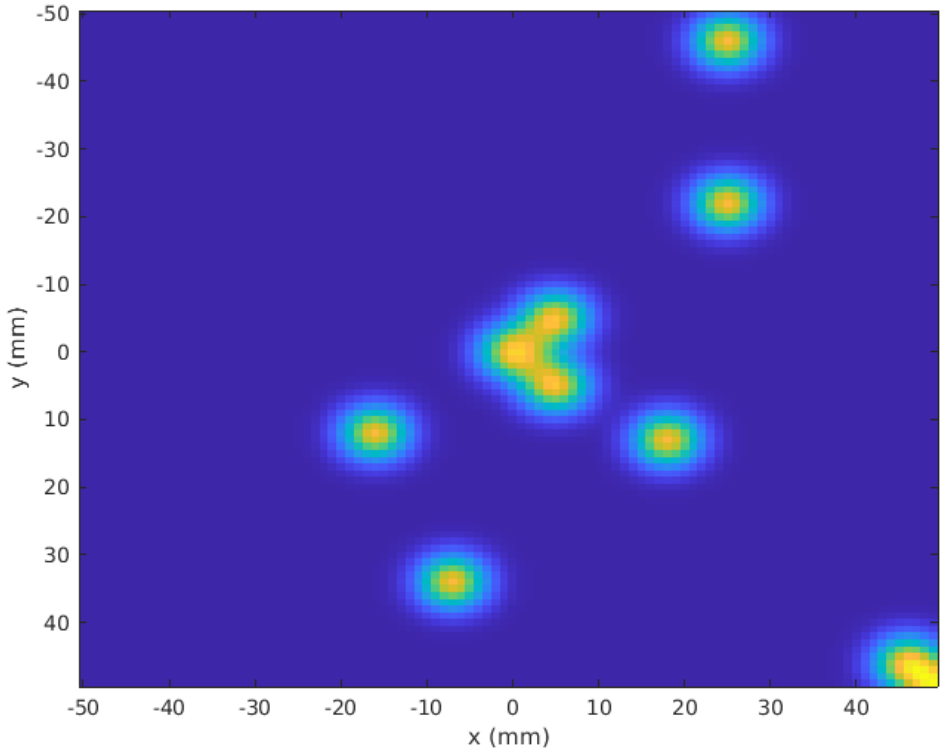


The images are resolved, but barely, as there is a slight dip in the middle.



This looks like the line plot, but viewed from the top angle. The Irradiance peaks are barely touching.

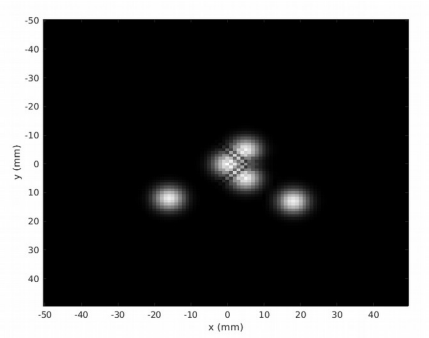
3.



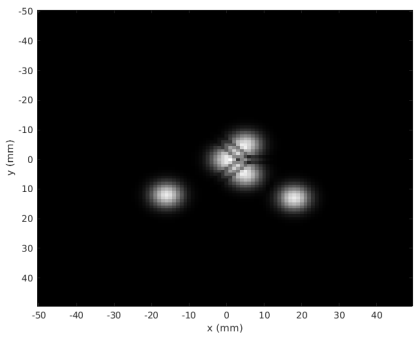
There is interference phenomena. It looks like the intensities of objects that are close together are influencing each other.

In the resolution section, the beams were Hyper Gaussian, so there was far less power around the edges.

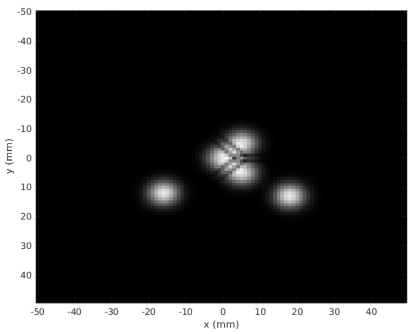
4.



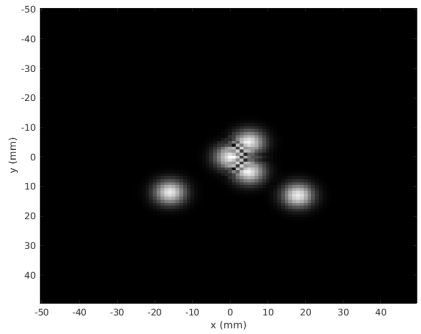
Defocus pi



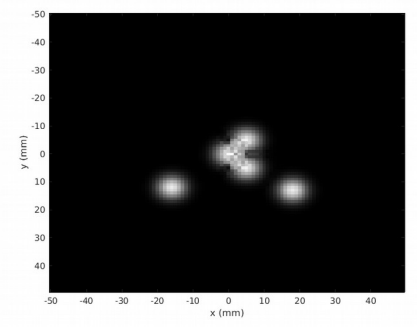
Defocus 2pi



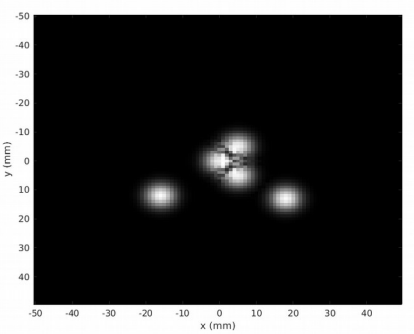
Defocus 6pi



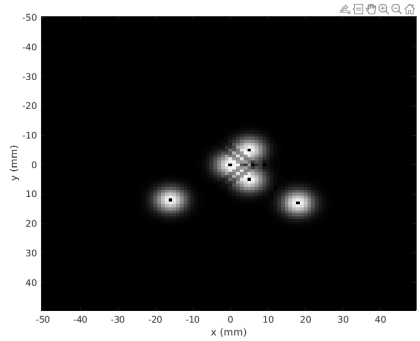
Horizontal Coma pi



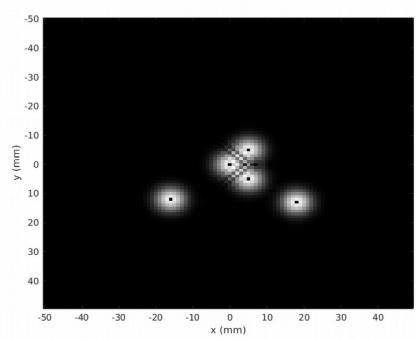
Horizontal Coma 2pi



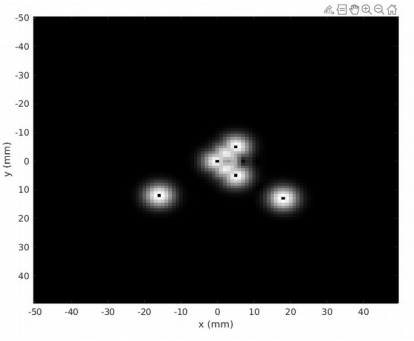
Horizontal Coma 6pi



Oblique Astigmatism pi

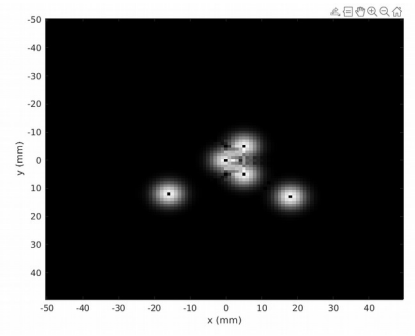
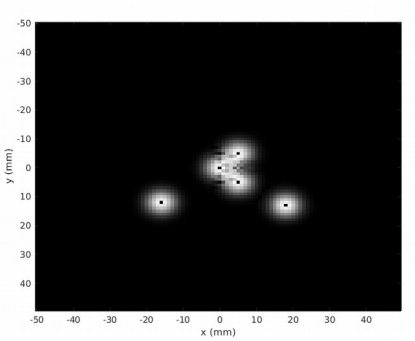
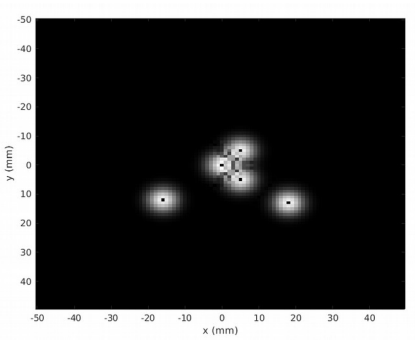
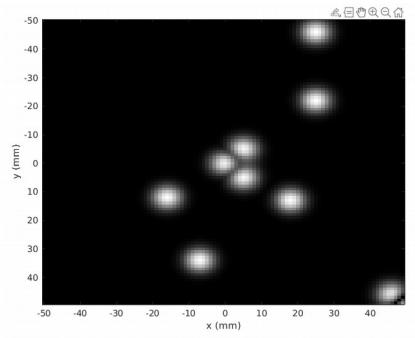
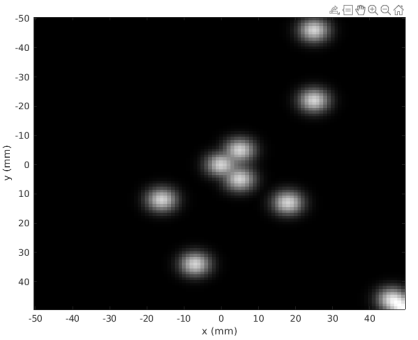
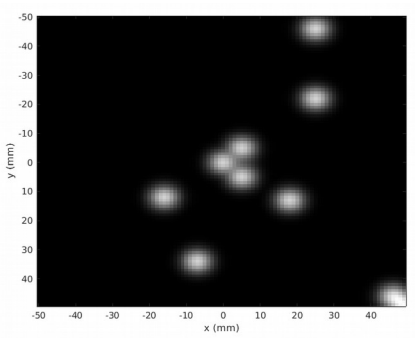
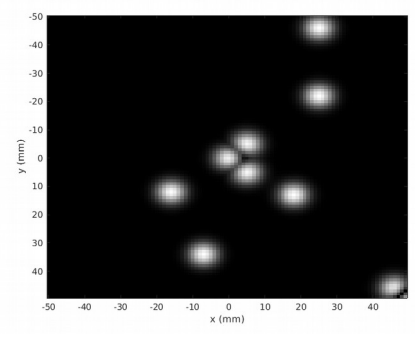
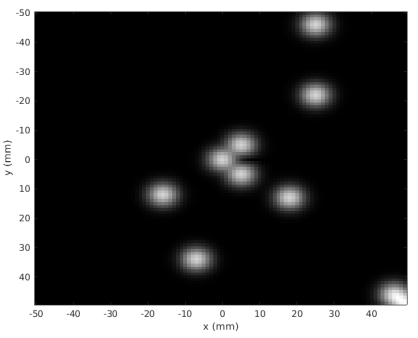
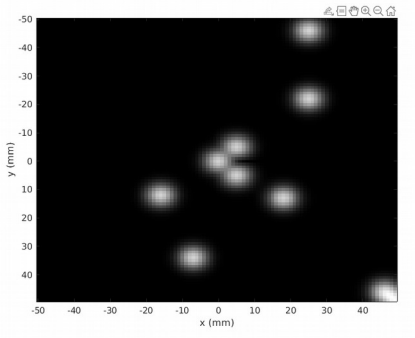


