Project #3

DIGITAL IMAGE PROCESSING

Student: 黎文雄 (Le Van Hung)

Student ID: 0860831

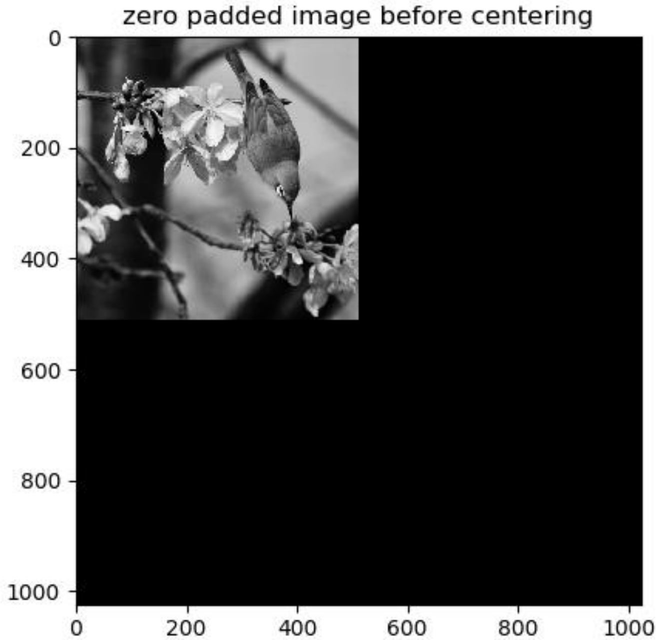
Requirement:

- According to the DFT property of Laplacian, it appears we may implement Laplacian operation by designing a digital filter with frequency response $H(u,v) = K(u^2+v^2)$ where K is a scaling factor to make magnitude of H(u,v) in the range [0,1]. Use this frequency-domain scheme to find the Laplacian image for the bird image.
- *Your report should contain:*
- Source codes
- Figures of Fourier magnitude spectra of bird image after applying Laplacian filtering.
- Figure of the Fourier magnitude of Laplacian filter H(u,v)
- Figure of the output image
- Table of top 25 DFT frequencies (u,v) after Laplacian filtering.

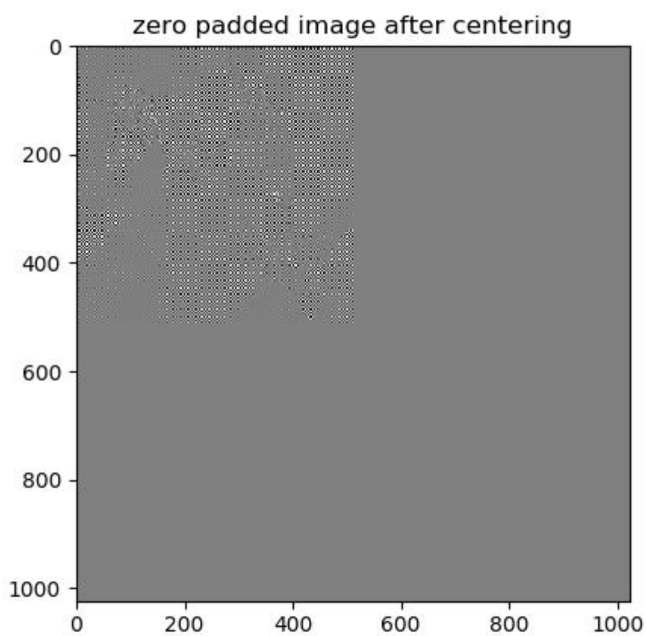
Original image

I will do this task by following step by step:

