DIP HW 5-1

311505011 人工智慧碩二 黃丰楷

1. Please use Inverse Filter and Wiener Filter to correct the image corrupted severe turbulence of the assigned image, ‘Fig5.25.jpg’, and print out the source code and the corrected image? (40)

- **Degradation function**

  First, I designed the degradation function as following:

$$H(u,v) = e^{-k[(u+M/2)^2+(v-N/2)^2]^{5/6}}$$

- **Full inverse filter**

  Then the estimated restored image using full filtering transfer function as following, where $G(u,v)$is the Fourier transform of the blurred image.

$$\hat{F}(u,v) = \frac{G(u,v)}{H(u,v)}$$

- **Radially inverse filter**

  However, even if we know the degradation function, we cannot recover the undegraded image exactly because the Fourier transform of the noise $N(u,v)$ is not known. Furthermore, If the degradation function has zero or very small values, then the ratio $N(u,v)/H(u,v)$ could easily dominate the original image. That is, I cut off values of the ratio $G(u,v)/H(u,v)$ outside a radius 60, and it was implemented by applying to the ratio a Butterworth lowpass function of order 10, which is formulated as:

$$H(u,v) = \frac{1}{1+[D(u,v)/D_0]^{2n}},$$

where D_0 is cutoff frequency (set to 60), and n is the order of the Butterworth filter (set to 10).

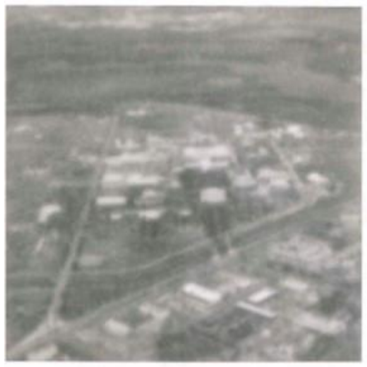
- **Wiener Filter**

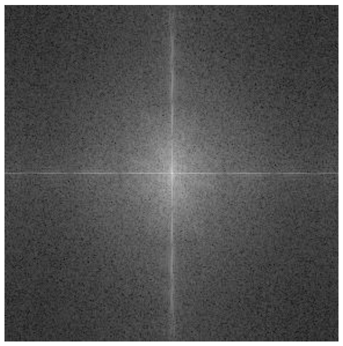
  Wiener filter is an approach that incorporates both the degradation function and statistical characteristics of noise into the restoration process. The transfer function is:

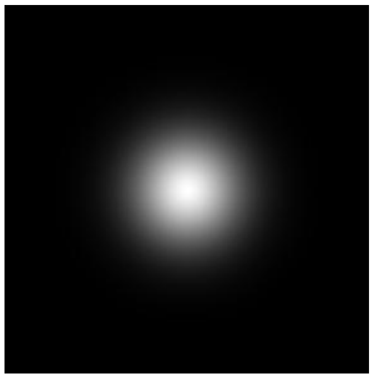
$$\hat{f}(u,v) = \left[\frac{1}{H(u,v)}\frac{|H(u,v)|^2}{|H(u,v)|^2+K}\right],$$

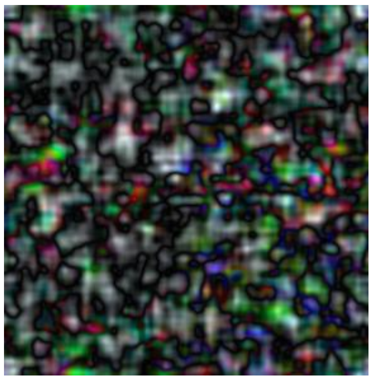
  where $K = 0.0001$.

