Image Processing and Computer Vision- MATLAB

UE19CS333

6th Semester, Academic Year 2021-22

Date: 13-03-2022

Team : ASHWA

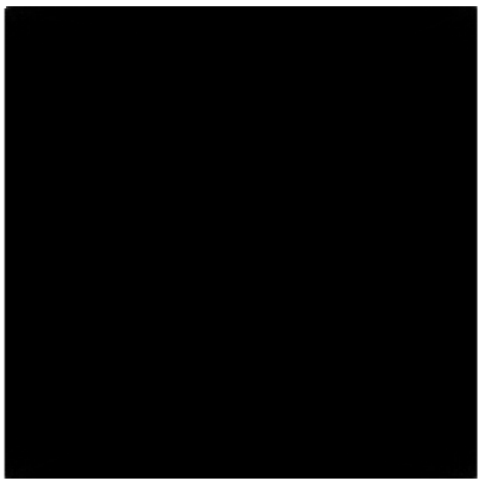
Name:	SRN:	Section:
LAKSHMI M A	PES1UG20CS813	I
NAGESH B S	PES1UG20CS816	C
ARCHIT RAJ	PES1UG19CS598	B
DEGALA GAGAN	PES1UG19CS135	C

ASSIGNMENT:4

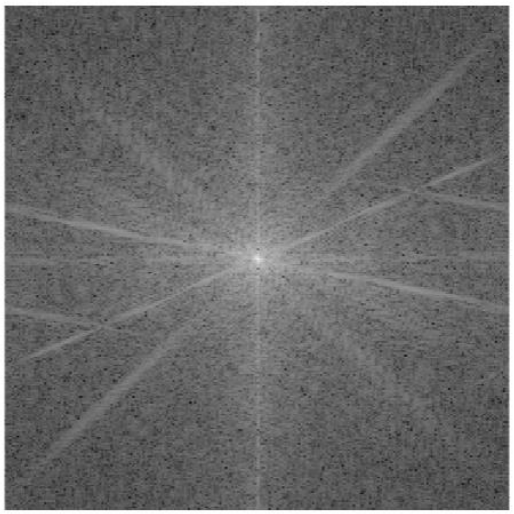
1. Read the cameraman image and display the Fourier Spectrum

```
m=imread('cameraman.tif');
CM = fft2(cm);
S = abs(CM); %same as (S=sqrt(real(CM).*real(CM))+(image(CM).*(imag(CM)))
figure; imshow(S,[]); FshiftCM = fftshift(CM); S2 = log(1+abs(FshiftCM));
figure; imshow(S2, []);
PhaseAngle = atan2(imag(FshiftCM),real(FshiftCM)); figure;
imshow(PhaseAngle,[]);
```

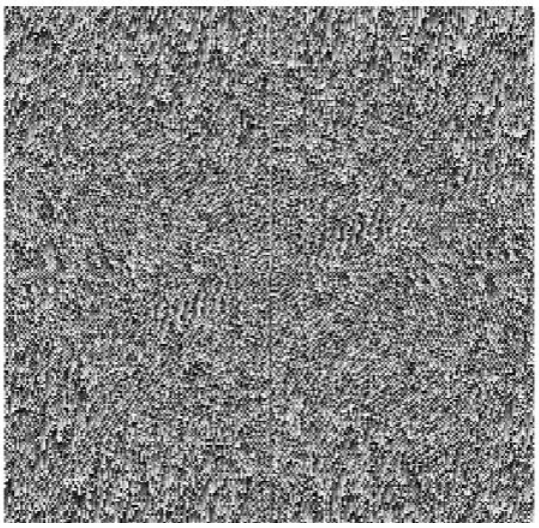
```
cm=imread('cameraman.tif');
CM = fft2(cm);
S = abs(CM); %same as (S=sqrt(real(CM).*real(CM))+(image(CM).*(imag(CM)))
figure; imshow(S,[]);
```



```
FshiftCM = fftshift(CM);
S2 = log(1+abs(FshiftCM));
figure; imshow(S2, []);
```



```
PhaseAngle = atan2(imag(FshiftCM),real(FshiftCM));
figure;
imshow(PhaseAngle,[]);
```