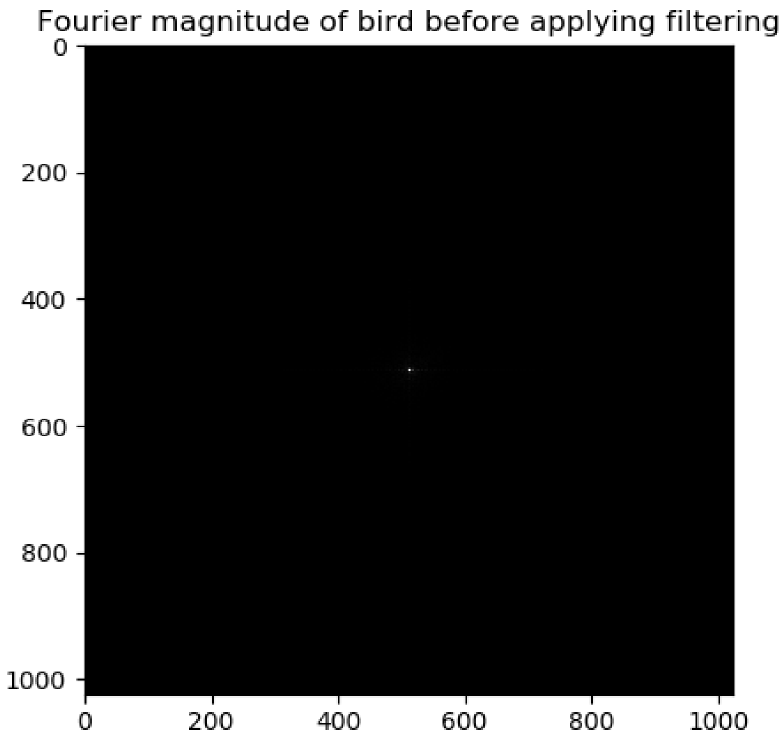
Step1: Zero padding to image to get new image with size (1024 x 1024)



Step3: Centering picture by multiplying input image (f(x,y) by $(-1)^{x+y}$



Step2: compute the DFT F(u,v)



Step3: Create filter transfer function H(u,v) with size (P=2*N = 1024; Q = 2*M = 1024) and the center at (P/2. Q/2) .

$$H(u,v) = -\frac{(u-512)^2 + (v-512)^2}{2*512^2*512^2}$$

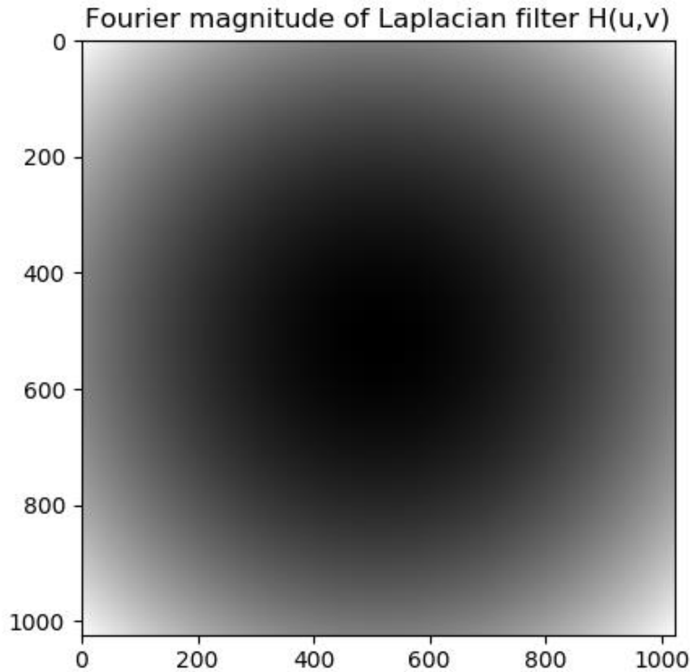
Where:

