

project3

source code

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

def get_norm_magnitude(ft):
    mg = np.abs(ft)
    scale = 1/(np.log(np.max(mg)+1))
    mg = scale*np.log(mg+1)*255
    mg = np.clip(np.uint8(mg), 0, 255)
    return mg

def get_masked(H, radius):
    rows, cols = H.shape
    crow, ccol = rows//2, cols//2

    mask = np.zeros((rows, cols), np.float64)
    for x in range(rows):
        for y in range(cols):
            dist = np.linalg.norm(np.array([x, y]) - np.array([crow, ccol]))
            if dist < radius:
                mask[x, y] = H[x, y]
            else:
                mask[x, y] = 1

    return mask

# read degraded image
g = plt.imread('Bird 2 degraded.tif')
g = np.float64(g)/255.0
print(g.shape)

# dtft
G = np.fft.fft2(g)
G = np.fft.fftshift(G)

# magnitude of dtft
G_m = get_norm_magnitude(G)
cv2.imwrite('G_magnitude.png', G_m)
```

```
# degradation model H
H = np.zeros((G.shape), np.float64)
k = 0.001 # mild turbulence
for u in range(H.shape[0]):
    for v in range(H.shape[1]):
        H[u][v] = math.exp(-k*(math.pow(
            (u-H.shape[0]/2)**2+(v-H.shape[1]/2)**2, 5/6)))

H_m = get_norm_magnitude(H)
cv2.imwrite('H_magnitude.png', H_m)