Oblique Astigmatism 2pi

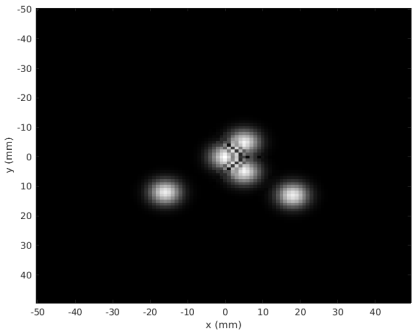
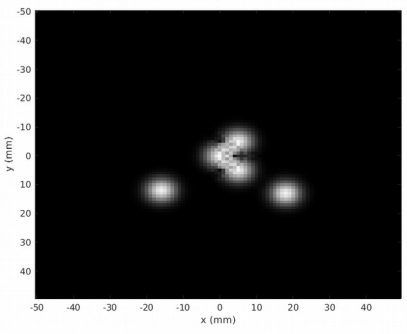
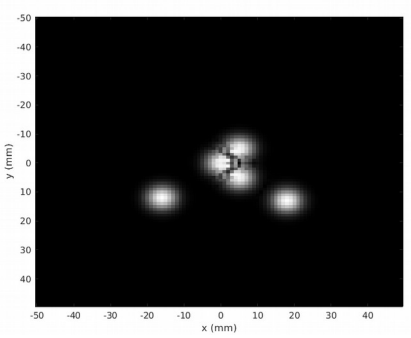
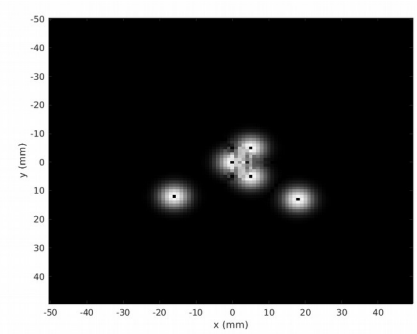
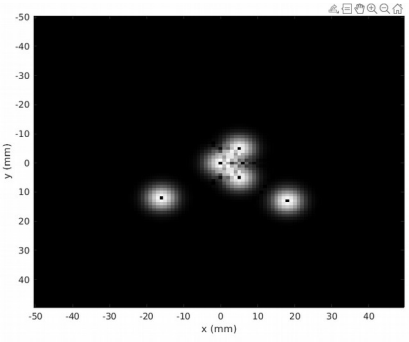
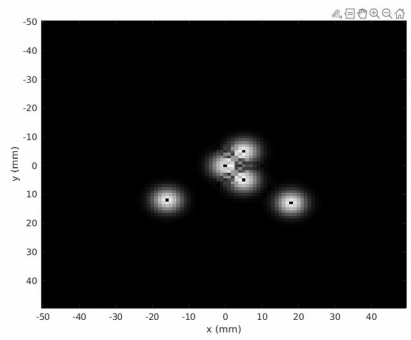
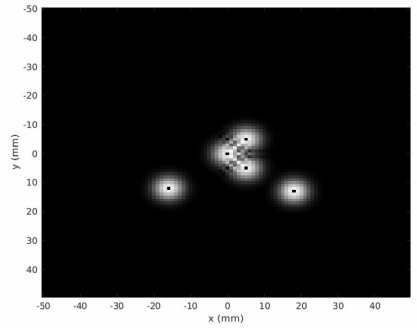
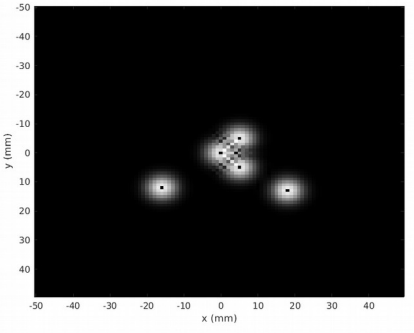
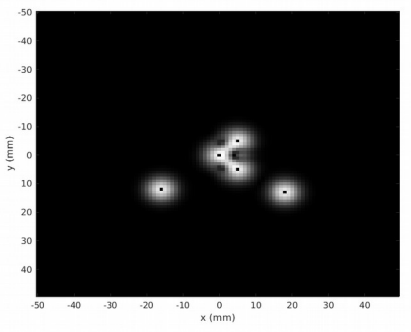


Oblique Astigmatism 6pi

It seems that the patterns are brightest towards the center at 6pi, since there is the most interference. For all the ones that are close together, there is lots of interference.

 Oblique Trefoil Pi	 Oblique Trefoil 2Pi	 Oblique Trefoil 6Pi
 Tilt Pi	 Tilt 2Pi	 Tilt 6Pi
 Tip Pi	 Tip 2pi	 Tip 6pi

For Tilt and Tip, they are brightest at Pi. This is because the phases do not cancel out.

 Vertical Coma Pi	 Vertical Coma 2Pi	 Vertical Coma 6Pi
 Defocus Pi	 Defocus 2Pi	 Defocus 6Pi
 Vertical Astigmatism Pi	 Vertical Astigmatism 2Pi	 Vertical Astigmatism 6Pi

Appendix Code

```
x = -512:1:512-1; %initial mesh grid for the space domain
y = -512:1:512-1;
[X,Y] = meshgrid(x,y);
E_1 = exp(-((X.^2 + Y.^2)/((90.27)^2)).^50); %The initial electric field
x2 = -1:1/(512):1-(1/(512)); %initial mesh grid for the space domain
y2 = -1:1/(512):1-(1/(512));
[X2,Y2] = meshgrid(x2,y2);
E_2 = (1/((10*(50e-3))^2))*ifftshift(ifft2(fftshift(E_1))); %The fourier transform of the electric field
E_3 = exp(-((X.^2 + Y.^2)/((90.27)^2)).^50).*exp(1i*0.035.*X); %The second electric field
E_4 = (1/((10*(50e-3))^2))*ifftshift(ifft2(fftshift(E_3))); %The fourier transform of the electric field

figure(1);
%plot(x2,abs(E_2(1024/2+1,:)).^2); xlabel('x (mm)'); ylabel('PSF');
%hold on;
%plot(x2,abs(E_4(1024/2+1,:)).^2);
%plot(x2,abs(E_2(1024/2+1,:).^2+E_4(1024/2+1,:).^2));
%xlim([-0.1,0.1])
%xticklabels([-120,-60,0,60,120])
imagesc(x,y,abs(E_2.^2 + E_4.^2).^0.5)
xlim([-20,20])
ylim([-20,20])
xlabel('x (mm)')
ylabel('y (mm)')

ax = gca;
ax.YDir = 'normal';
colorbar
colormap gray

x = -50:1:50-1; %initial mesh grid for the space domain
y = -50:1:50-1;
[X,Y] = meshgrid(x,y);

E_1 = exp(-(X).^2/16).*exp(-(Y).^2/16).*exp(1i*pi*(3*((X/4).^2 + (Y/4).^2)-1).*Y);
E_2 = exp(-(X-5).^2/16).*exp(-(Y-5).^2/16).*exp(1i*pi*(3*(((X-5)/4).^2 + ((Y-5)/4).^2)-1).*(Y-5));
E_3 = exp(-(X-5).^2/16).*exp(-(Y+5).^2/16).*exp(1i*pi*(3*(((X-5)/4).^2 + ((Y+5)/4).^2)-1).*(Y+5));
E_4 = exp(-(X+16).^2/16).*exp(-(Y-12).^2/16).*exp(1i*pi*(3*(((X+16)/4).^2 + ((Y-12)/4).^2)-1).*(Y-
12));
E_5 = exp(-(X-18).^2/16).*exp(-(Y-13).^2/16).*exp(1i*pi*(3*(((X-18)/4).^2 + ((Y-13)/4).^2)-1).*(Y-
13));


figure(1);
imagesc(x,y,abs(E_1 + E_2 + E_3 + E_4+ E_5))
colormap gray
xlabel('x (mm)')
```

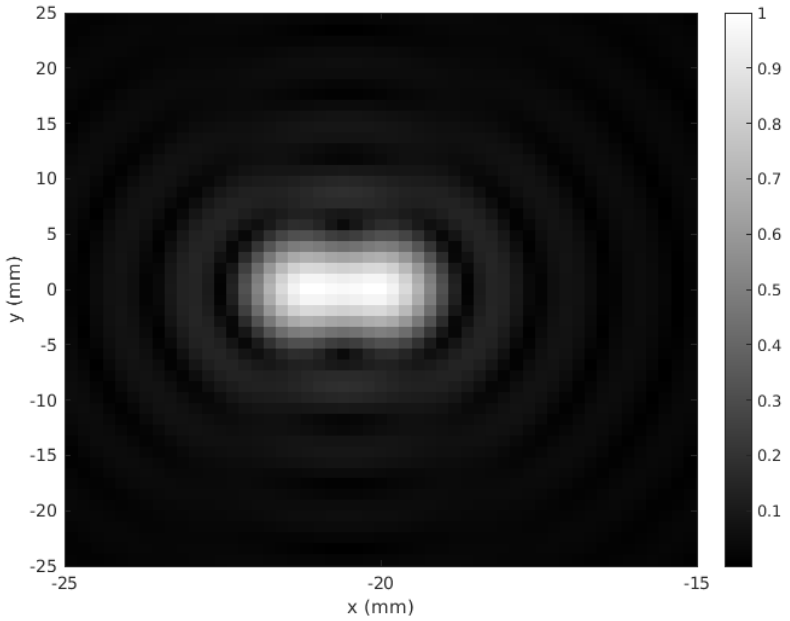
```
ylabel('y (mm)')
```

Part 3: Incoherent Images

The key principle of this part is to understand that the incoherent impulse function is the absolute value of the coherent impulse function squared.