a. Create a rectangle of black and white strips and apply fft2 without the shift operation

```
p=zeros(256,256);
p(1:128,1:end)=255;
figure; imshow(uint8(p));
```

b. **Display the Fourier Spectrum of the strip image 'p'**
c. **Use fftshift and then display the Fourier Spectrum**
d. **Use fftshift and apply the log transform to display the Fourier Spectrum and be able to see something ☺ (use colormap gray if you use imagesc to display the image)**
e. **Paste the lines of code for 1b-1e and images here.**

```
CM = fft2(p);
>> S = abs(CM); %same as
(S=sqrt(real(CM).*real(CM))+(image(CM).*(imag(CM)))
>> figure; imshow(S,[]);
>> FshiftCM = fftshift(CM);
>> S2 = log(1+abs(FshiftCM));
>> figure; imshow(S2, []);
>> PhaseAngle = atan2(imag(FshiftCM),real(FshiftCM));
>> figure; imshow(PhaseAngle,[]);
```

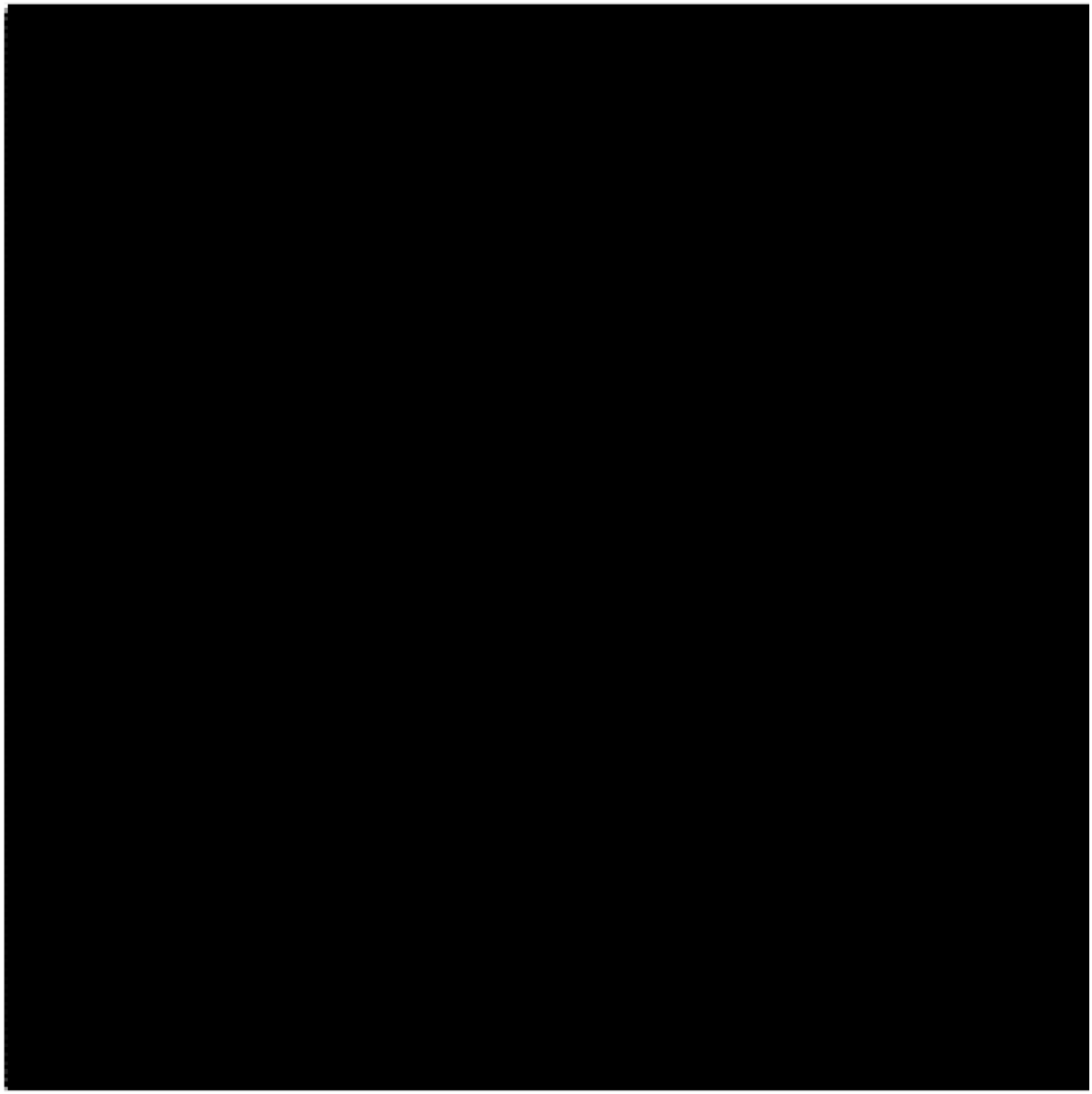


Figure:1

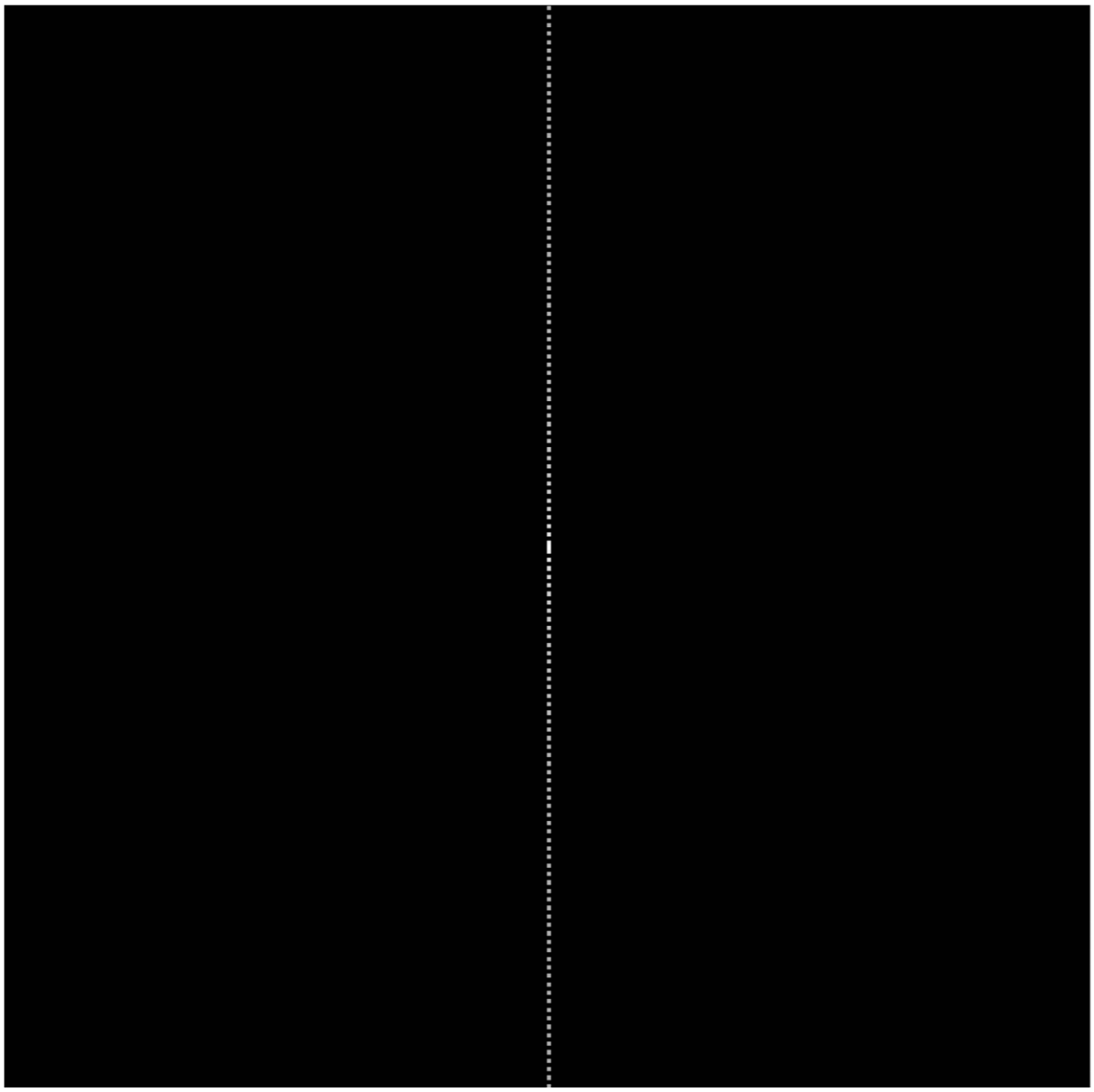


Figure: 2

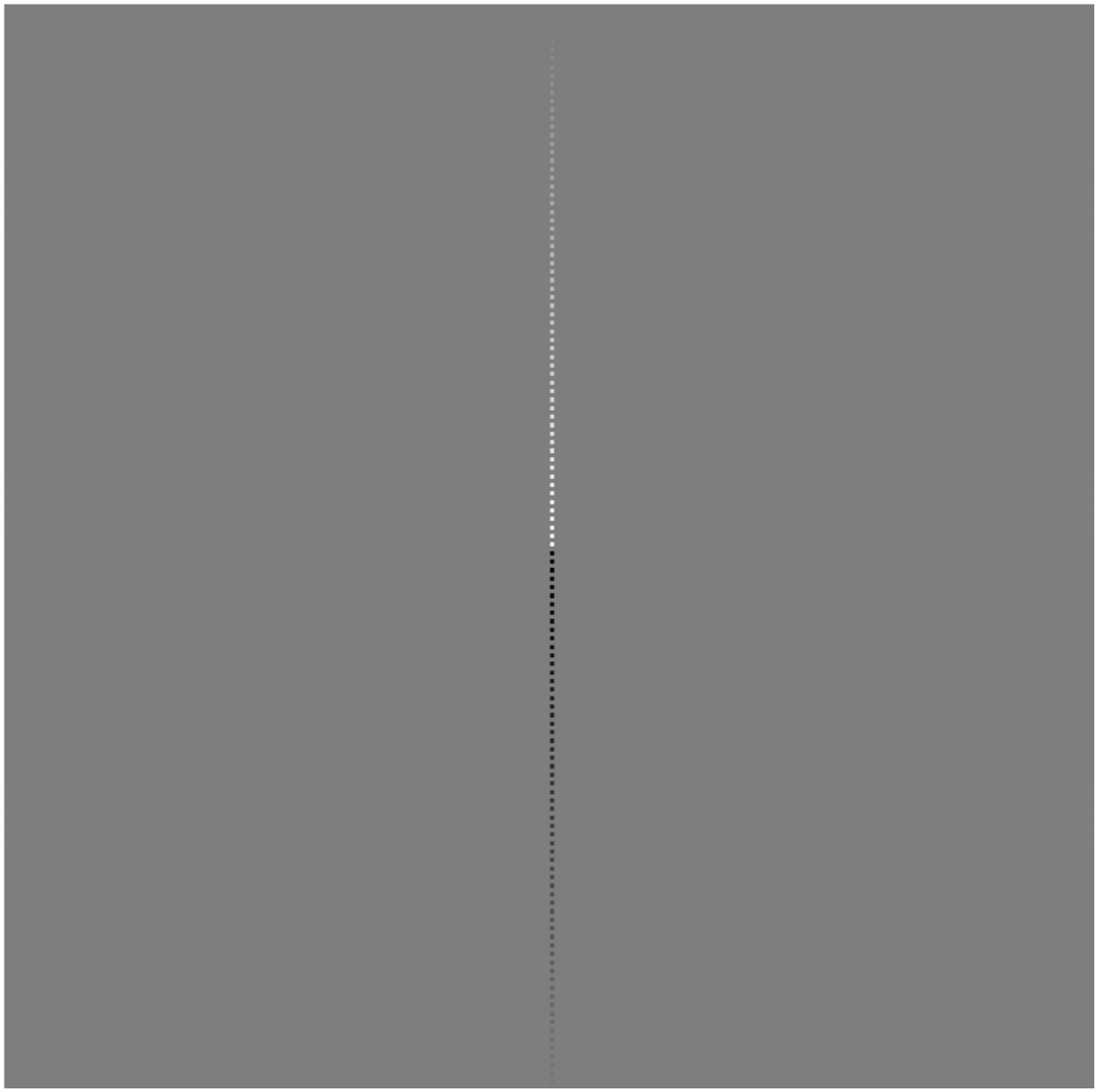


Figure:3

a. **Find the inverse Fourier transform of cameraman after fftshift; obtain the original image using ifftshift. Paste the code and images here.**

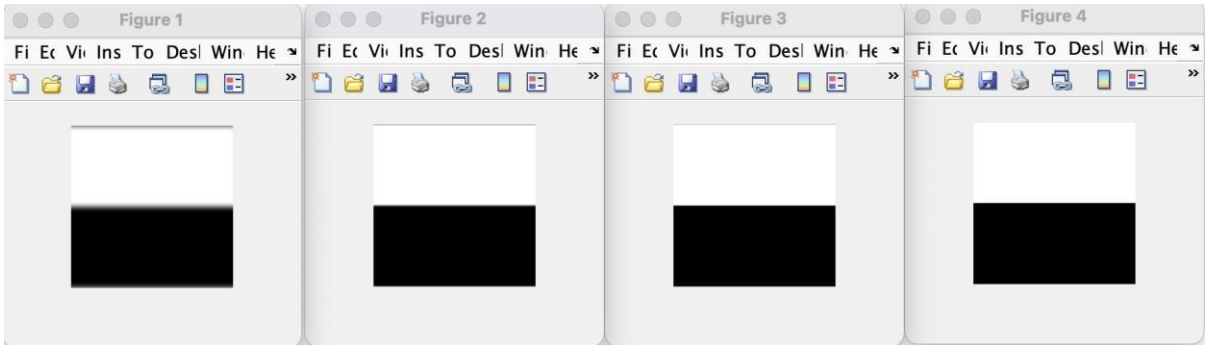
```
p = imread('cameraman.tif');
CM = fft2(p);
S = abs(CM);
FshiftCM = fftshift(CM); inv =
