Computational Physics Assignement 7

Dara Corr - ID: 18483836

This assignment investigates using Fourier Transforms in Imaging Systems and Image Processing. The Theory of Convolution is used to apply and remove blur from images. Convolution involves a convolution integral in the Spatial domain

$$I(x) = \int O(x)P(x - x')dx'$$

where P(x) is the Point Spread Function, I(x) is the output image and O(x) is any input object.

This process is simplified in the Frequency domain (Fourier domain):

$$FTI(f) = FTO(f) \cdot FTP(f)$$

where FTI is the Fourier Transform of the Image, FTO is the Fourier Transform of the Object and FTP is the Fourier transform of the point spread function.

The first part of the assignment looks at image blurring using Fourier Analysis and a Gaussian blur. The first step is to make an image with a simple shape on it using the PIL and pylab libraries.The next step is to blur the image. This is done by taking the 2D Discrete Fourier transform of the image, applying the Gaussian filter to the image, applying the inverse transform to the image and finally displaying the blurred image. The formula for the Gaussian blur is as follows:

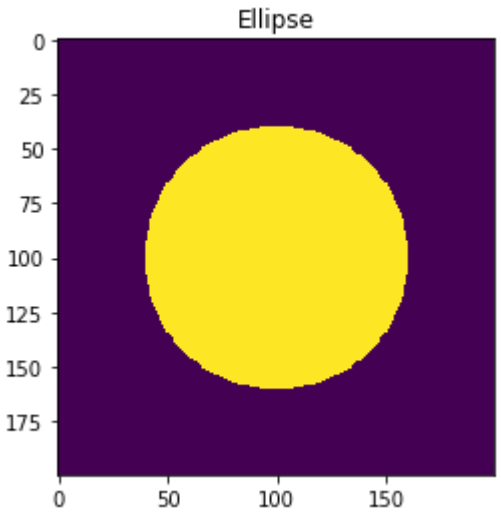
$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{\frac{(x^2+y^2)}{2\sigma^2}}$$

where x is the distance from the origin on the horizontal axis, y is the distance from the origin on the vertical axis and σ is the standard deviation of the Gaussian Distribution.

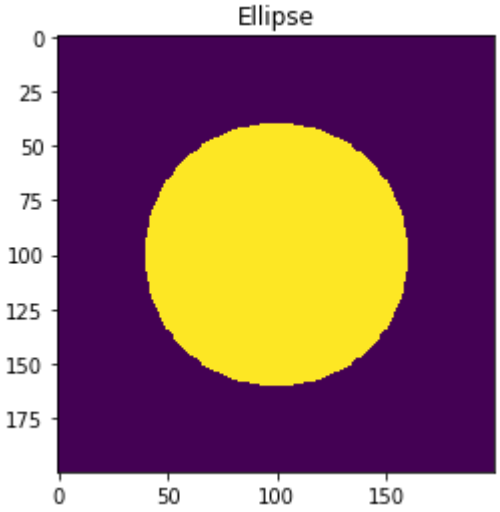
The second part of the assignment consists of creating the Magnitude Spectrum of an image. We are given an image of a cat to display its Magnitude Spectrum. To do this, compute the 2-dimensional discrete Fourier Transfrom of the Image, Shift the zero-frequency component to the centre of the image using the fftshift() function in numpy. Take the log of the spectrum and evaluate the absolute value of values in the spectrum to give the Magnitude Spectrum. The final result will be the output of the Magnitude Spectrum displayed to the user.

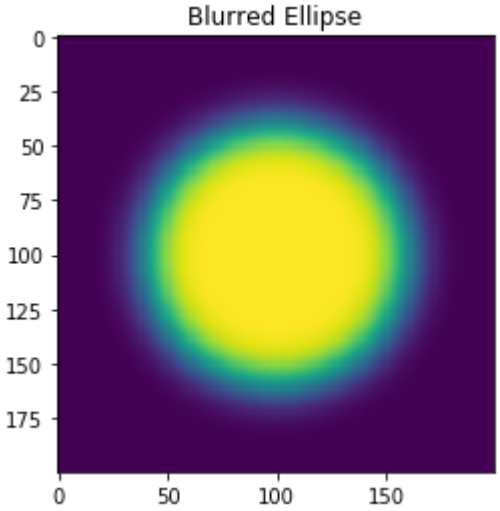
Finally, we are tasked with finding the original image from the output of the Magnitude Spectrum in the previous task. To do this I took the Spectrum I found in task 4, inverted the fftshift and used the Inverse Fourier Transform. This resulted in the original image of the cat that was used in the previous step, which was obtained by reversing the procedures done in the previous task.

In [5]:
```
#task 1
import pylab
import matplotlib.pyplot as plt
import numpy as np
from PIL import Image
from PIL import ImageDraw

w,h = 200,200 #set size of square (width 'w' and height 'h') for image to be drawn within

image = Image.new('I', (w,h)) #create a new instance of an image (the background)
draw = ImageDraw.Draw(image) #draw the image
draw.ellipse((40, 40, 160, 160), fill = 'white', outline ='white') #draw ellipse on the image
pylab.imshow(image) #show the image
pylab.title('Ellipse') #give title ellipse
pylab.show() #display ellipse
```