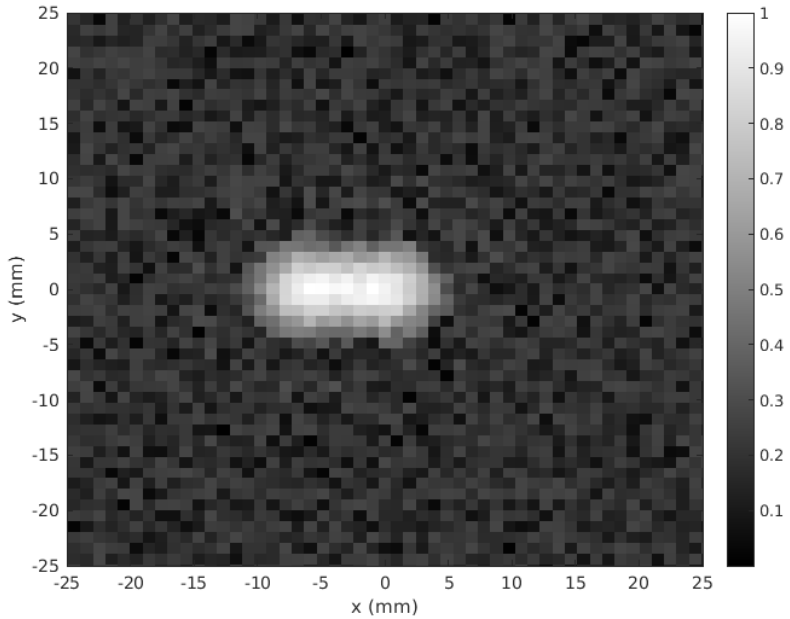
$$\tilde{h}(x,y) = |h(x,y)|^2$$

1.

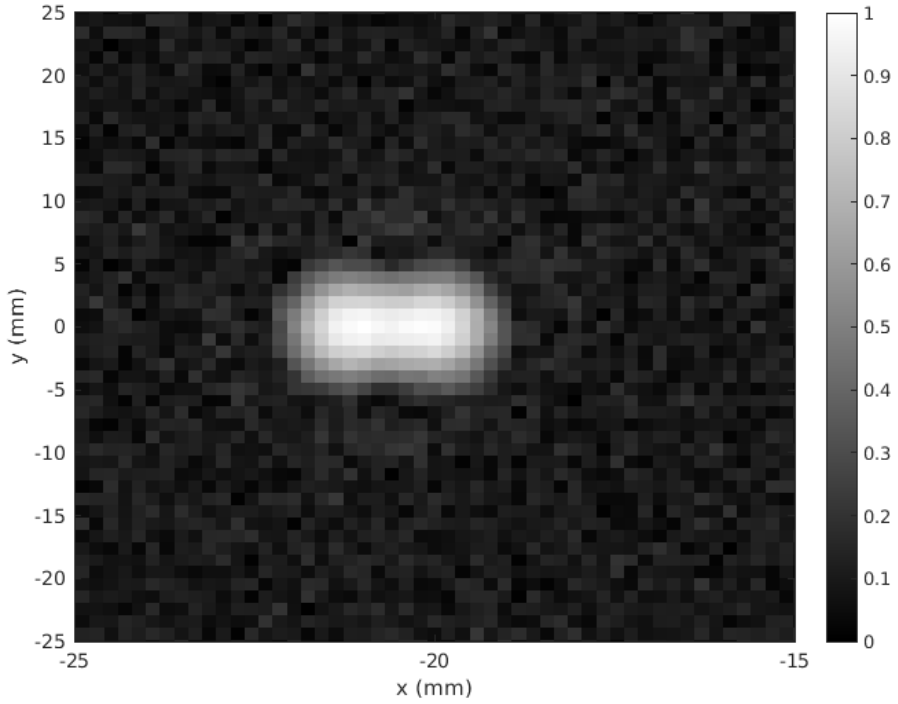


Compared to Part 2, the PSFs are smoother, but they have less contrast because the impulse response does not have a phase component. The two peaks are far more resolvable than before.

2.



This is a uniform distribution of noise for a single seed, where the value of η is 50.

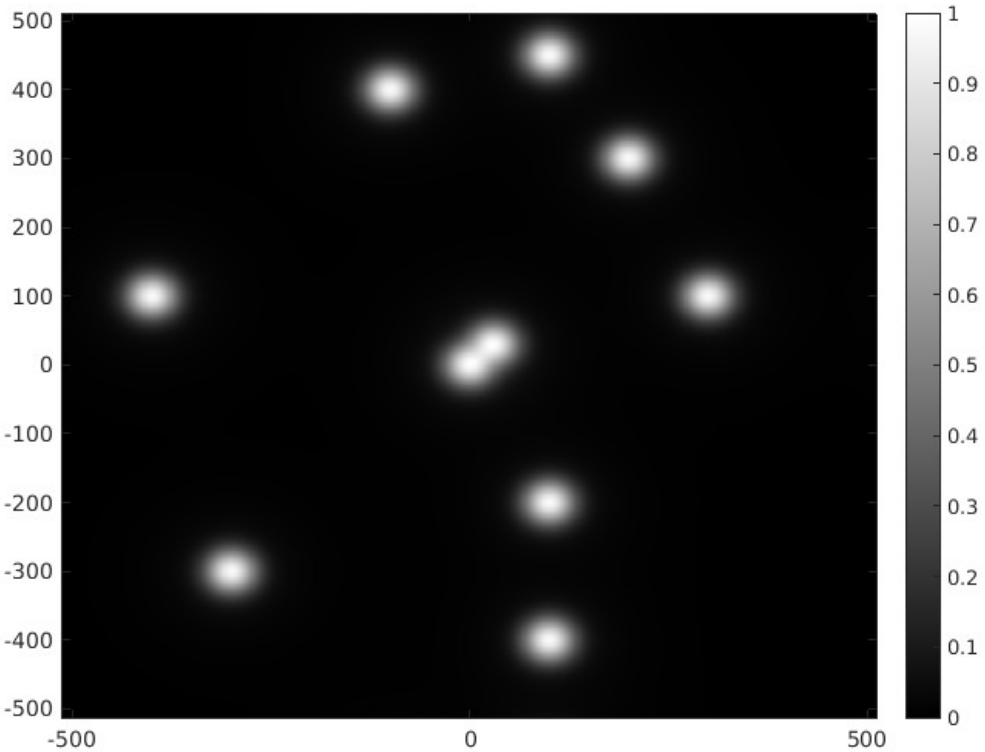


This is a plot of ten averaged intensities:

$$I_{i,ave}(x_i, y_i) = (1/N) \sum_{j=1}^{N} I_{j,n,i}$$

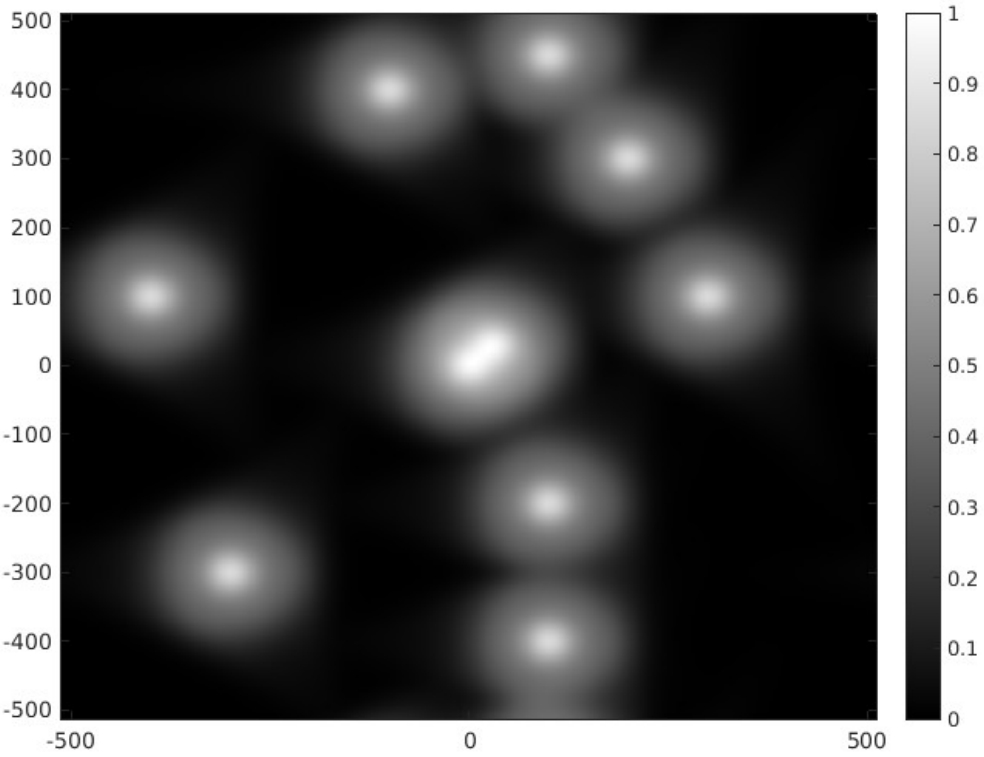
The diffraction rings are no longer visible because they have been filtered out by the noise, so the peaks are easier to resolve in this case.

3.



Compared to Part 2.3,the Gaussian objects are smaller, but the contrast is not as high because the phase is eliminated.