ifft2(FshiftCM); inv =
abs(inv); figure;
imshow(uint8(real(inv)));
m = ifftshift(inv);
figure; imshow(uint8(real(im)));
```

2. **Filter this using a Gaussian filter with different sigma; let's start with sig=10.**

```
[M,N] = size(p); P =
 fft2(double(p)); sig
 = 10;
H = lpfilter('gaussian',M,N,sig); %hpfilter can be used for high pass
filters with 'ideal' or 'btw' or 'gaussian'
figure; imshow(H);
G = H.*P; %filtering in the frequency domain without fftshift to
facilitate visual interpretation
g = ifft2(G); %returning to the space domain
g = uint8(g);%this is the filtered image
```

a. **Apply the Gaussian filter using sig = 30, 60, 100; as the variance of the Gaussian increases, what happens to the black-white interface in p? Paste the lines of code and images here.**

```
[M,N] = size(p);
P = fft2(double(p));
sig = 10;
H = lpfilter('gaussian',M,N,sig);
G = H.*P;
 = ifft2(G);
 = uint8(g);
figure; imshow(g);
>> sig = 30;
H = lpfilter('gaussian',M,N,sig);
 = H.*P;
 = ifft2(G);
g = uint8(g);
figure; imshow(g);
>> sig = 60;
H = lpfilter('gaussian',M,N,sig);
G = H.*P;
 = ifft2(G);
g = uint8(g); figure; imshow(g);
>> sig = 100;
H = lpfilter('gaussian',M,N,sig);
 = H.*P;
g = ifft2(G);
g = uint8(g);
 figure; imshow(g);
```