$u = 0,1,2....1023$

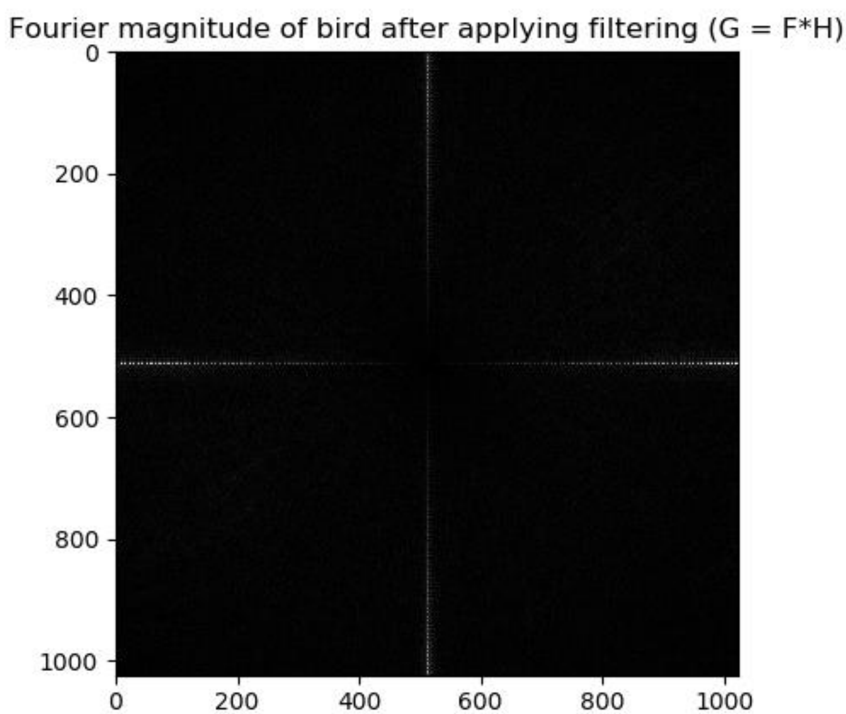
$v = 0,1,2....1023$

$H(u,v)_{\min} = 0 \quad with \;\; u = 512, v = 512 \; (in center)$

$H(u,v)_{\max} = 1 \quad with \;\; u = 0, v = 0$



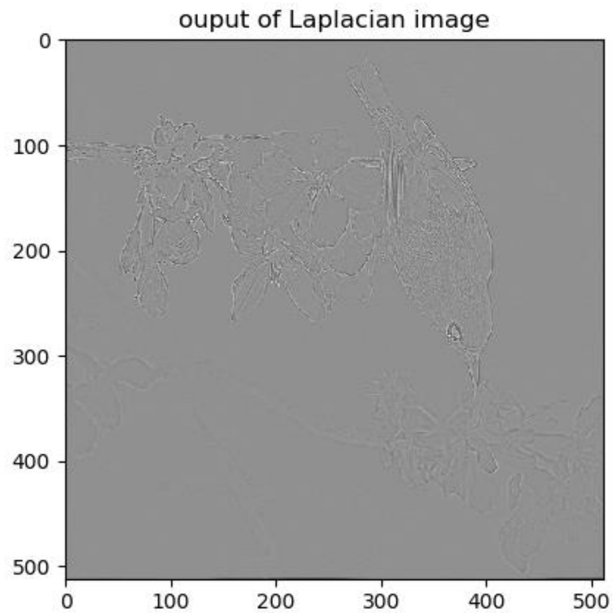
Step4: compute G(u,v)=H(u,v)*F(u,v) using element wise.

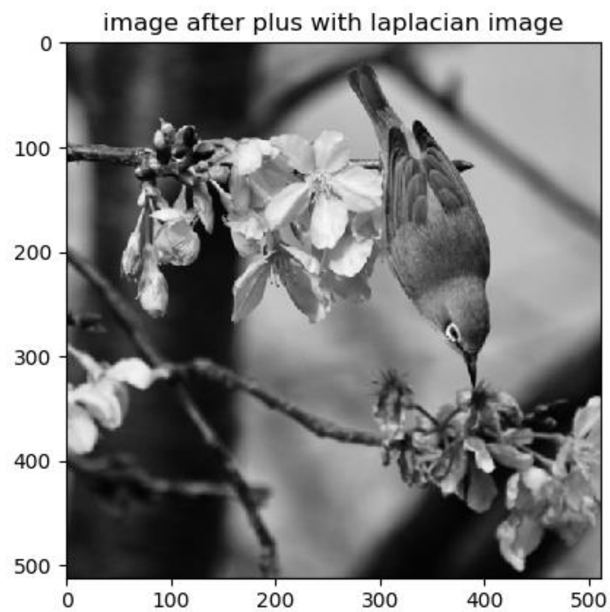


Step5: obtain the filter image by computing the IDFT of G(u,v) and get the real part of G^{-1}

$$g_p(x,y) = \left(\text{real}\left[\Im^{-1}\left\{G(u,v)\right\}\right]\right)(-1)^{x+y}$$

Step6: obtain the final filtered result, g(x,y) of the same size as the input image, by extracting the 512 x 512 region from the top,left quadrant of $g_p(x,y)$





- Table of top 25 DFT frequencies (u,v) after Laplacian filtering.

