CS-663 Assignment 5 Q3

Soham Naha (193079003)
Akshay Bajpai (193079002)
Mohit Agarwala (19307R004)

November 15, 2020

3

Consider the image with the low frequency noise pattern shared in the homework folder in the form of a .mat file. Your task is to (a) write MATLAB code to display the log magnitude of its Fourier transform, (b) to determine the frequency of the noise pattern by observing the log magnitude of the Fourier transform and guessing the interfering frequencies, and (c) to design and implement (in MATLAB) an ideal notch filter to remove the interference(s) and display the restored image. To this end, you may use the fft2, ifft2, fftshift and ifftshift routines in MATLAB. [15 points]

Solution:

```python
import scipy.io
import cv2
import numpy as np
import sys,os
import matplotlib.pyplot as plt

def get_image(filename):
    """Extract the image from mat file
    inputs: filename(filepath of the .mat file)
    outputs: the image as numpy array after normalizing
    """
    f = scipy.io.loadmat(filename)
    out = np.array(f['Z'])
    return normalize_image(out)

def normalize_image(image):
    """Normalize the image to remain between 0-1
    Take the image and use the Min-Max normalization criterion to normalize the images

    inputs: image(input image to be normalized)
    outputs: normalized(normalized image)
    """
    out = image.copy()
    normalized = (out-np.min(out))/(np.max(out)-np.min(out))
    return normalized


def notch(f_img):

    r1=247
    c1=251

    r2=11
    c2=6

    r3= 0
    c3= 0

    D0=4
    for i in range(-D0+1,D0):
```

```
            for j in range (-D0+1,D0):
                f_img[r1+i][c1+j]=0
                f_img[r2+i][c2+j]=0


    f_img[r1-D0][c1]=0
    f_img[r1+D0][c1]=0
    f_img[r1][c1-D0]=0
    f_img[r1][c1+D0]=0

    f_img[r2-D0][c2]=0
    f_img[r2+D0][c2]=0
    f_img[r2][c2-D0]=0
    f_img[r2][c2+D0]=0

    f_img[r3+1][c3+1]=0
    f_img[r3-1][c3-1]=0
    f_img[r3+1][c3-1]=0
    f_img[r3-1][c3+1]=0

    f_img[r3-D0][c3]=0
    f_img[r3+D0][c3]=0
    f_img[r3][c3-D0]=0
    f_img[r3][c3+D0]=0

    return f_img


if __name__=="__main__":

    filename="../data/image_low_frequency_noise.mat"
    img = get_image(filename)

    #input_image
    f_img = np.fft.fft2(img)
    fshift_img = np.fft.fftshift(f_img)
    input_magnitude_spectrum = np.log(1+np.abs(fshift_img))

    plt.figure()
    plt.subplot(121)
    plt.tight_layout()
    plt.imshow(img, cmap = 'gray')
    plt.colorbar(aspect=5, shrink=0.5)
    plt.title('Input Image')
    plt.subplot(122)
    plt.tight_layout()
    plt.imshow(input_magnitude_spectrum, cmap = 'inferno')
    plt.colorbar(aspect=5, shrink=0.5)
    plt.title('Magnitude Spectrum of Input Image')
    plt.savefig("../images/InputImageAndMagnitudeSpectrum.png", bbox_inches="tight")

    r,c = img.shape
    x = np.array([i for i in range(r)])
    y = np.array([i for i in range(c)])
    X,Y = np.meshgrid(x,y)

    fig = plt.figure()
    ax = plt.axes(projection ='3d')
    plt.set_cmap("inferno")
    surf = ax.plot_surface(X, Y, input_magnitude_spectrum, cmap="inferno")
    plt.title('Magnitude Plot of Input image')
    fig.colorbar(surf, ax=ax)
    plt.savefig("../images/InputImageMagnitudeSpectrumPlot.png",bbox_inches="tight")
```

```
    #Restored_image
    f_img = notch(f_img)
    restored_image = np.fft.ifft2(f_img)
    restored_image = np.abs(restored_image, dtype=float)

    plt.figure()
    plt.subplot(121)
    plt.tight_layout()
    plt.imshow(img, cmap = 'gray')
    plt.colorbar(aspect=5, shrink=0.5)
    plt.title('Input Image')
    plt.subplot(122)
    plt.tight_layout()
    plt.imshow(restored_image, cmap = 'gray')
    plt.colorbar(aspect=5, shrink=0.5)
    plt.title('Restored Image')
    plt.savefig("../images/InputImageAndRestoredImage.png", bbox_inches="tight",cmap="gray")

    f_restored_image = np.fft.fft2(restored_image)
    fshift_restored_image = np.fft.fftshift(f_restored_image)
    restored_magnitude_spectrum = np.log(1+np.abs(fshift_restored_image))
    plt.figure()
    plt.subplot(121)
    plt.tight_layout()
    plt.imshow(restored_image, cmap = 'gray')
    plt.colorbar(aspect=5,shrink=0.5)
    plt.title('Restored Image')
    plt.subplot(122)
    plt.tight_layout()
    plt.imshow(restored_magnitude_spectrum, cmap = 'inferno')
    plt.colorbar(aspect=5, shrink=0.5)
    plt.title('Magnitude Spectrum of Restored Image')
    plt.savefig("../images/RestoredImageAndMagnitudeSpectrum.png", bbox_inches="tight")

    fig = plt.figure()
    ax = plt.axes(projection ='3d')
    plt.set_cmap("inferno")
    surf = ax.plot_surface(X, Y, restored_magnitude_spectrum, cmap="inferno")
    plt.title('Magnitude Plot of Restored image')
    fig.colorbar(surf, ax=ax)
    plt.savefig("../images/RestoredImageMagnitudeSpectrumPlot.png",bbox_inches="tight")
```

