Table of Contents

Description

```
%title: MRAF Algorithm
%desc: an IFTA to determine phase distribution in CGH for beam shaping
%using amplitude freedom in certain region
%author: Muhammad Syahman Samhan
%email: mssamhan@students.itb.ac.id
%last update: November 27, 2019
%adapted from Gerchberg and Saxton (1971) and Pasienski & DeMarco
 (2008)
%inspired from
    %1. Musa Aydin (Sultan Mehmet Vakif University)
    %2. Dae Gwang Choi (KAIST, 2019)
    %3. Pasienski & DeMarco (2008)
```

General Variables

clc clear all

```
close all
pixelx = 1024;
pixely = 1024;
padx = 2*pixelx;
pady = 2*pixely;
sizex = 0.864;
sizey = 0.864;
x = linspace(-sizex, sizex, padx);
y = linspace(-sizey, sizey, pady);
[X,Y] = meshgrid(x,y);  %meshgrid for Input Beam
slmaperture = (X > -sizex/2 & X < sizex/2 & Y > -sizey/2 & Y <
 sizey/2);
x0=0; y0=0;             %center of CGH and Input Beam
i_num = 30;             %number of iteration
error = [];             %error array is empty for first
```

Input Variables

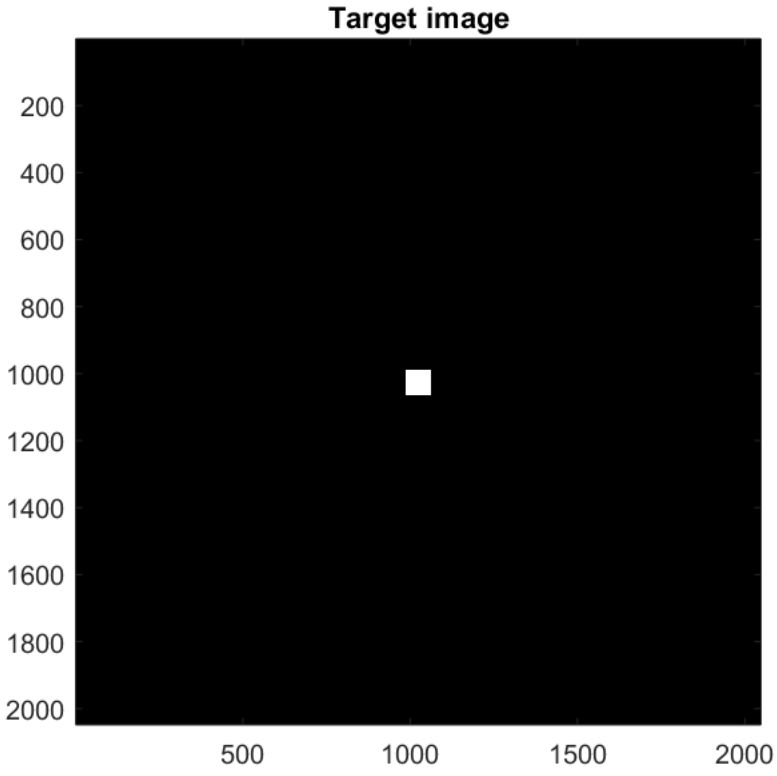
```
R = 200;
mu = 0;
B_lin = 0;
B_con = 0;
aspect_ratio = 1;
m = 0.98;
```

Generate Gaussian Input Beam

```
sigma = 0.74;                                   %beam waist in cm
 (need to be corrected)
A = 1;                                          %input amplitude
theta = ((X-x0).^2 + (Y-y0).^2)./(2*sigma^2);   %phase of Input beam
input = A*exp(-theta).*slmaperture;             %input amplitude for a
 given position
```