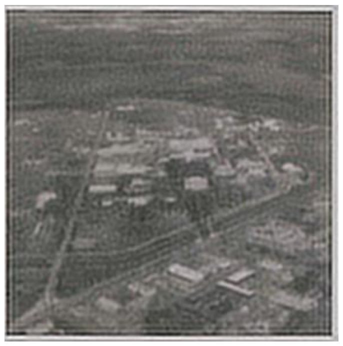
- **Results**

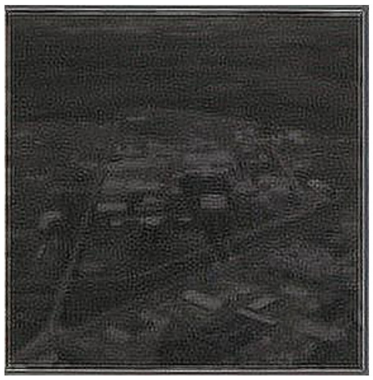

(a) Blurred image


(b) *G(u,v)*


(c) *H(u,v)*


(d) Result of full inverse filter


(e) Result of radially inverse filter


(f) Result of Wiener filter

- **Code**

```
import cv2
import matplotlib.pyplot as plt
import numpy as np


def ifft(input):
    ifft_img = np.fft.ifft2(np.fft.ifftshift(input))
    ifft_img = np.abs(ifft_img)
    ifft_img = (ifft_img/ifft_img.max()) *255


    return ifft_img




img = cv2.imread("Fig5.25.jpg")
plt.imsave("1_original.jpg",img, cmap='gray')


M, N, c  = img.shape
restored_inv_full = np.zeros_like(img)
restored_inv_rap = np.zeros_like(img)
restored_wie = np.zeros_like(img)


## degradation function H(u, v)
k = 0.0025
x, y = np.mgrid[0:M, 0:N]
H = np.exp(-k * (((x-M/2)**2 + (y-N/2)**2)**(5/6)) )
plt.imsave("1_H.jpg",np.log(abs(H) + 1), cmap='gray')


## Butter filter
radio = 60
n = 10
x, y = np.mgrid[-M//2:M//2, -N//2:N//2]
butter = 1/(1+((x**2+y**2)/radio**2)**(n))   # 課本 p.282公式

fig = plt.figure()
plt.plot(butter.ravel())
plt.savefig("1_butter.jpg")

for i in range(c):
    G = np.fft.fftshift(np.fft.fft2(img[:,:,i]))
    plt.imsave("1_G.jpg",np.log(abs(G) + 1), cmap='gray')


    ## inverse filter full
    F_full = G/H
    ##inverse filter radially
    F = F_full*butter

    ## wiener filtering
    K = 0.0001
    weiner_term = (abs(H)**2/(abs(H)**2+K))
    F_wie = (G/H) * weiner_term

    restored_inv_full[:,:,i] = ifft(F_full)
    restored_inv_rap[:,:,i] = ifft(F)
    restored_wie[:,:,i] = ifft(F_wie)


plt.imsave("1_Inverse_Full.jpg",restored_inv_full, cmap='gray')
plt.imsave("1_Inverse_Radially.jpg",restored_inv_rap, cmap='gray')
plt.imsave("1_Wiener.jpg",restored_wie, cmap='gray')
```

2. Please use Inverse Filter and Wiener Filter to correct the image corrupted by motion blur of the assigned image, 'book-cover-blurred.tif', and print out the source code and the corrected image?(40)

- **Degradation function**

  First, I designed the degradation function as following:

$$H(u,v) = \frac{T}{\pi(ua+vb)} \sin\left[\pi(ua+vb)\right] e^{-j\pi(ua+vb)},$$

  where $a = b = 0.1$, $T = 1$.

- **Full inverse filter**

  Then the estimated restored image using full filtering transfer function as following, where $G(u,v)$is the Fourier transform of the blurred image.

$$\hat{F}(u,v) = \frac{G(u,v)}{H(u,v)}$$

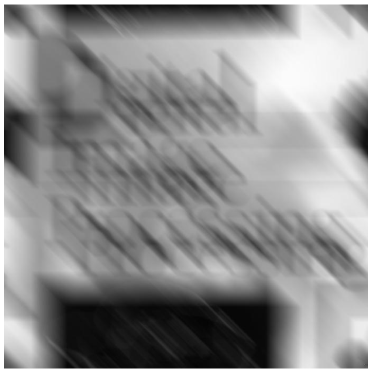
- **Wiener Filter**

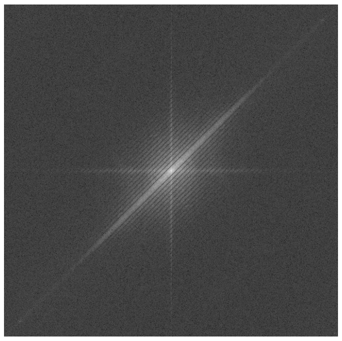
Wiener filter is an approach that incorporates both the degradation function and statistical characteristics of noise into the restoration process. The transfer function is:

