

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:

```
1  import scipy.io
2  import cv2
3  import numpy as np
4  import sys,os
5  import matplotlib.pyplot as plt
6
7  def get_image(filename):
8      """Extract the image from mat file
9      inputs: filename(filepath of the .mat file)
10     outputs: the image as numpy array after normalizing
11     """
12     f = scipy.io.loadmat(filename)
13     out = np.array(f['Z'])
14     return normalize_image(out)
15
16 def normalize_image(image):
17     """Normalize the image to remain between 0-1
18     Take the image and use the Min-Max normalization criterion to normalize the images
19
20     inputs: image(input image to be normalized)
21     outputs: normalized(normalized image)
22     """
23     out = image.copy()
24     normalized = (out-np.min(out))/(np.max(out)-np.min(out))
25     return normalized
26
27
28 def notch(f_img):
29
30     r1=247
31     c1=251
32
33     r2=11
34     c2=6
35
36     r3= 0
37     c3= 0
38
39     D0=4
40     for i in range(-D0+1,D0):
```

```

41         for j in range (-D0+1,D0):
42             f_img[r1+i][c1+j]=0
43             f_img[r2+i][c2+j]=0
44
45
46         f_img[r1-D0][c1]=0
47         f_img[r1+D0][c1]=0
48         f_img[r1][c1-D0]=0
49         f_img[r1][c1+D0]=0
50
51         f_img[r2-D0][c2]=0
52         f_img[r2+D0][c2]=0
53         f_img[r2][c2-D0]=0
54         f_img[r2][c2+D0]=0
55
56         f_img[r3+1][c3+1]=0
57         f_img[r3-1][c3-1]=0
58         f_img[r3+1][c3-1]=0
59         f_img[r3-1][c3+1]=0
60
61         f_img[r3-D0][c3]=0
62         f_img[r3+D0][c3]=0
63         f_img[r3][c3-D0]=0
64         f_img[r3][c3+D0]=0
65
66     return f_img
67
68
69 if __name__=="__main__":
70
71     filename="../data/image_low_frequency_noise.mat"
72     img = get_image(filename)
73
74     #input_image
75     f_img = np.fft.fft2(img)
76     fshift_img = np.fft.fftshift(f_img)
77     input_magnitude_spectrum = np.log(1+np.abs(fshift_img))
78
79     plt.figure()
80     plt.subplot(121)
81     plt.tight_layout()
82     plt.imshow(img, cmap = 'gray')
83     plt.colorbar(aspect=5, shrink=0.5)
84     plt.title('Input Image')
85     plt.subplot(122)
86     plt.tight_layout()
87     plt.imshow(input_magnitude_spectrum, cmap = 'inferno')
88     plt.colorbar(aspect=5, shrink=0.5)
89     plt.title('Magnitude Spectrum of Input Image')
90     plt.savefig("../images/InputImageAndMagnitudeSpectrum.png", bbox_inches="tight")
91
92     r,c = img.shape
93     x = np.array([i for i in range(r)])
94     y = np.array([i for i in range(c)])
95     X,Y = np.meshgrid(x,y)
96
97     fig = plt.figure()
98     ax = plt.axes(projection = '3d')
99     plt.set_cmap("inferno")
100     surf = ax.plot_surface(X, Y, input_magnitude_spectrum, cmap="inferno")
101     plt.title('Magnitude Plot of Input image')
102     fig.colorbar(surf, ax=ax)
103     plt.savefig("../images/InputImageMagnitudeSpectrumPlot.png",bbox_inches="tight")
104
105

```

```

106 #Restored_image
107 f_img = notch(f_img)
108 restored_image = np.fft.ifft2(f_img)
109 restored_image = np.abs(restored_image, dtype=float)
110
111 plt.figure()
112 plt.subplot(121)
113 plt.tight_layout()
114 plt.imshow(img, cmap = 'gray')
115 plt.colorbar(aspect=5, shrink=0.5)
116 plt.title('Input Image')
117 plt.subplot(122)
118 plt.tight_layout()
119 plt.imshow(restored_image, cmap = 'gray')
120 plt.colorbar(aspect=5, shrink=0.5)
121 plt.title('Restored Image')
122 plt.savefig("../images/InputImageAndRestoredImage.png", bbox_inches="tight", cmap="gray")
123
124 f_restored_image = np.fft.fft2(restored_image)
125 fshift_restored_image = np.fft.fftshift(f_restored_image)
126 restored_magnitude_spectrum = np.log(1+np.abs(fshift_restored_image))
127 plt.figure()
128 plt.subplot(121)
129 plt.tight_layout()
130 plt.imshow(restored_image, cmap = 'gray')
131 plt.colorbar(aspect=5, shrink=0.5)
132 plt.title('Restored Image')
133 plt.subplot(122)
134 plt.tight_layout()
135 plt.imshow(restored_magnitude_spectrum, cmap = 'inferno')
136 plt.colorbar(aspect=5, shrink=0.5)
137 plt.title('Magnitude Spectrum of Restored Image')
138 plt.savefig("../images/RestoredImageAndMagnitudeSpectrum.png", bbox_inches="tight")
139
140 fig = plt.figure()
141 ax = plt.axes(projection = '3d')
142 plt.set_cmap("inferno")
143 surf = ax.plot_surface(X, Y, restored_magnitude_spectrum, cmap="inferno")
144 plt.title('Magnitude Plot of Restored image')
145 fig.colorbar(surf, ax=ax)
146 plt.savefig("../images/RestoredImageMagnitudeSpectrumPlot.png", bbox_inches="tight")