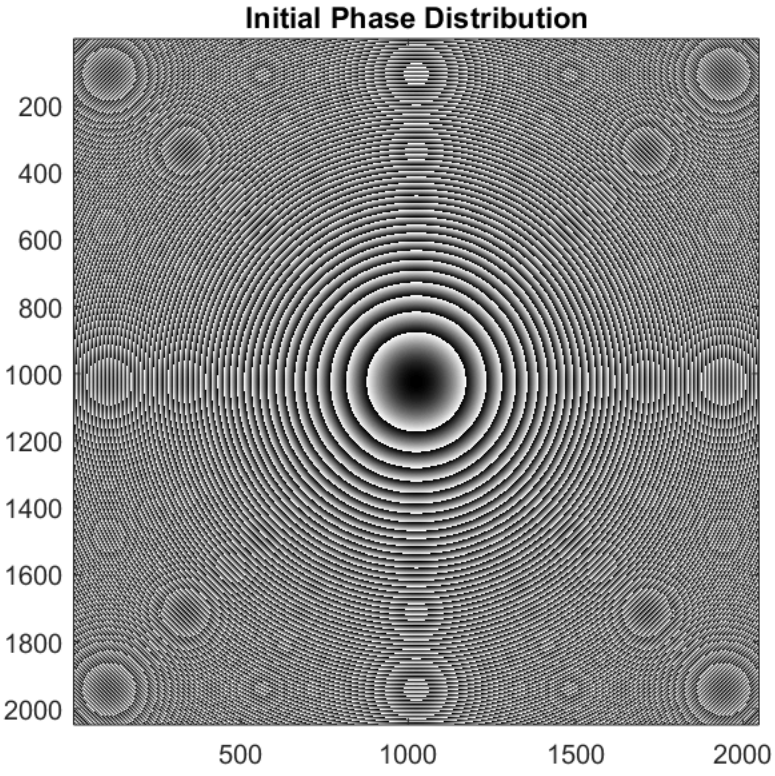
Target Image

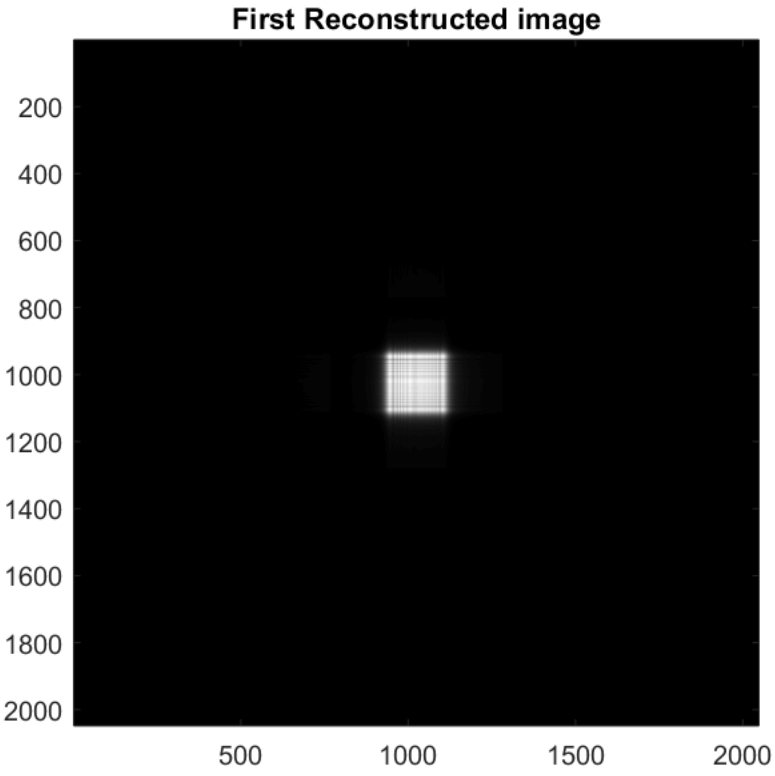
```
Target_Ori = rgb2gray(imread('square.bmp'));      %target image input
Target = double(Target_Ori);                    %changing the target
 into matrix of doubles with precision

 figure %Original Target Image
    imagesc(Target), axis image, colormap('gray');
    title('Target image')

SR_Ori = rgb2gray(imread('SR.bmp'));            %signal region input
threshold = 0.5;
SR = (double(SR_Ori) >= threshold);             %change signal region
 into matrix of 1 and 0
NR = 1-SR;                                      %noise region =
 outside of signal region
MR = SR;                                        %define measure region
 as the same with signal region
N_m = nnz(MR);
Target_m = MR.*Target;
```