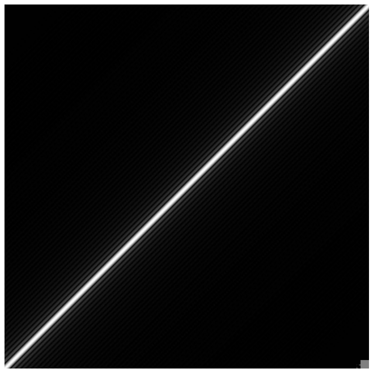
$$\hat{f}(u,v) = \left[\frac{1}{H(u,v)} \frac{|H(u,v)|^2}{|H(u,v)|^2 + K} \right],$$

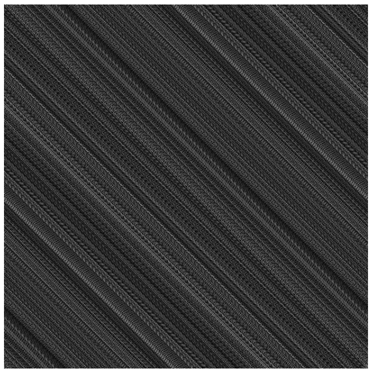
where $K = 0.0001$.

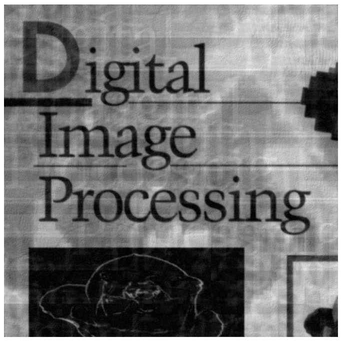
- **Results**


(a) Bluured image


(b) *G(u,v)*


(c) *H(u,v)*


(d) Result of full inverse filter


(e) Result of Wiener filter

- **Code**

```
import cv2
import matplotlib.pyplot as plt
import numpy as np
```

```
def ifft(input):
    ifft_img = np.fft.ifft2(np.fft.ifftshift(input))
    ifft_img = np.abs(ifft_img)
    ifft_img = (ifft_img/ifft_img.max()) *255


    return ifft_img





img = cv2.imread("book-cover-blurred.tif")
plt.imsave("2_original.jpg",img, cmap='gray')

M, N, c  = img.shape
restored_inv_full = np.zeros_like(img)
restored_wie = np.zeros_like(img)


## degradation function H(u, v)
a = 0.1
b = 0.1
T = 1

u, v = np.mgrid[-M//2:M//2, -N//2:N//2]
H = (T/(np.pi*(u*a+v*b))*np.sin(np.pi*(u*a+v*b)))*np.exp(np.pi*(u*a+v*b)*-1j)
H[np.isnan(H)] = T
plt.imsave("2_H.jpg",np.log(abs(H) + 1), cmap='gray')





for i in range(c):
    G = np.fft.fftshift(np.fft.fft2(img[:,:,i]))
    plt.imsave("2_G.jpg",np.log(abs(G) + 1), cmap='gray')


    ## inverse filter full
    F_full = G/H

    ##inverse filter radially

    ## wiener filtering
    K = 0.0001
    weiner_term = (abs(H)**2/(abs(H)**2+K))
    F_wie = (G/H) * weiner_term

    restored_inv_full[:,:,i] = ifft(F_full)
    restored_wie[:,:,i] = ifft(F_wie)
```

```
plt.imsave("2_Inverse_Full.jpg",restored_inv_full, cmap='gray')
plt.imsave("2_Wiener.jpg",restored_wie, cmap='gray')
```

3. Please comment and compare your two design filters? (20)

In the first image, using full inverse filtering alone yields poor results. This is primarily because the Fourier transform of the noise is unknown, making even slight alterations cause a significant impact. Consequently, employing a butterworth lowpass filter to truncate the transfer function visibly enhances the clarity of the picture compared to the original. In contrast, the Wiener filter, by incorporating a Wiener term, effectively constrains the randomness of unknown noises, leading to a clearer restoration without truncation.

Moving to the second image, which suffers from motion blur, the Gaussian degradation function used in the first scenario proves ineffective. Instead, employing inverse filtering with a degradation function involving a positive constant still produces an unusable image. The noise in the inverse filtered image is overwhelmingly strong, obscuring the image's content. However, the Wiener filter excels in restoring images afflicted by motion blur.

Considering both scenarios, post Wiener filtering, the picture's brightness tends to appear darker compared to the original.