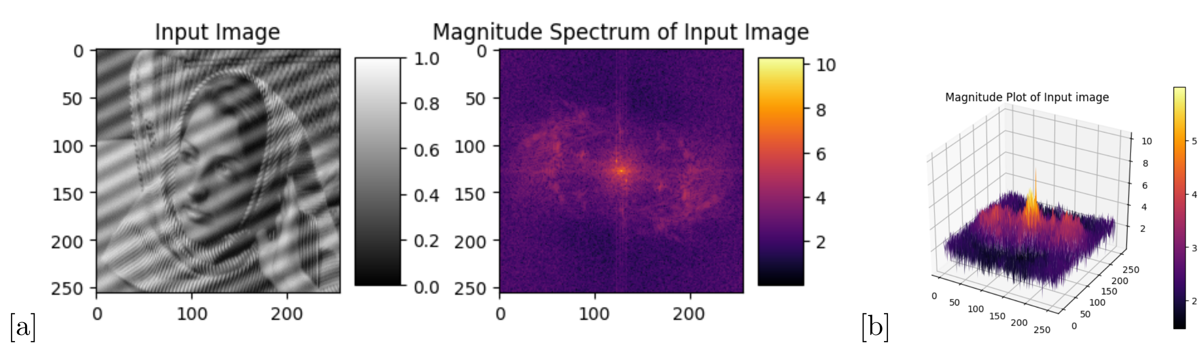

Figure 1: (a) Input Image (b) Magnitude Spectrum of input image

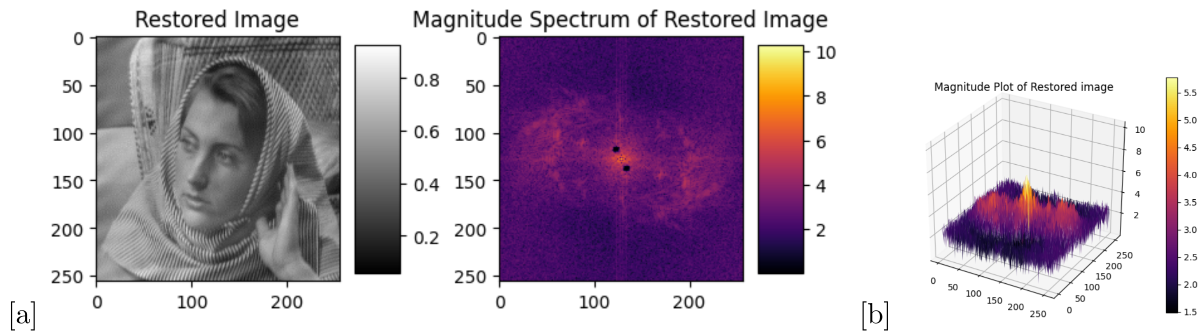

Figure 2: ((a) Restored Image (b) Magnitude Spectrum of restored image

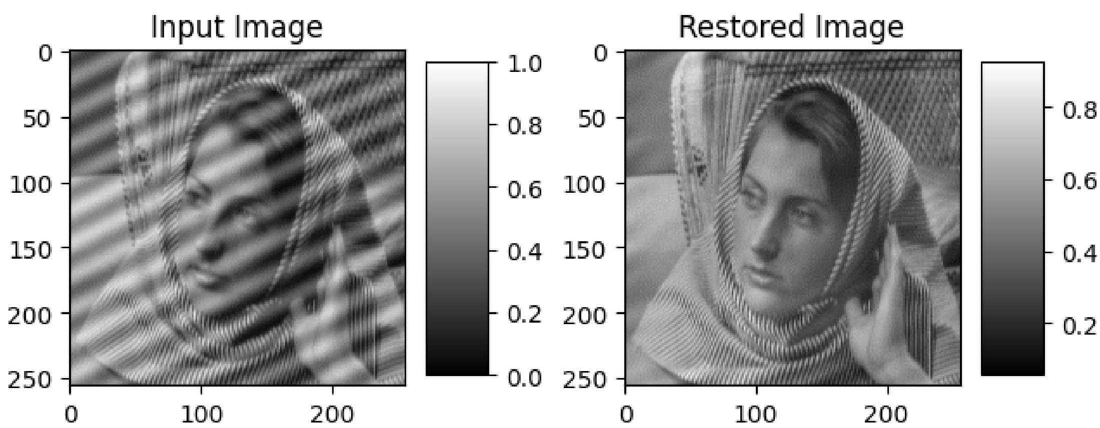

Figure 3: Comparison of Input and Restored Images

Discussion :
The Fourier Transform of the input image gave high values at $(u, v) = (247, 251), (11, 6)$. We construct a notch filter to pass regions at these locations, to remove low frequency noise. The restored image has the line noise remove to some extent from the input image.