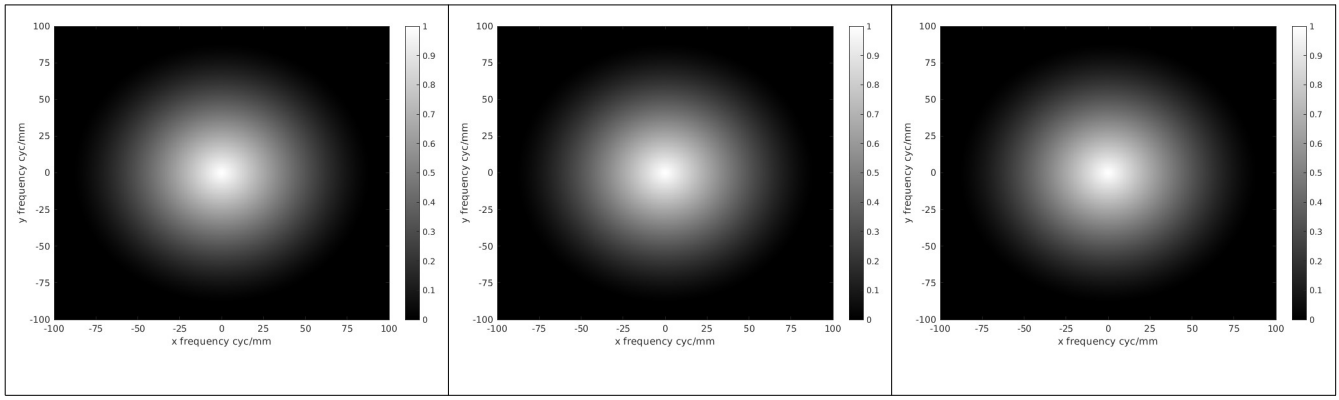
This is for an Oblique Trefoil. The diffraction rings

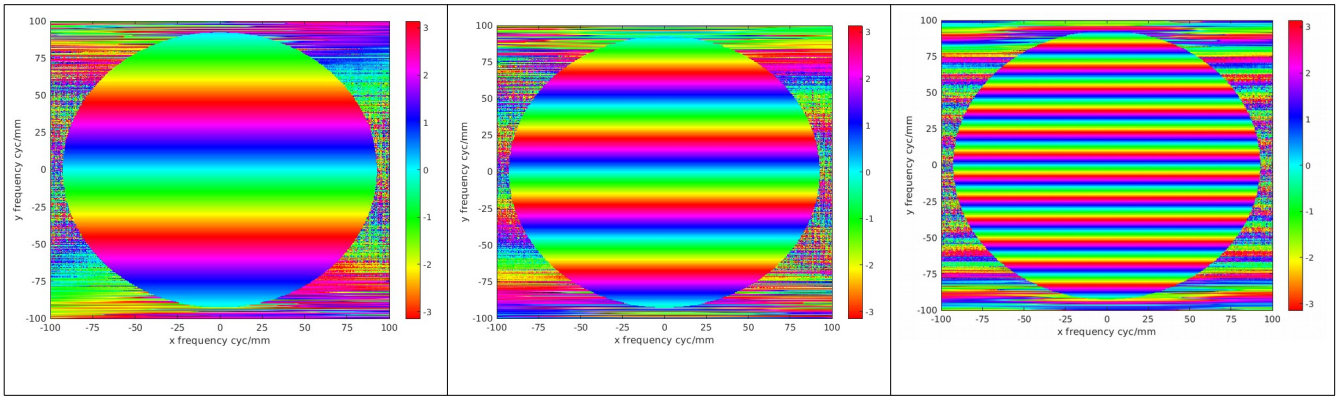


surrounding the points are far more visible, because of the lower contrast.
4.

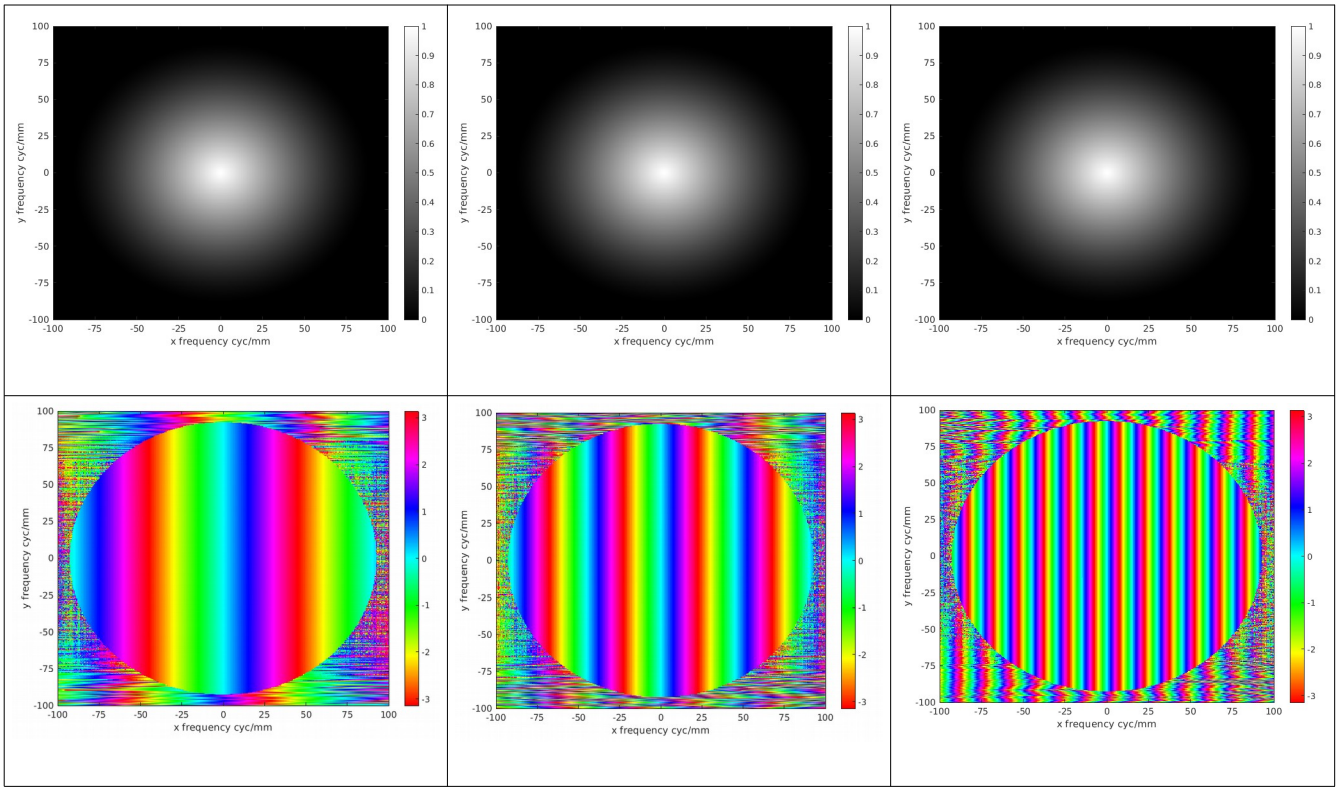
In the following series of images, the phase changes are π, 2π, and 6π from left to right. The amplitudes are the top three and the phases are the bottom three.

Tilt

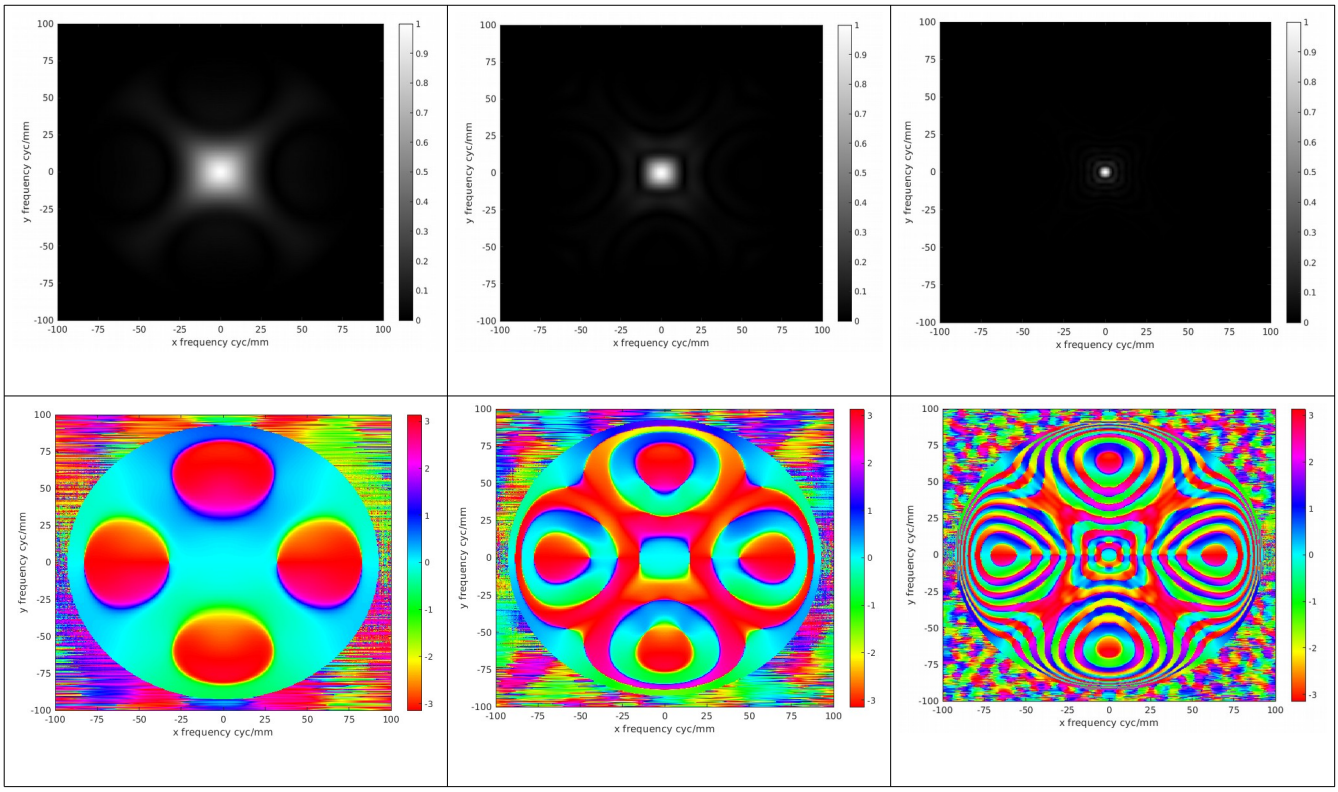



y frequency cyc/mm
x frequency cyc/mm
100
75
50
25
0
-25
-50
-75
-100
3
2
1
0
-1
-2
-3