Phase Initialization Step

```
alpha = aspect_ratio/(1+aspect_ratio);
A_quad = 4*R*(alpha*X.^2 + (1-alpha)*Y.^2);
A_lin = B_lin*(X*cos(mu) + Y*sin(mu));
A_con = B_con*sqrt(X.^2+Y.^2);
A = mod(A_quad + A_con + A_lin ,2*pi);
B = abs(input).*exp(1i*A);
C = fftshift(fft2(fftshift(B)));
figure %CGH Phase Distribution Result
     imagesc(abs(A)), axis image, colormap('gray');
     title('Initial Phase Distribution');
%
figure %First Reconstructed Image
     imagesc(abs(C)), axis image, colormap('gray');
     title('First Reconstructed image')
```


Initial Phase Distribution
200
400
600
800
1000
1200
1400
1600
1800
2000
500
1000
1500
2000


First Reconstructed image
200
400
600
800
1000
1200
1400
1600
1800
2000
500
1000
1500
2000

Perform MRAF (Mixed-Region Amplitude Freedom) Algorithm

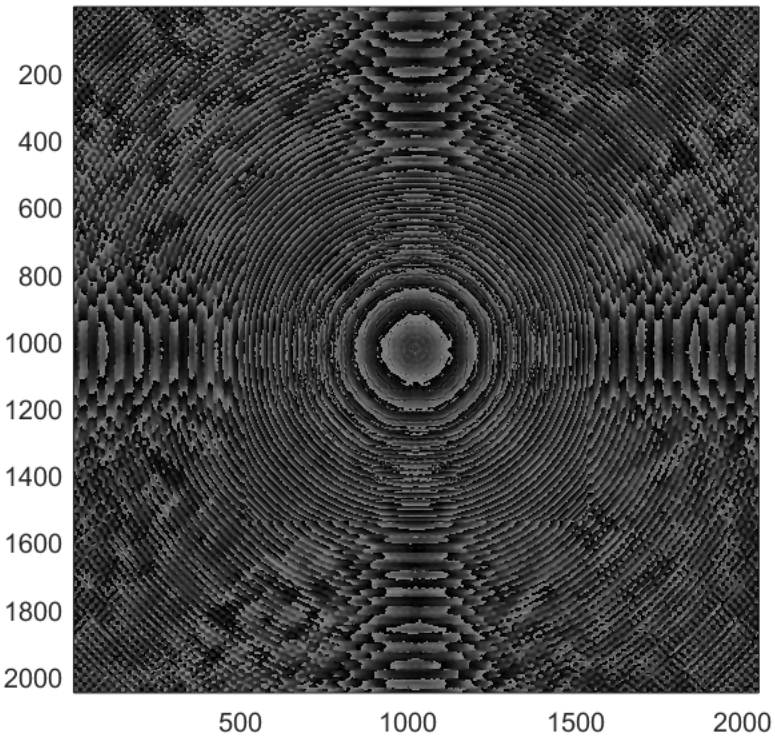
```
for i=1:i_num
    D = (m*abs(Target).*SR + (1-m)*abs(C).*NR).*exp(1i*angle(C)); %the
 main line of MRAF algorithm
    A = fftshift(ifft2(fftshift(D)));
    B = abs(input).*exp(1i*angle(A));
    C = fftshift(fft2(fftshift(B)));
    %Error Calculation: Root Mean Squre Error of Measure Region
        C_m = MR.*C;
        error_cur = sqrt((1/N_m)*sum(sum(((abs(C_m)./
max(max(abs(C_m)))) - abs(Target_m)./max(max(abs(Target_m)))).^2 ) ));
        error = [error; error_cur];
end
```