```

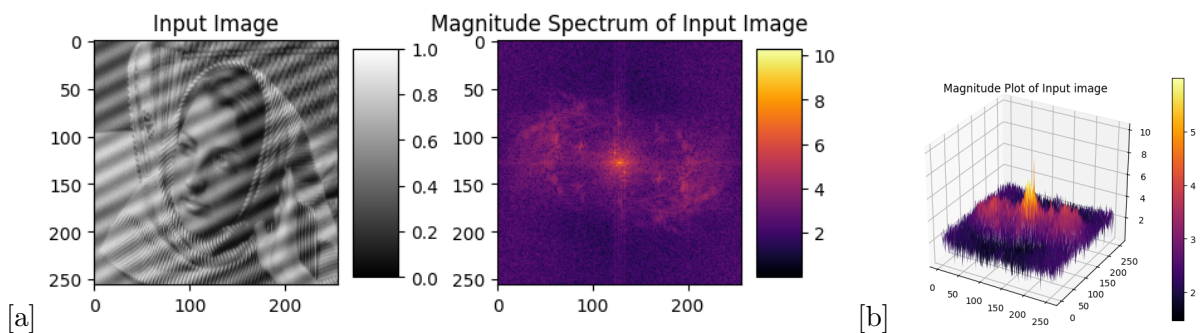


Figure 1: (a) Input Image (b) Magnitude Spectrum of input image

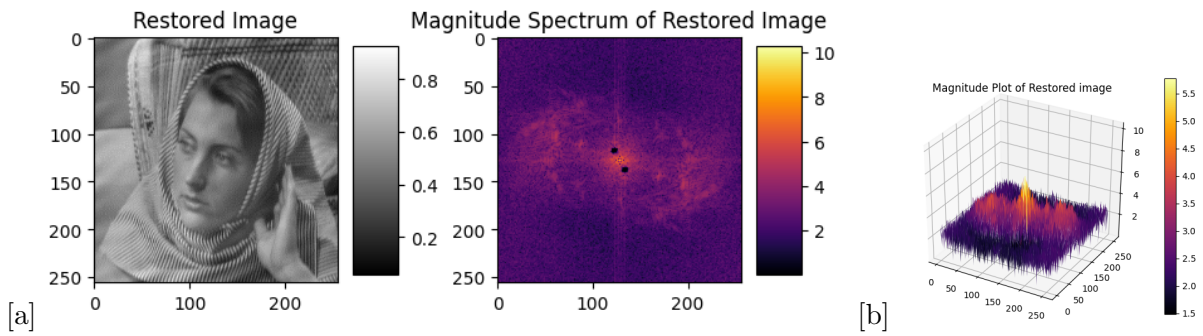


Figure 2: ((a) Restored Image (b) Magnitude Spectrum of restored image

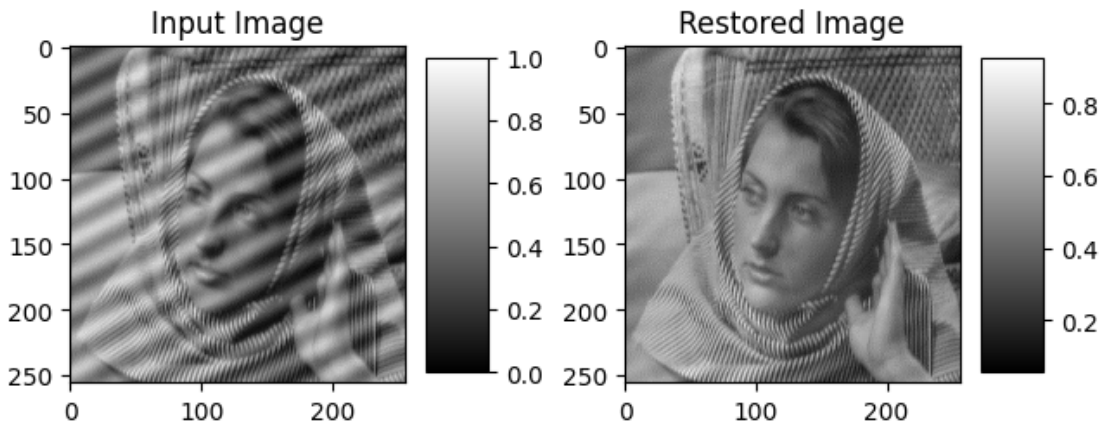


Figure 3: Comparison of Input and Restored Images

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.