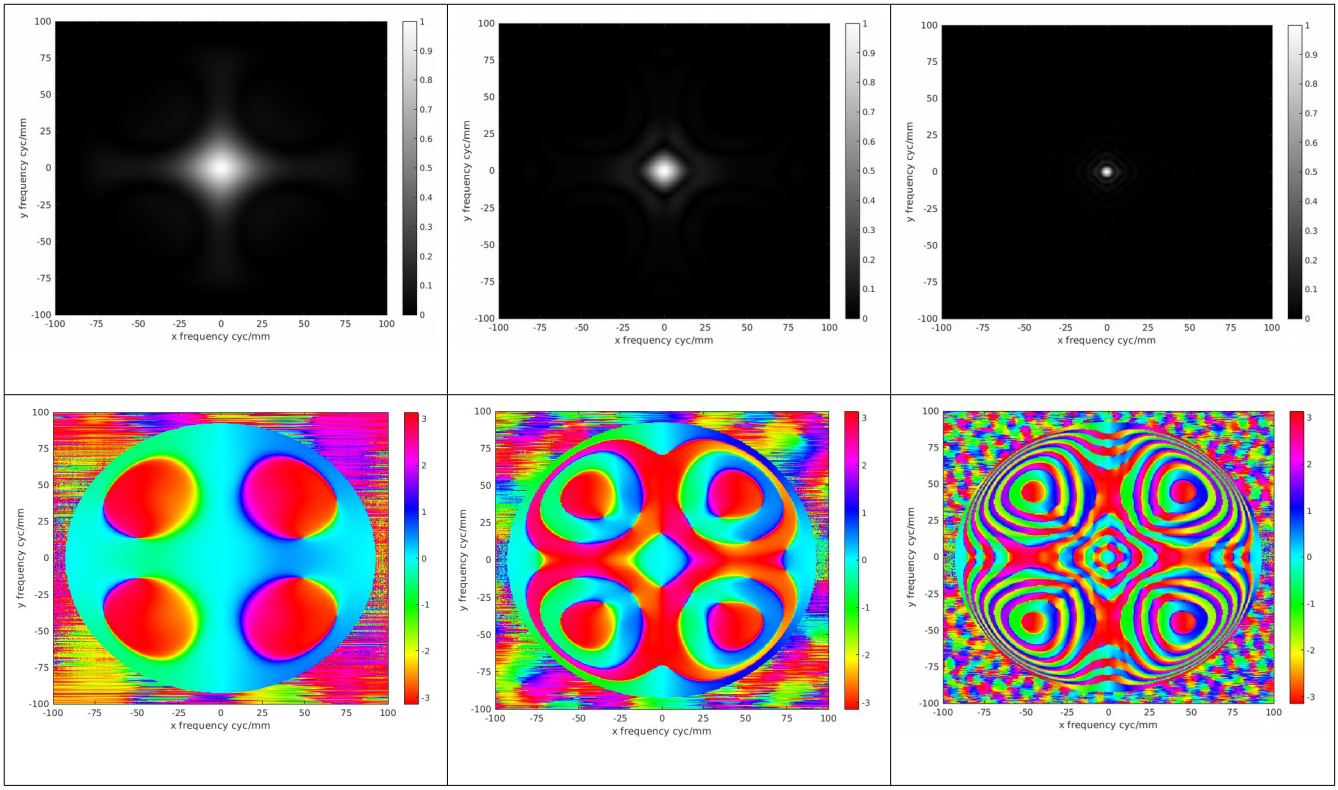
Tip


y frequency cyc/mm
x frequency cyc/mm
100
75
50
25
0
-25
-50
-75
-100
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
3
2
1
0
-1
-2
-3