Calibration Section by Dae Gwang

```
Crop=angle(A)+pi;   % change range, from -pi to pi into 0 to 2pi
aaaa=136;           % maximum gray level for 2pi phase shift (depends
 on SLM)
Crop=floor(aaaa.*Crop./(2.0*pi)); % converting the phase shift into
 gray level

% This line changes the black into the white
for j=1:pady
    for k=1:padx
        if Crop(j,k) == 0;
            Crop(j,k)=aaaa;
        end
    end
end

% This line is the main calibration section, which we got after
 experiment
for qi=1:pady
    for qj=1:padx
        k= Crop(qi,qj);

 Crop(qi,qj)=round(0.863*k-0.00596*(k)^2+0.00008035*k^3-0.000000238*k^4+0.00000000
    end
end
```

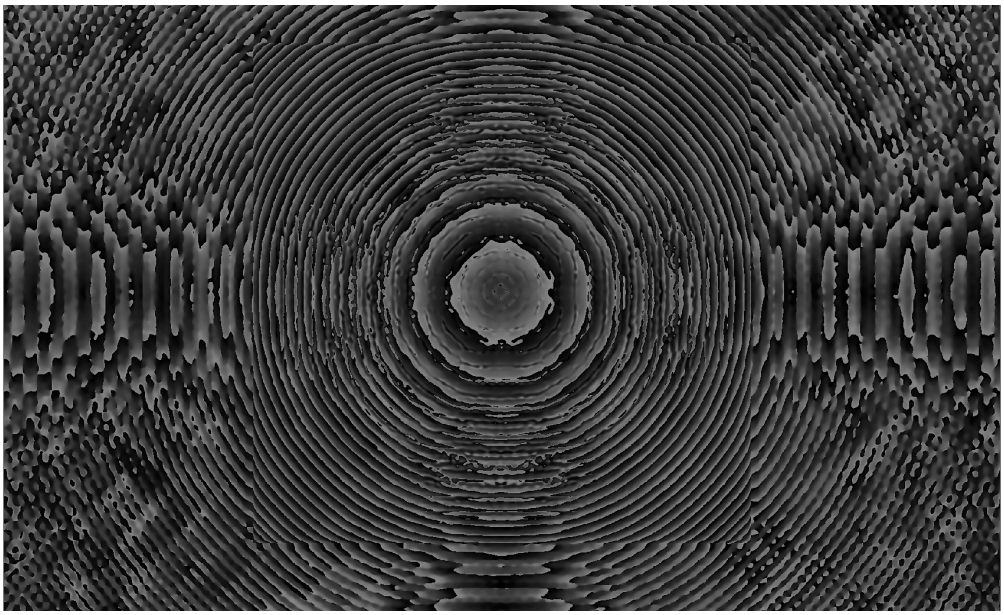
Figure of Calibrated CGH

```
Crop=Crop';

figure
imagesc(Crop), axis image ,colormap('gray'), caxis([0,255]);
```