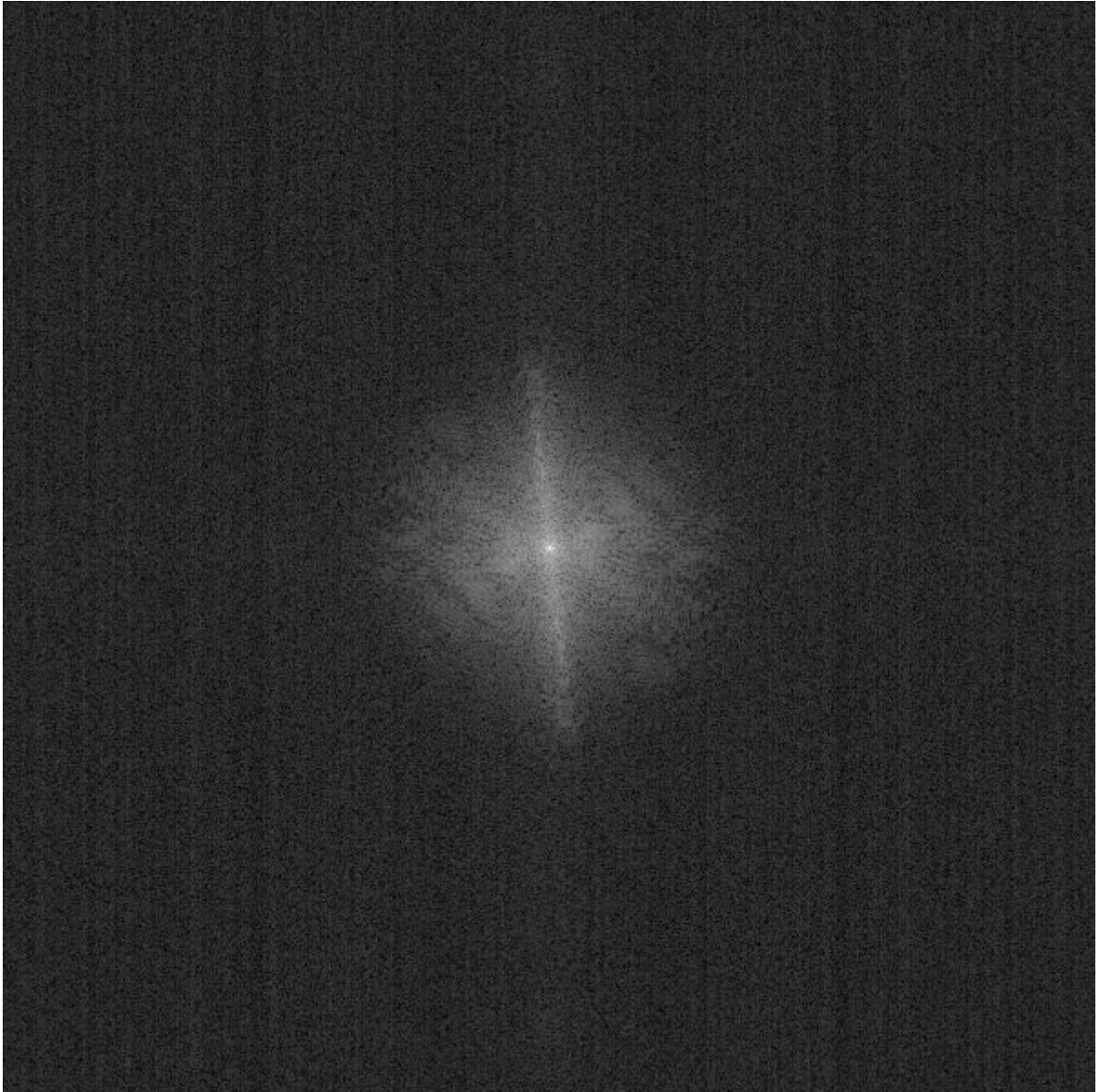
# inverse filter
for r in [50, 85, 120]:
    F_hat = G/get_masked(H,r)
    F_hat = np.nan_to_num(F_hat)
    cv2.imwrite('F_hat_magnitude_%d.png'%r, get_norm_magnitude(F_hat))
    F_hat = np.fft.ifftshift(F_hat)
    f_hat = np.fft.ifft2(F_hat)
    f_hat = np.clip(np.uint8(np.abs(f_hat)*255), 0, 255)
    cv2.imwrite('restoration_%d.png'%r, f_hat)
```

黃瑞得

0860078

huangjuite@gmail.com

degraded image FFT



degradation model H

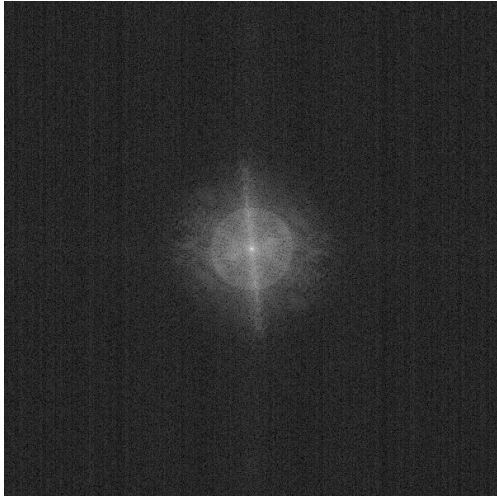
mild atmospheric turbulence, estimate $k = 0.001$

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



restoration

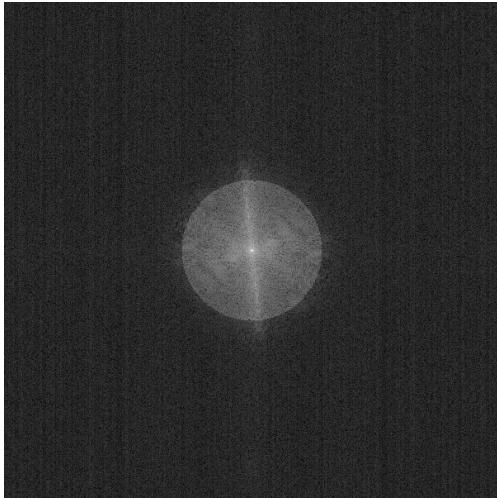
F_hat magnitude radius 50



restoration radius 50



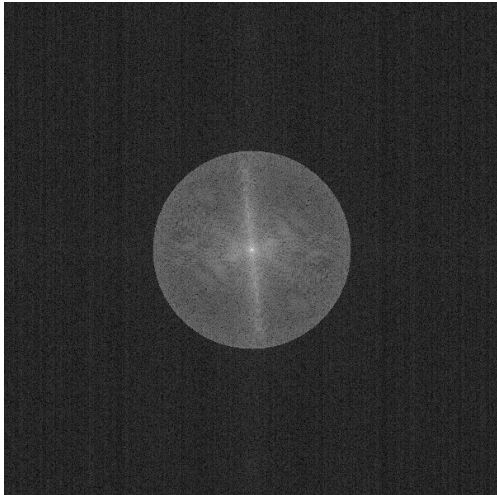
F_hat magnitude radius 80



restoration radius 80



F_hat magnitude radius 120



restoration radius 120