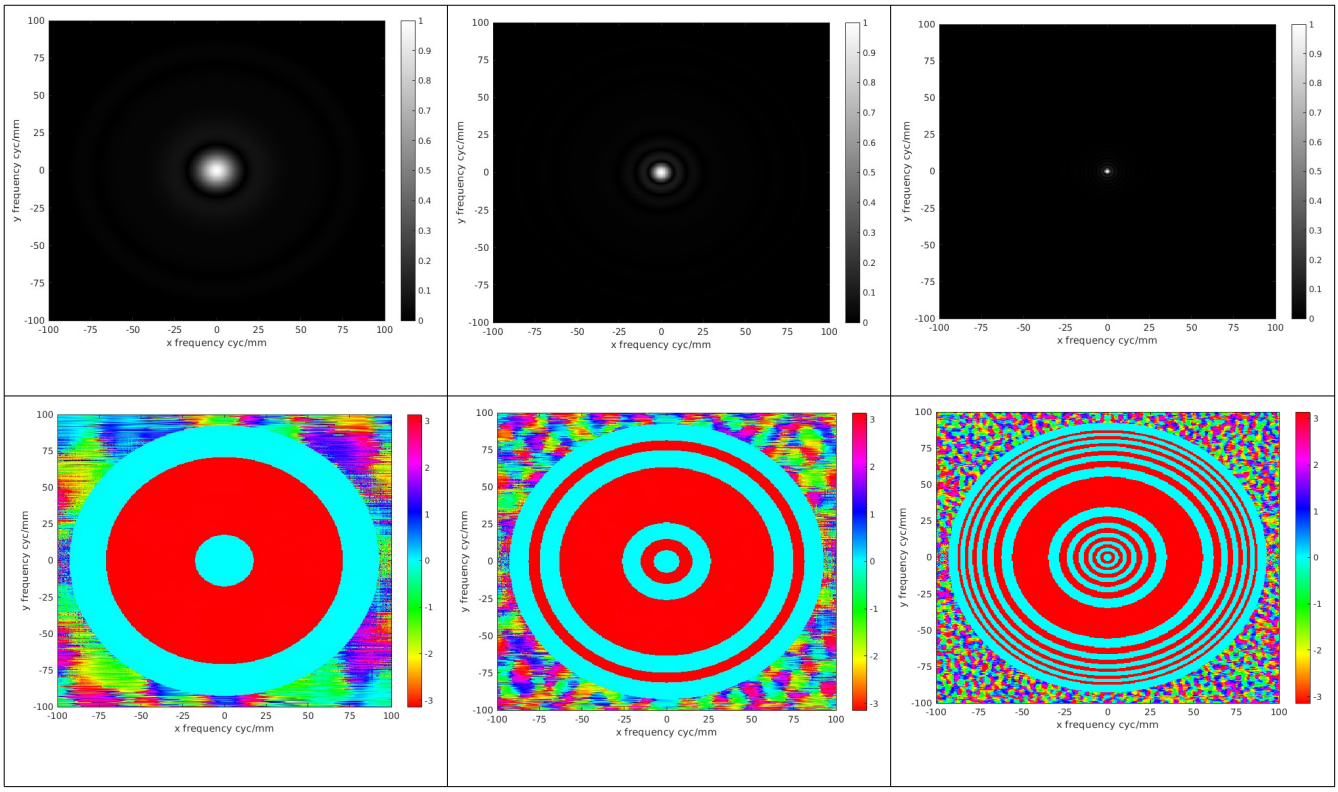
Vertical Astigmatism



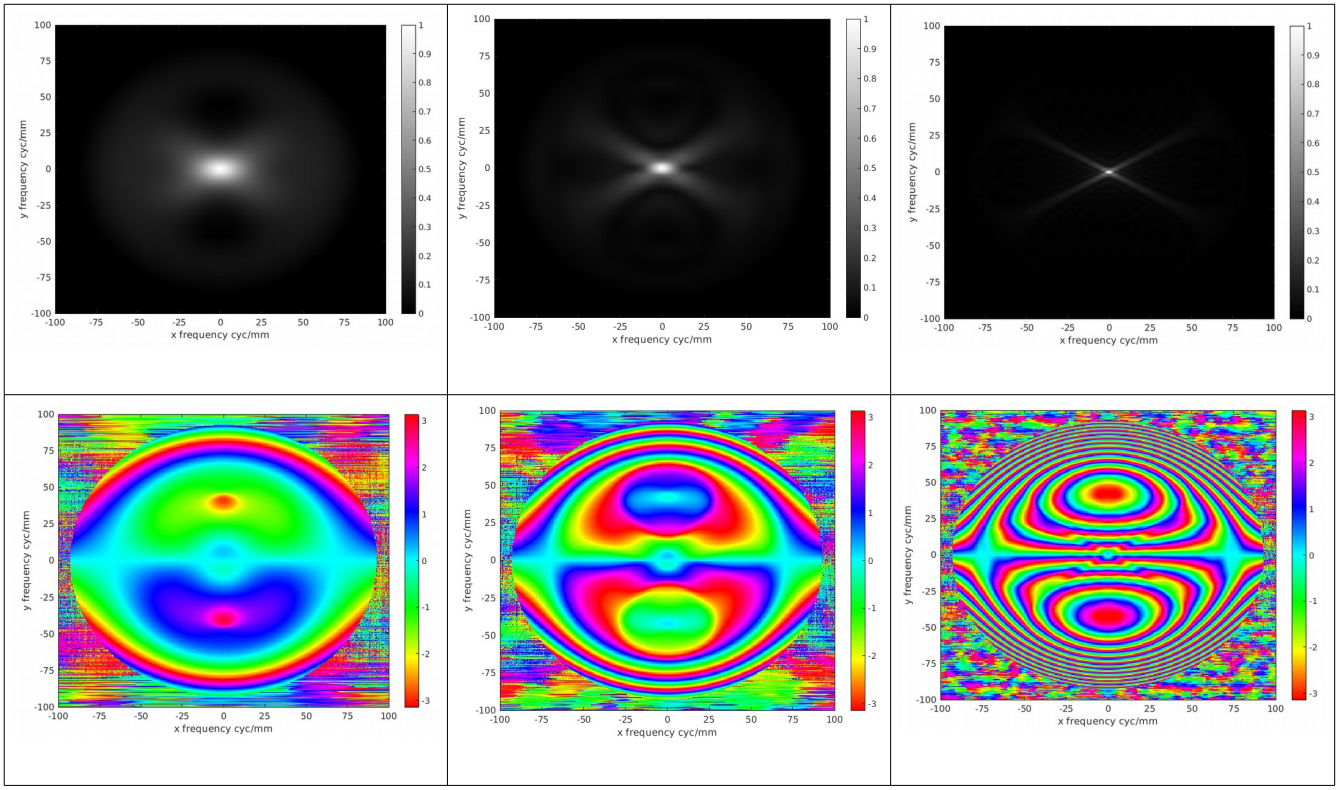
Oblique Astigmatism



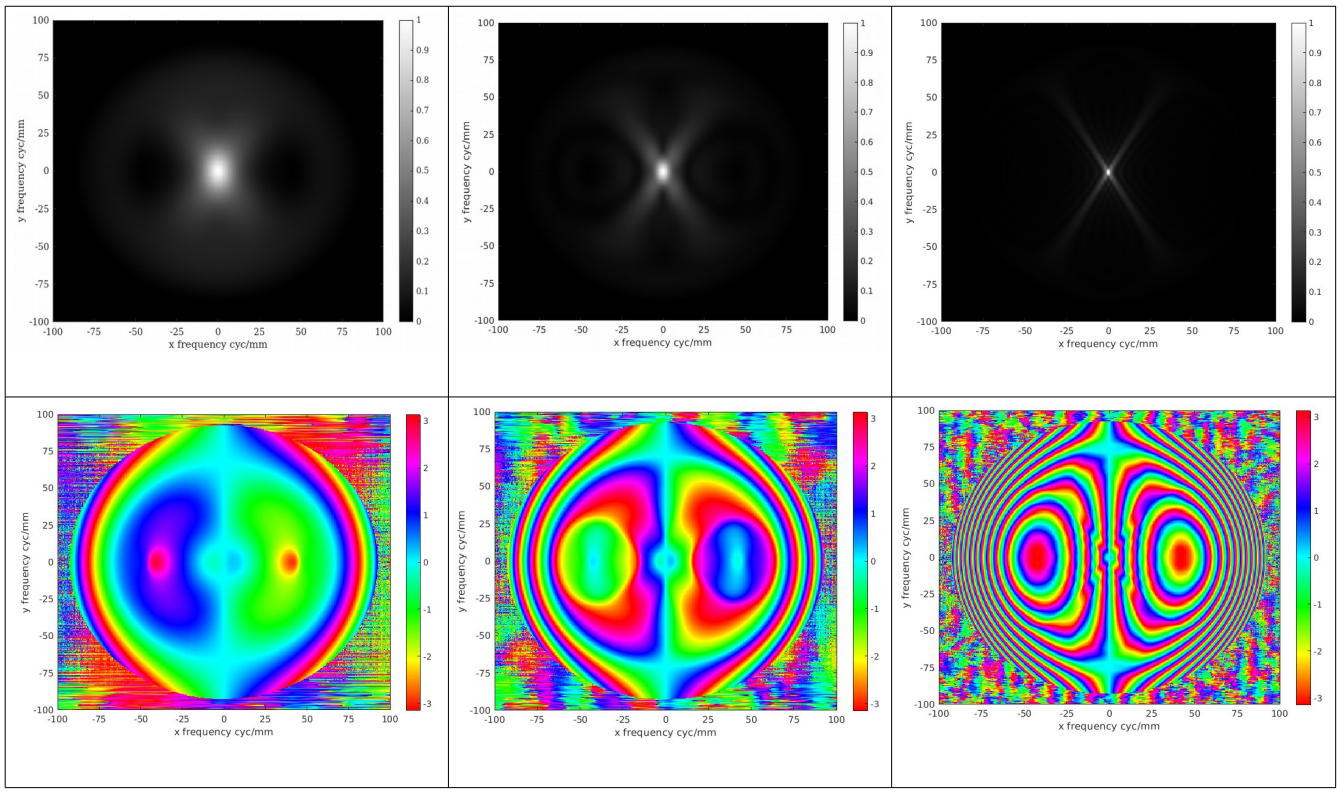
Defocus



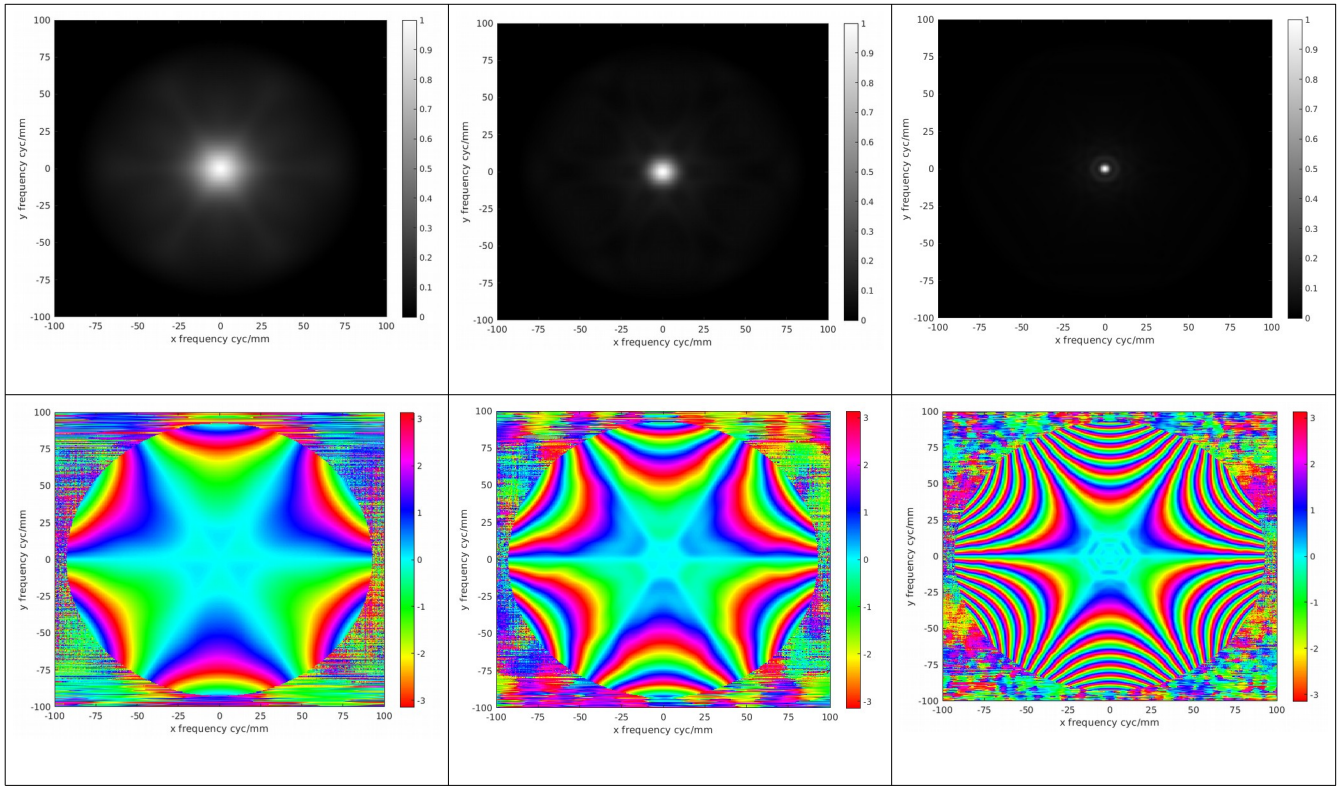
Vertical Coma



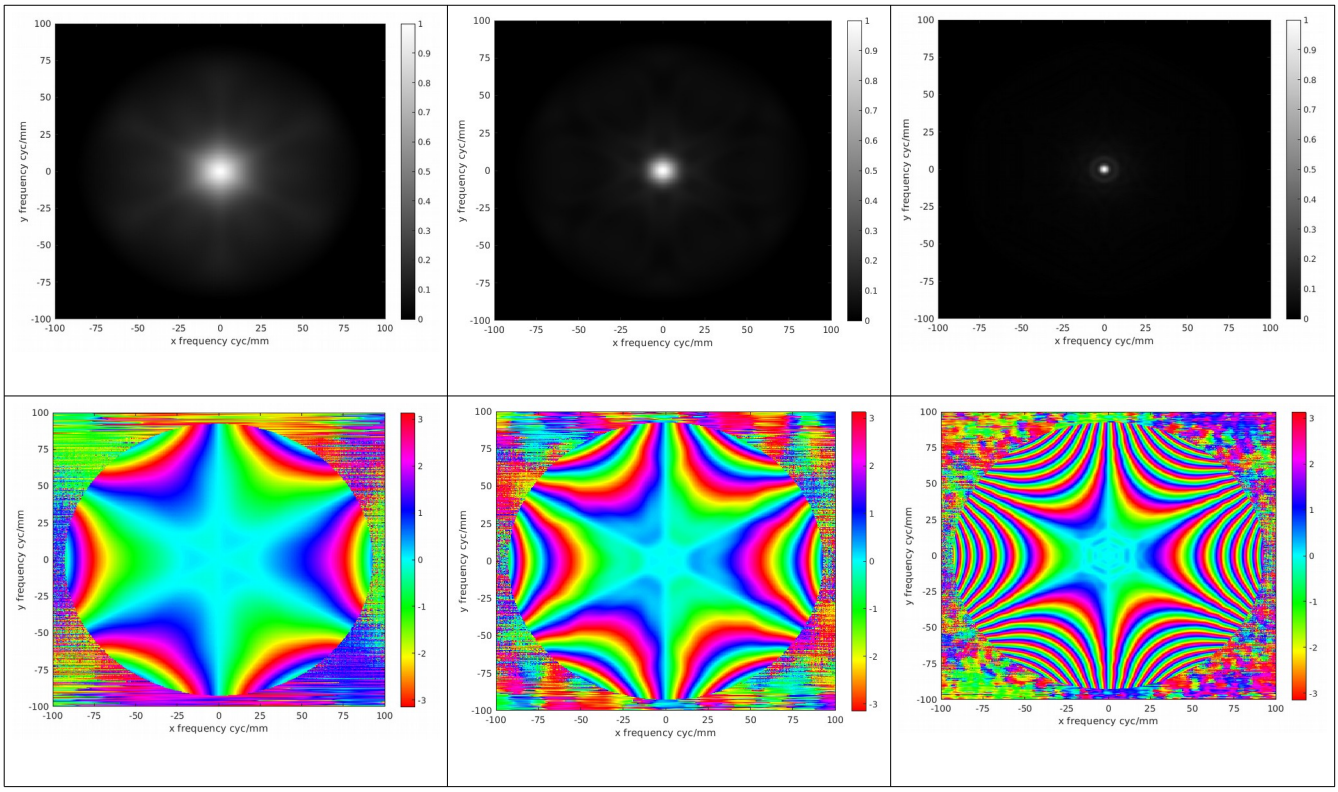
Horizontal coma



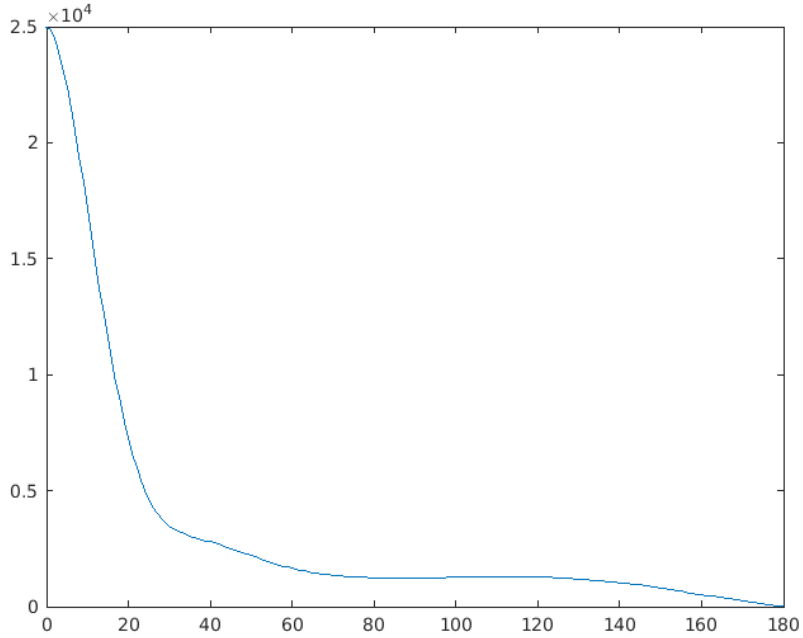
Vertical Trefoil



Oblique Trefoil

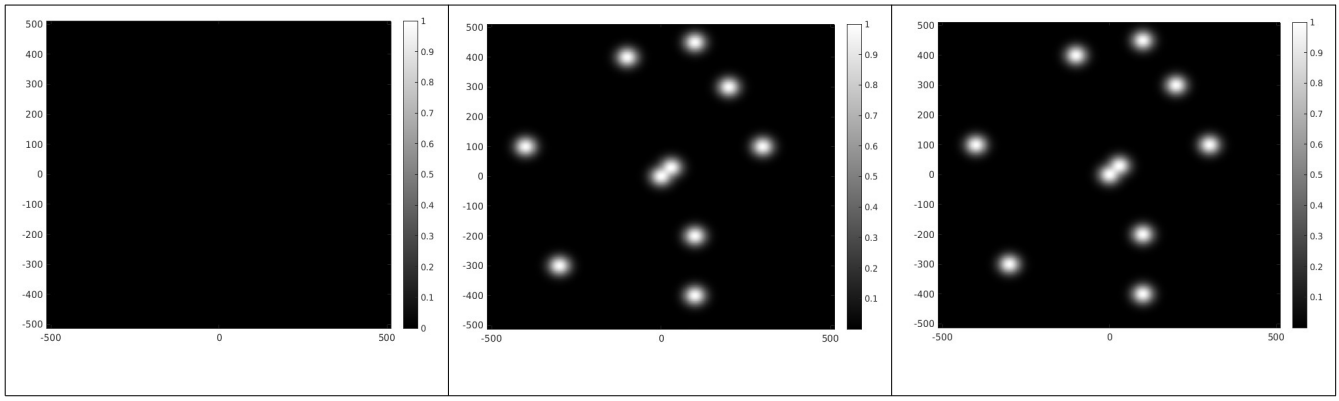


As the degree of aberration increases, the frequency of the phases increases, and the function is spread out more over the plane. This is because degree indicates frequency. The MTF reaches zero at radius/(phase aberration).



This is a line plot of the MTF of the oblique trefoil at phase 2π.

5.



At NSR = 0, the image is not visible because there is division by zero, which is not possible.

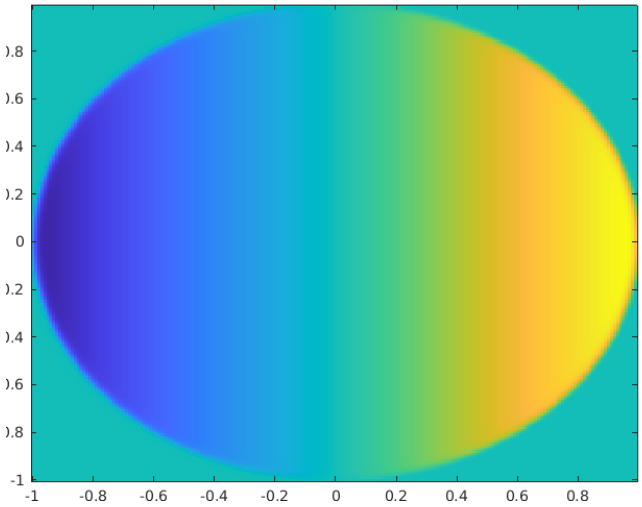
At NSR = 0.01 and 0.25, the reconstruction is much better because there NSR offsets division by zero. An NSR of 0.25 creates a little bit more noise than an NSR of 0.01.

Code

```
x = -512:1:512-1; %initial mesh grid for the optical spatial frequency domain
y = -512:1:512-1;
[X,Y] = meshgrid(x,y);
E_1 = exp(-((X.^2 + Y.^2)/((90.27)^2)).^50); %The initial electric field
x2 = -1:1/(512):1-(1/(512)); %initial mesh grid for the space domain
y2 = -1:1/(512):1-(1/(512));
[X2,Y2] = meshgrid(x2,y2);
E_2 = zeros(1024); %The fourier transform of the electric field
E_2(513,513) = 1;
E_2(513,507) = 1;
E_1 = ifftshift(fft2(fftshift(E_1)));
E_1 = abs(E_1).^2; %the important thing is to take the absolute value of and square
the impulse function
E_1 = fftshift(ifft2(ifftshift(E_1)));
E_3 = ifftshift(fft2(fftshift(E_2))).*(E_1);
E_4 = fftshift(ifft2(ifftshift(E_3)));
E_4(1,1) = 0;

rng(1) %random seed
r1 = -1 + 2*rand(1024,1024); %every cell (1024x1024) has a unique random value

E_5 = E_4 + 50*r1;
E_5(1,1) = 0; %set one pixel to zero

figure(1);

imagesc(x2,y2,sqrt(abs(E_5)/max(abs(E_5),[],'all')))

xlabel('x (mm)')
ylabel('y (mm)')
xlim([-0.05,0.05])
ylim([-0.05,0.05])
xticklabels([-25,-20,-15,-10,-5,0,5,10,15,20,25])