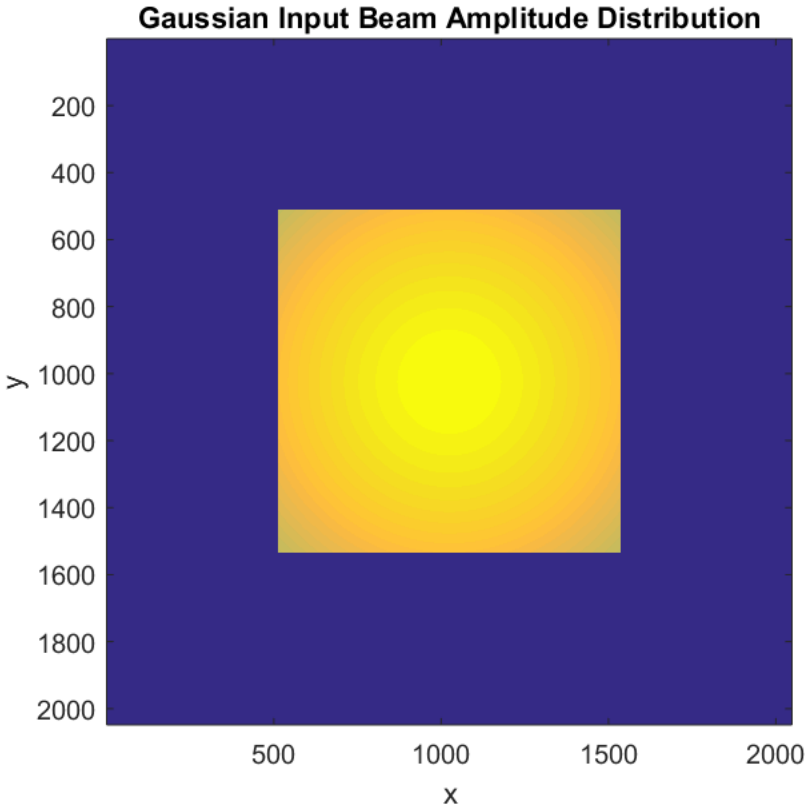
Setting the position, magnification, etc. of the CGH

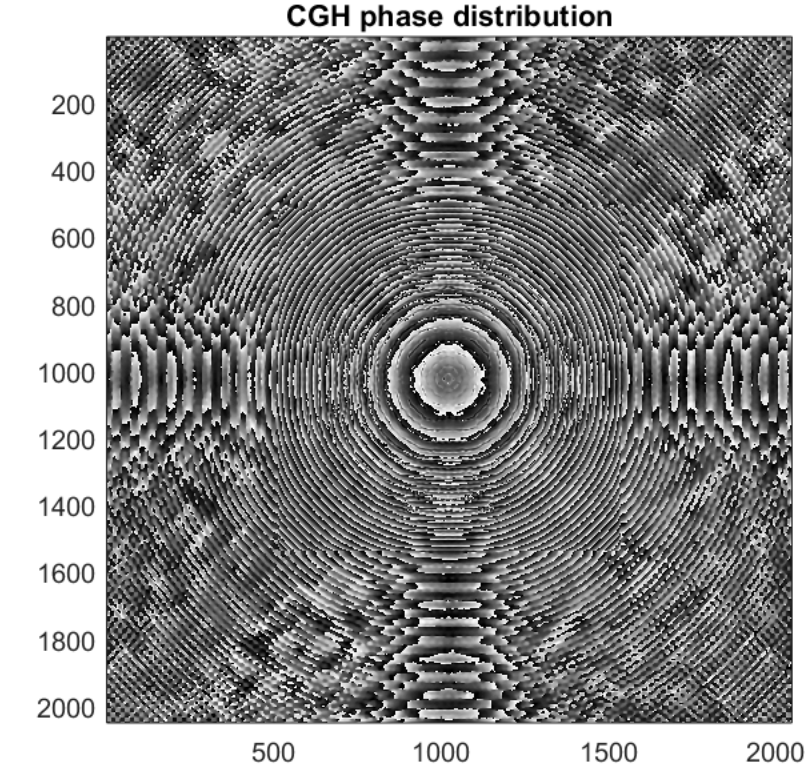
```
magni=1.65;

set(gcf, 'Units', 'Normalized', 'OuterPosition', [1 0.00 1 1]);
set(gca, 'Units', 'normalized','Position',[0. -0.3 1 magni]);
set(gcf, 'Toolbar', 'none', 'Menu', 'none','menubar','none','NumberTitle','off');
set(gca,'Yticklabel',[],'Xticklabel',[],'ytick',[],'xtick',[])
```