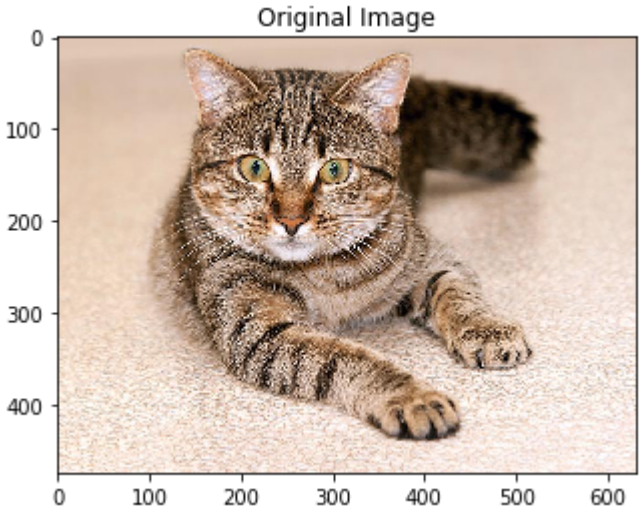


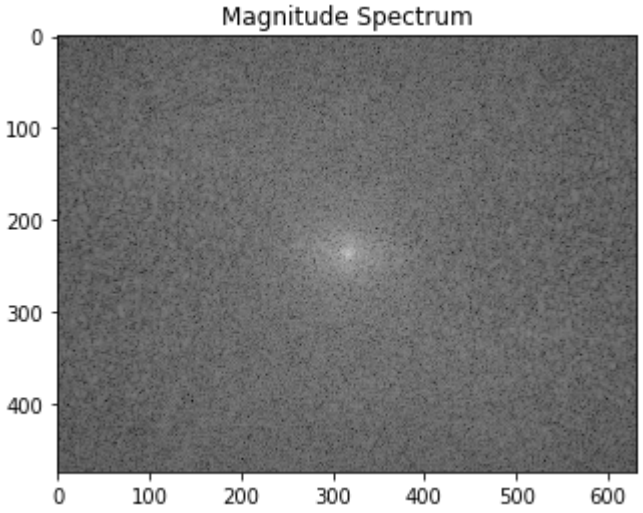
In [84]:
```
#task 2
w,h = 200,200

image = Image.new('I', (w,h))#draw first image
draw = ImageDraw.Draw(image)
draw.ellipse((40, 40, 160, 160), fill = 'white', outline ='white')
pylab.imshow(image)
pylab.title('Ellipse')
pylab.show()


image2 = Image.new('I', (w,h))#create second image
draw2 = ImageDraw.Draw(image2)
draw2.ellipse((40, 40, 160, 160), fill = 'white', outline ='white')


def FT_gaussblur(image,Sigma): #task 3

    from scipy import ndimage #necessary libraries and functions are imported
    import matplotlib.pyplot as plt
    import pylab
    import numpy.fft

    input_ = numpy.fft.fft2(image2) #define image to be blurred as the input a
nd apply fast fourier transform
    result = ndimage.fourier_gaussian(input_, sigma = Sigma) #result is blurre
d
    result = numpy.fft.ifft2(result) #result is inverse transformed to give us
a blurred image

    pylab.imshow(result.real)#image is displayed (only real part since FFT giv
es complex result)
    pylab.title("Blurred Ellipse")
    pylab.show()


FT_gaussblur(image2, 10)#call function to display blurred image
```


Ellipse
0
25
50
75
100
125
150
175
0
50
100
150

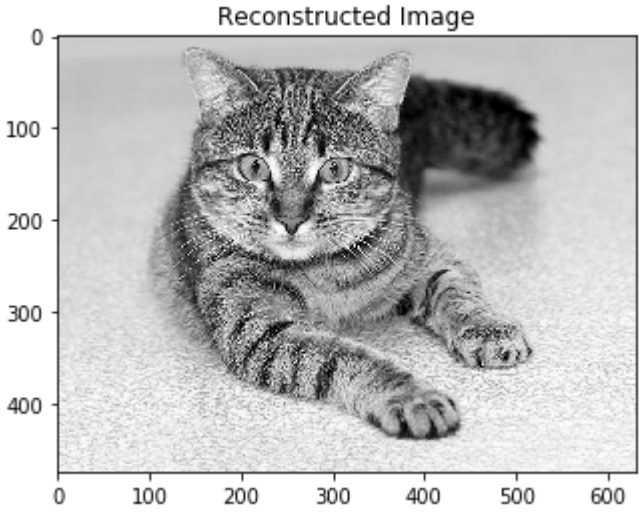

Blurred Ellipse
0
25
50
75
100
125
150
175
0
50
100
150

In [71]:
```
#task4
import numpy.fft
import matplotlib.pyplot as plt

cat  = Image.open("cat.jpg") #open image of cat and save as cat
plt.imshow(cat)
plt.title("Original Image")#plot image and give it a title
plt.show()

cat = cat.convert("L") #makes cat image greyscale

f = np.fft.fft2(cat) #get fourier transform of image
fshift = np.fft.fftshift(f) #fourier shift image
spectrum = (fshift) #define image spectrum
magnitude_spectrum = (np.abs(fshift)) #get absolute values to get magnitude sp
ectrum

plt.imshow(20*np.log(magnitude_spectrum), cmap = 'gray')#get log of magnitude
 spectrum and plot it
plt.title("Magnitude Spectrum")# give plot a title and display it
plt.show()
```





In [82]:
```
#task 5

#reverse process of getting the image spectrum in the last task to reconstruct
original image

inverse_shift = np.fft.ifftshift((spectrum)) #inverse the fft shift
img_back = np.fft.ifft2(inverse_shift) # invert the fourier transform
img_back = np.abs(img_back) # get absolute value to display image

plt.imshow(img_back, cmap = 'gray') #display reconstructed image with cmap app
lied
plt.title("Reconstructed Image")
plt.show()
```



The Reconstructed Image looks the same as the original image with the gray CMAP applied. If you remove the CMAP, the image's colour is a combination of greens and yellows instead of the colours seen in the original image.