yticklabels([-25,-20,-15,-10,-5,0,5,10,15,20,25])
ax = gca;
ax.YDir = 'normal';
colorbar
colormap gray

x = -1:1/(512):1-(1/(512)); %initial mesh grid for the space domain
y = -1:1/(512):1-(1/(512));
[X,Y] = meshgrid(x,y);

x2 = -512:1:512-1; %initial mesh grid for the space domain
y2 = -512:1:512-1;
[X2,Y2] = meshgrid(x2,y2);

A = exp(-((X2.^2 + Y2.^2)/(((90.27)^2))).^50).*exp(1i*2*pi.*(((X2.^2 +
Y2.^2)/(((90.27)^2))).^1.5).*cos(3*atan(Y./(X+.001)))); %The initial electric field
A(:,1:512) = conj(A(:,1:512)); %important to take phase shift into account
I_1 = exp(-(X).^2/0.002).*exp(-(Y).^2/0.002);
I_2 = exp(-(X-(30/(512))).^2/0.002).*exp(-(Y-(30/(512))).^2/0.002);
I_3 = exp(-(X-(100/(512))).^2/0.002).*exp(-(Y+(200/(512))).^2/0.002);
I_4 = exp(-(X+(400/(512))).^2/0.002).*exp(-(Y-(100/(512))).^2/0.002);
I_5 = exp(-(X-(200/(512))).^2/0.002).*exp(-(Y-(300/(512))).^2/0.002);
I_6 = exp(-(X+(300/(512))).^2/0.002).*exp(-(Y+(300/(512))).^2/0.002);
I_7 = exp(-(X+(100/(512))).^2/0.002).*exp(-(Y-(400/(512))).^2/0.002);
```

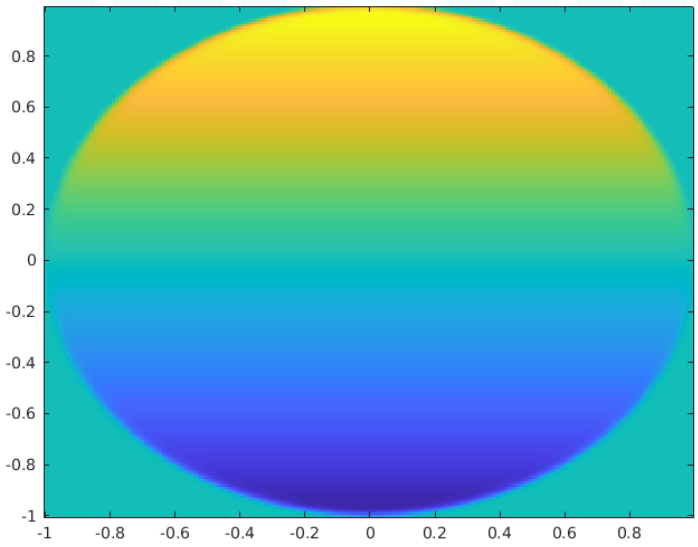
```
I_8 = exp(-(X-(100/(512))).^2/0.002).*exp(-(Y-(450/(512))).^2/0.002);
I_9 = exp(-(X-(100/(512))).^2/0.002).*exp(-(Y+(400/(512))).^2/0.002);
I_10 = exp(-(X-(300/(512))).^2/0.002).*exp(-(Y-(100/(512))).^2/0.002);

I_0 = I_1 + I_2 + I_3 + I_4 + I_5 + I_6 + I_7 + I_8 + I_9 + I_10;

A = ifftshift(fft2(fftshift(A)));
A = abs(A).^2; %the important thing is to take the absolute value of and square the
impulse function
A = fftshift(ifft2(ifftshift(A)));

E_3 = ifftshift(fft2(fftshift(I_0))).*A;
E_4 = fftshift(ifft2(ifftshift(E_3)));
E_4(1,1) = 0; %set pixel to zero

%figure(1);
%imagesc(x2,y2,sqrt(abs(E_4)))
%plot(x2,abs(A(1024/2+1,:)));
%xlim([0 180]);

E_5 = fftshift(ifft2(ifftshift((conj(A).*E_3)./(abs(A).^2 + 0.25))));

imagesc(x2,y2,sqrt(abs(E_5)/max(abs(E_5),[],'all')))
%imagesc(x2,y2,angle(A))

%xlim([-200,200])
%ylim([-200,200])
%xticklabels([-100,-75,-50,-25,0,25,50,75,100])
%yticklabels([-100,-75,-50,-25,0,25,50,75,100])
%xlabel('x frequency cyc/mm')
%ylabel('y frequency cyc/mm')
ax = gca;
ax.YDir = 'normal';
colorbar
colormap gray
```