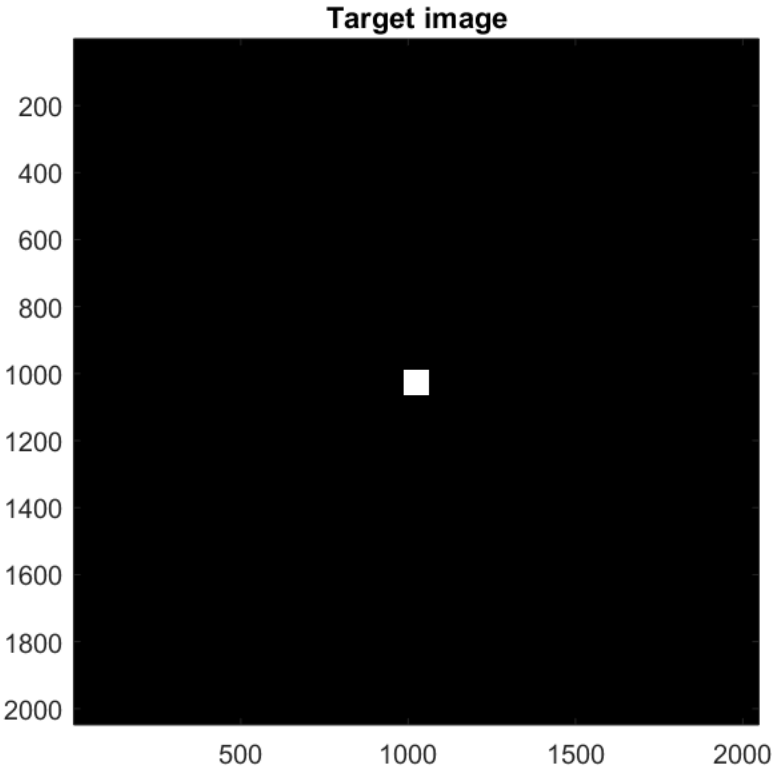


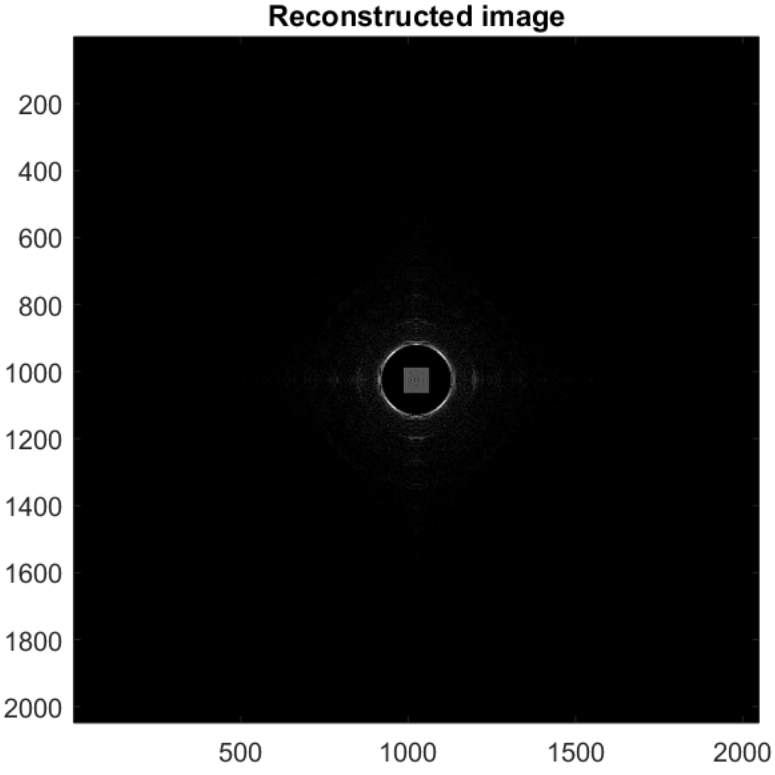
Show Result

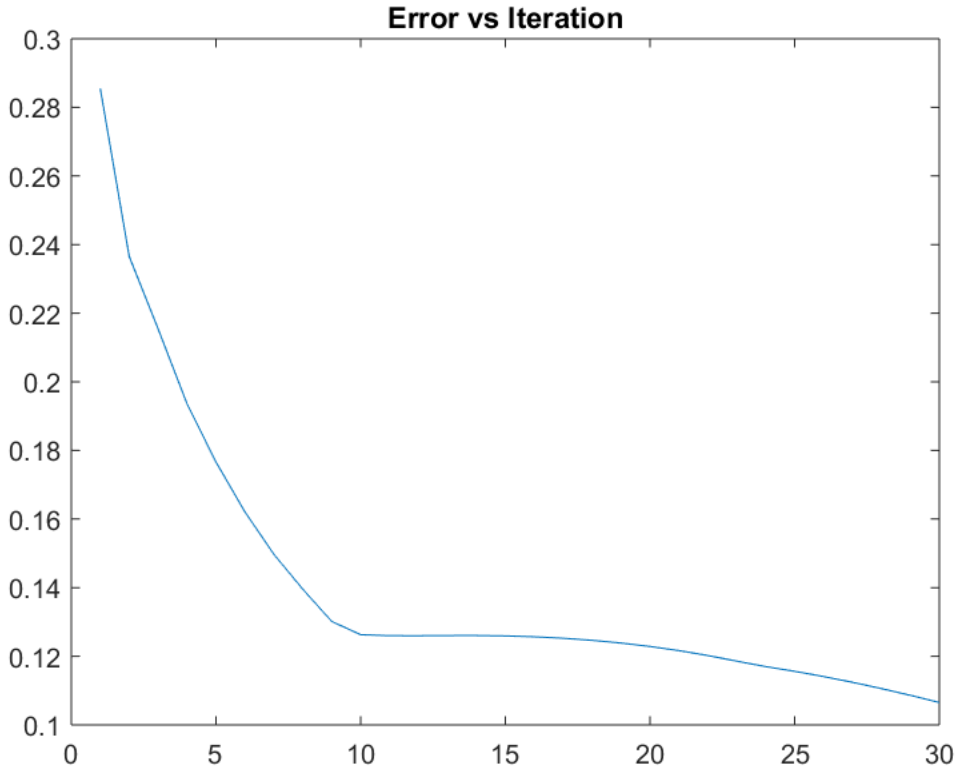
```
figure %Input Beam Distribution
    imagesc(input), axis image;
    title('Gaussian Input Beam Amplitude Distribution')
    xlabel('x')
    ylabel('y')

figure %CGH Phase Distribution Result
    imagesc(Crop), axis image, colormap('gray');
    title('CGH phase distribution');

figure %Original Target Image
    imagesc(Target./max(max(Target))),axis image, colormap('gray');
    title('Target image')

figure %Reconstructed Image
    imagesc(abs(C)./max(max(abs(C)))), axis image, colormap('gray');
    title('Reconstructed image');

figure %Error vs iteration
    i = 1:1:i;
    plot(i,(error'));
    title('Error vs Iteration');
```


Gaussian Input Beam Amplitude Distribution
200
400
600
800
1000
1200
1400
1600
1800
2000
y
500
1000
1500
2000
x


CGH phase distribution
200
400
600
800
1000
1200
1400
1600
1800
2000
500
1000
1500
2000


Target image
200
400
600
800
1000
1200
1400
1600
1800
2000
500
1000
1500
2000


Reconstructed image
200
400
600
800
1000
1200
1400
1600
1800
2000
500
1000
1500
2000



Published with MATLAB® R2016b