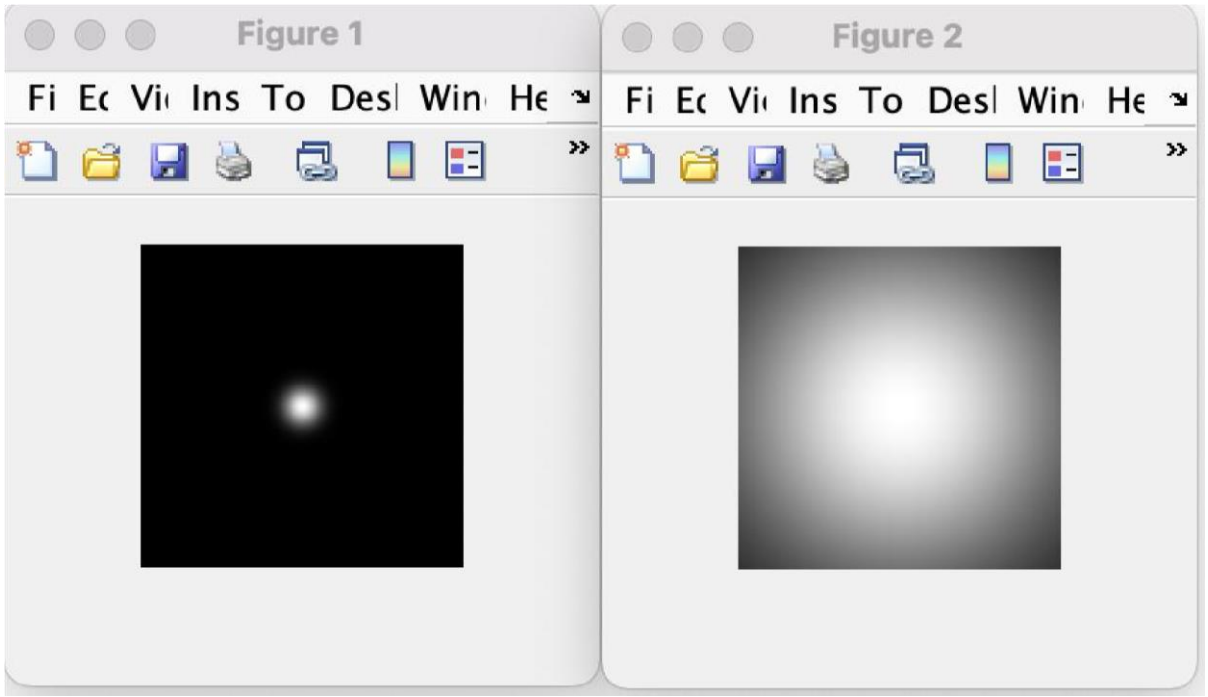


sigma = 10 sigma = 30 sigma = 60 sigma = 100

The black white interface gets more and more sharper as sigma increases.

b. **Use fftshift to visualize the spectrum of the Gaussian sig = 10 versus sig =100 (with proper centering) and also display the corresponding images in the spatial domain.**

```
sig = 10;
H = lpfilter('gaussian',M,N,sig); h1 = fftshift(H); imshow(h1);
sig = 100;
H = lpfilter('gaussian',M,N,sig); h1 = fftshift(H); figure; imshow(h1);
```



Sigma=10 **Sigma=100**

c. **Define a low pass (lpfilter) and an ideal high pass filter (hpfilter) and apply these on the cameraman image. Do you see the ringing effect from the 2D sinc function?**

```
    p = imread('cameraman.tif');
    [M,N] = size(p);
    P = fft2(double(p));
    D0 = 50;
    Hlow = lpfilter('ideal', M, N, D0);
    G=Hlow.*P;
    g = ifft2(G);g = uint8(g);
figure;imshow(g);
```