Tilt



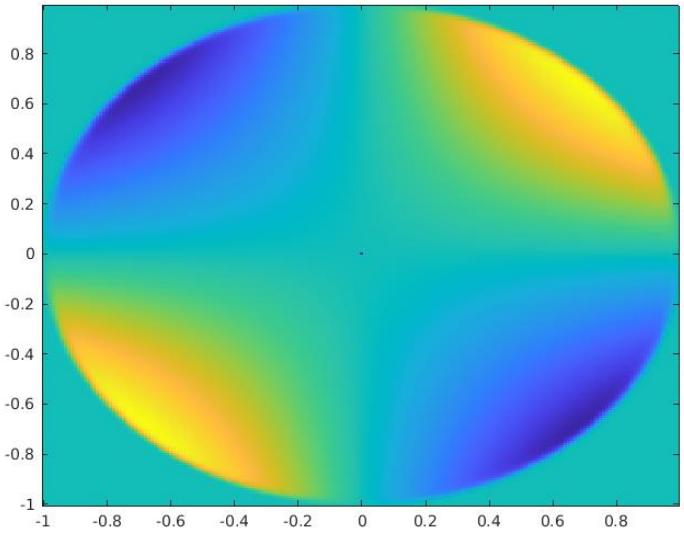

```
x = -1:0.01:1-0.01; %initial mesh grid for the space domain
y = -1:0.01:1-0.01;
[X,Y] = meshgrid(x,y);
E_1 = exp(-((X.^2 + Y.^2)).^50).*X; %The initial electric field

%E_1 = -E_1; %important to take phase shift into account

imagesc(x,y,E_1)

ax = gca;
ax.YDir = 'normal';
```

Tip



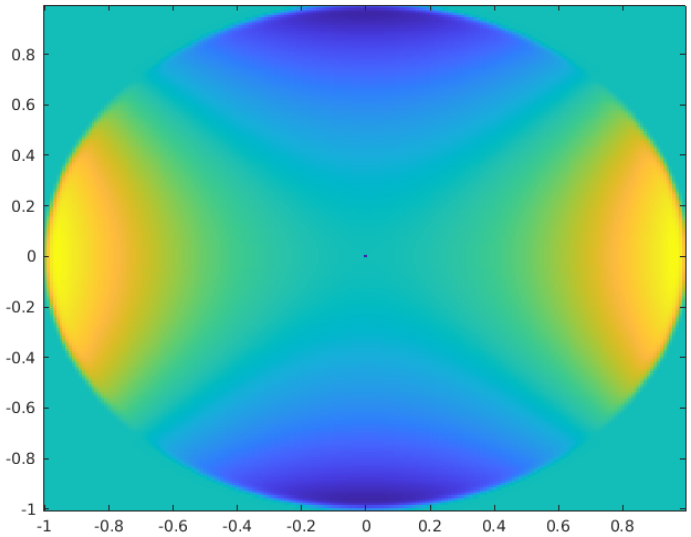
```
x = -1:0.01:1-0.01; %initial mesh grid for the space domain
y = -1:0.01:1-0.01;
[X,Y] = meshgrid(x,y);
E_1 = exp(-((X.^2 + Y.^2)).^50).*Y; %The initial electric field

%E_1 = -E_1; %important to take phase shift into account

imagesc(x,y,E_1)

ax = gca;
ax.YDir = 'normal';
```

Oblique Astigmatism




```
x = -1:0.01:1-0.01; %initial mesh grid for the space domain
y = -1:0.01:1-0.01;
[X,Y] = meshgrid(x,y);
E_1 = exp(-((X.^2 + Y.^2)).^50).*(X.^2 + Y.^2).*sin(2*atan(Y./X)); %The initial electric field

%E_1 = -E_1; %important to take phase shift into account

imagesc(x,y,E_1)

ax = gca;
ax.YDir = 'normal';
```

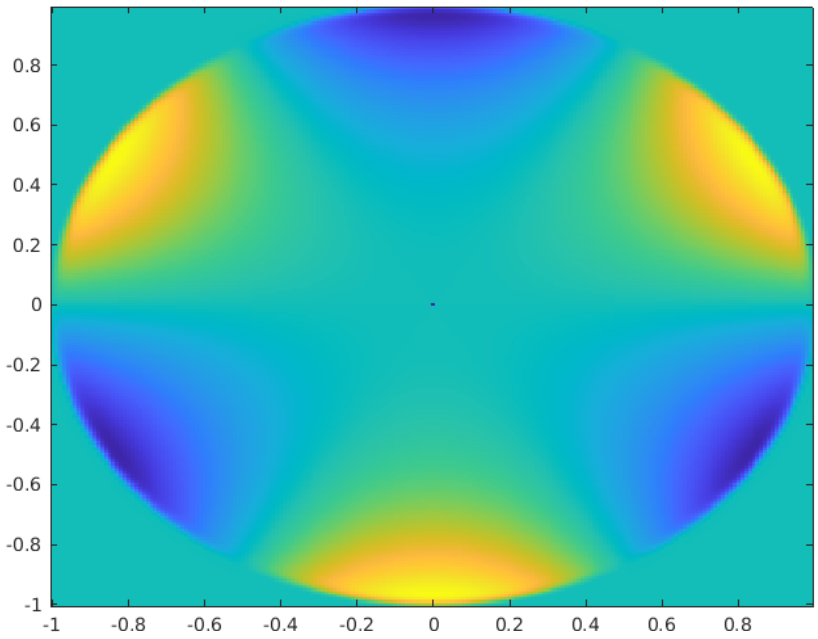
Vertical Astigmatism

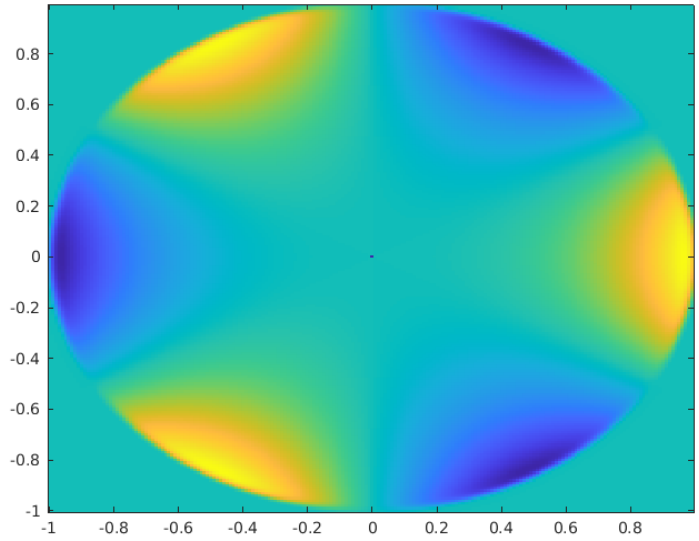



```
x = -1:0.01:1-0.01; %initial mesh grid for the space domain
y = -1:0.01:1-0.01;
[X,Y] = meshgrid(x,y);
E_1 = exp(-((X.^2 + Y.^2)).^50).*(X.^2 + Y.^2).*cos(2*atan(Y./X)); %The initial electric field

%E_1 = -E_1; %important to take phase shift into account

imagesc(x,y,E_1)

ax = gca;
ax.YDir = 'normal';
```

Vertical Trefoil

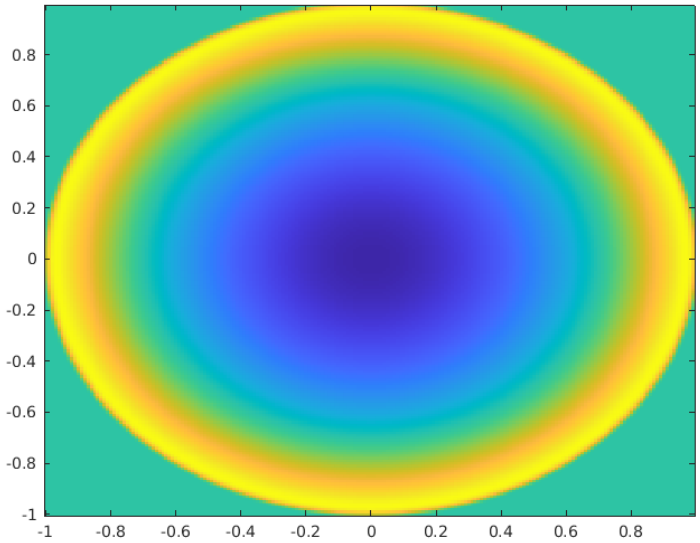
Horizontal Trefoil



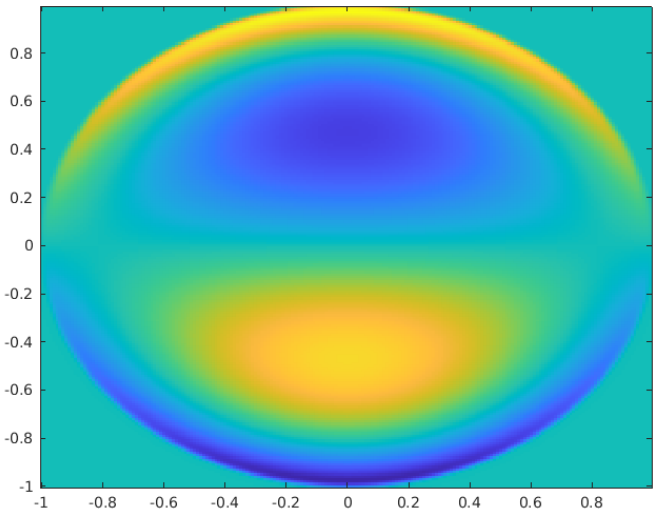



```
x = -1:0.01:1-0.01; %initial mesh grid for the space domain
y = -1:0.01:1-0.01;
[X,Y] = meshgrid(x,y);
E_1 = exp(-((X.^2 + Y.^2)).^50).*((X.^2 + Y.^2).^1.5).*cos(3*atan(Y./X)); %The initial electric field

E_1(:,1:100) = -E_1(:,1:100); %important to take phase shift into account

imagesc(x,y,E_1)

ax = gca;
ax.YDir = 'normal';
```

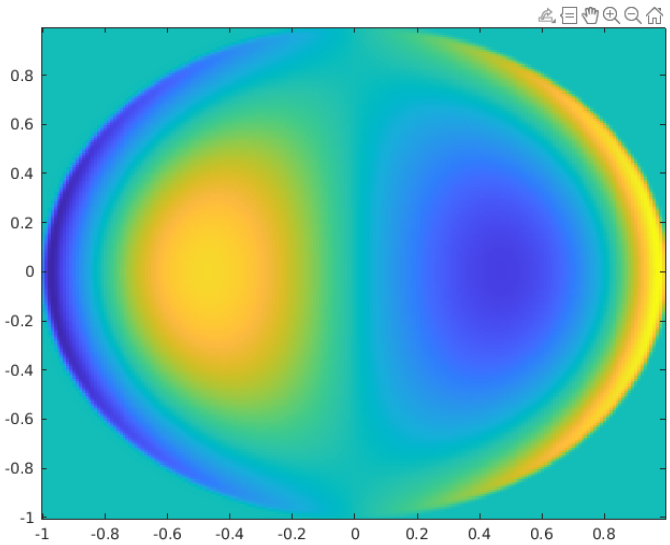
Defocus



Vertical Coma



Horizontal Coma


0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8