u	v	|F(u,v)|
512	1021	33336.19476621065
512	3	33336.19476621065
512	1023	32977.226214974944
512	1	32977.226214974944
512	1017	32621.762212720216
512	7	32621.762212720216
512	1011	31957.4559552902
512	13	31957.4559552902
512	1015	31538.114610405988
512	9	31538.114610405988
512	1007	31315.494180603644
512	17	31315.494180603644
512	1013	31310.64669798763
512	11	31310.64669798763
512	1019	31228.921810230717
512	5	31228.921810230717
512	1009	30931.837941813683
512	15	30931.837941813683
512	1003	30719.85401049823
512	21	30719.85401049823
512	1001	30561.37309926691
512	23	30561.37309926291
512	997	30530.85885783546
512	27	30530.85885783546
512	1005	30142.37275449174

- **Source Code (use python)**

```
#2020/04/22
#National Chiao Tung University
#Digital Image Processing
#Mini project NO.3
#Created by Le Van Hung (0860831)

# import library
from PIL import Image
import numpy as np
import matplotlib.pyplot as plt

# import image
```

```python
image = Image.open('Bird 1.tif')
image.show()

# convert image to array version
f = np.array(image,dtype='float')

#plot original picture
plt.imshow(f,cmap='gray')
plt.title("original image")
plt.show()
#define value
N = f.shape[0]
P = 2*N

# zeros padding to get new image fp 2N*2N
fp = np.zeros((2*N,2*N),dtype='float')
fp = np.array(fp)
fp[0:N,0:N] = f

#plot zero padded image before centering
plt.imshow(fp,cmap='gray')
plt.title("zero padded image before centering")
plt.show()

# multi fp with (-1)^(x+y)
for x in range(P):
    for y in range(P):
        fp[x,y] = fp[x,y]*pow(-1,(x+y))

# DFT
Fp = np.fft.fft2(fp)

# set up H (laplacian filter)
H = []
K = -1/(2*(N)*(N))
for u in range(P):
    for v in range(P):
        temp = K*((u-N)*(u-N)+(v-N)*(v-N))
        H.append(temp)
H = np.reshape(H,[P,P])
H_abs = np.abs(H)

# element wise F with H to get G
G = np.multiply(Fp,H)
# calculate gp (IDFT)
gp = np.fft.ifft2(G)
```

```python
# get real of gp
gp_real = gp.real
gp_real = np.array(gp_real,dtype='float')

# multi gp with (-1)^(x+y)
for x in range(N):
    for y in range(N):
        gp_real[x,y] = gp_real[x,y]*pow(-1,(x+y))

g = gp_real[0:N,0:N]

#plot zero padded image after centering
plt.imshow(fp,cmap='gray')
plt.title("zero padded image after centering")
plt.show()

# plot magnitude spectra of F
F_abs = np.abs(Fp)
plt.imshow(F_abs,cmap='gray')
plt.title("Fourier magnitude of bird before applying filtering")
plt.show()

# plot magnitude spectra of G = F*H
G_abs = np.abs(G)
plt.imshow(G_abs,cmap='gray')
plt.title("Fourier magnitude of bird after applying filtering (G
= F*H)")
plt.show()

# plot Fourier magnitude of Laplacian filter H(u,v)
plt.imshow(H_abs,cmap='gray')
plt.title("Fourier magnitude of Laplacian filter H(u,v)")
plt.show()

#plot Laplacian image
plt.imshow(g,cmap='gray')
plt.title("ouput of Laplacian image")
plt.show()

# plot image after plus with Laplacian image
plt.imshow(f+g,cmap='gray')
plt.title("image after plus with laplacian image")
plt.show()

# plot magnitude of original image
```

```
F = np.fft.fft2(f)
F_abs_0 = np.abs(F)
plt.imshow(F_abs_0,cmap='gray')
plt.title("Fourier magnitude of original image")
plt.show()

# show 25 top DFT frequencies (u,v) after Laplacian filtering
# sort 25 top of frequence [25 max abs()]
def find_max(array):
    len_array = array.shape[0]
    max = 0;
    row = 0;
    col = 0;
    for u in range(len_array):
        for v in range(len_array):
            if(array[u,v] >=max):
                max = array[u,v]
                row = u
                col = v
    return [row,col,max]
A_sort = []
G_temp = 1*G_abs
for i in range(25):
    temp = find_max(G_temp)
    G_temp[temp[0],temp[1]] = 0
    A_sort.append(temp)

A_sort = np.reshape(A_sort,(25,3))
```