Ideal lowpass filtered image

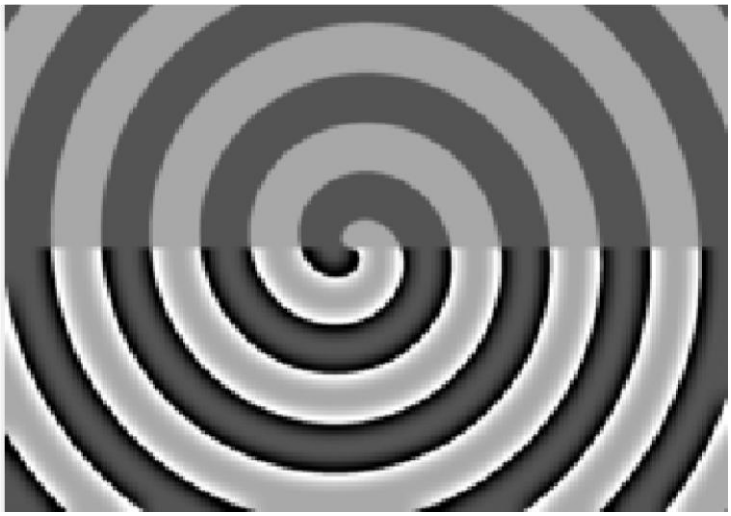
We can clearly see the ringing effect throughout the smoothened image. This is mainly because of the 2D sinc function.

```
p = imread('cameraman.tif');
[M,N] = size(p);
P = fft2(double(p));
D0 = 10;
Hhigh = hpfilter('ideal', M, N, D0);
G=Hhigh.*P;
g = ifft2(G);g = uint8(g);
figure;imshow(g);
```

Ideal highpass filtered image

d. **Use unsharp masking and/or high boost filtering on the spiral image (called UnsharpMasking_Spiral.png) to 'fix' the blur in the top half of the image.**

```
p = imread('UnsharpMasking_Spiral.png');
figure;imshow(p);
[M,N] = size(p);
P = fft2(double(p));
H = lpfilter('gaussian',M,N,10);
G=H.*P;
g = ifft2(G);g = uint8(g);
figure;imshow(g);
figure;imshow(p);
figure;imshow(g);
sharp = p-g;
figure;imshow(sharp);
k=6;
result = p+k*sharp;
figure;imshow(result);
```



Original Image

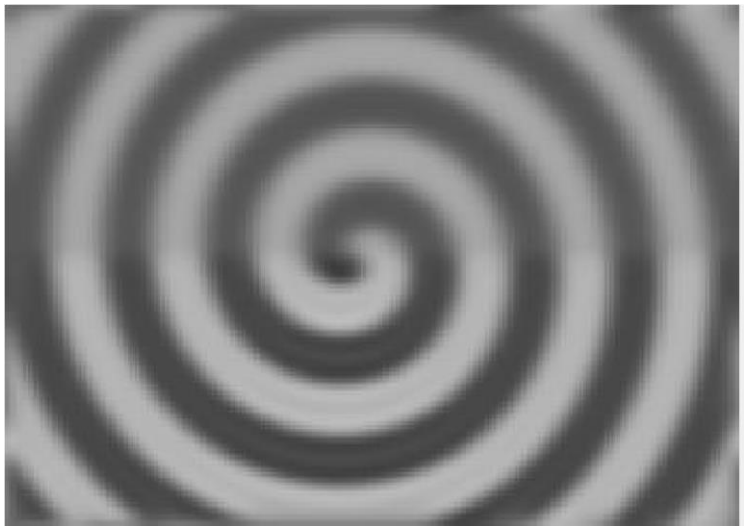


Image Smoothened using gaussian lowpass filter

Difference between original image and it's smoothened version

Final Unsharp Masked Image

We can see that the blur in the top part of the image has almost disappeared.