

影像處理 LAB3

學號:103062129 姓名 :李國緯

4-1 Two-Dimensional Fast Fourier Transform

做法說明

有了圖片的 Size 之後，要對原始圖片做 padding，
padarray 是內建的 function，這邊的長寬是為原圖的兩倍，得(b)。再來要做 center shifting，

```
[M,N] = size(A);  
B = padarray(A,[M,N],0,'post');
```

$$f(x,y)(-1)^{x+y}$$

這裡我實現的方法是，用 double for loop，並用一個 matrix 儲存每個 $(-1)^{x+y}$ ，最後再跟原始圖片做 array multiplication (in matlab .*) 得到圖片(c)。Spatial Domain 的處理在此已經完成，將圖片用 myDFT2 function 做 2D 的 Fourier Transform，轉換到 Fourier Domain。因為轉換完後，array type 會是 complex single (double)，為了展示 spectrum，先利用 abs 取得 real 的部分，然後再用 log transform 做強化，以利顯示。

```
D = myDFT2(C);          此時，圖片的值會超出 1，而非 0~1。  
D = abs(D);             因此我再用 mat2gray，將值重新 map 到 0 與 1 之間。  
D = 100*log(D+1);       得到圖片(d)。  
D = mat2gray(D);
```

利用 4-3 的 myGLPF，即 Gaussian Lowpass Filtering， $D0 = 30$ ，得(e)。
將 specturm(為 myDFT2 直接做出來的，不是經過 abs 與 log 過後的)，與 Gaussian Lopass Filter 做 array multiplication(.*)運算，得(f)。
此時 Fourier 已經完成，轉回 spatial domain，用 myIDFT2，再將中心移回得到(g)。
將 padding 去除之後可得 filtered output，為圖(h)。

myDFT2：基本的做法就是依照公式解。

$$F(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-j2\pi(ux/M + vy/N)}$$

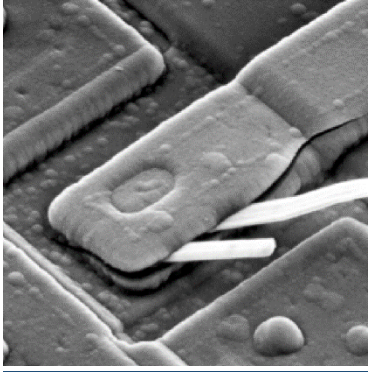
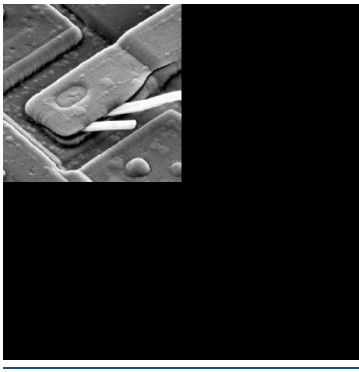
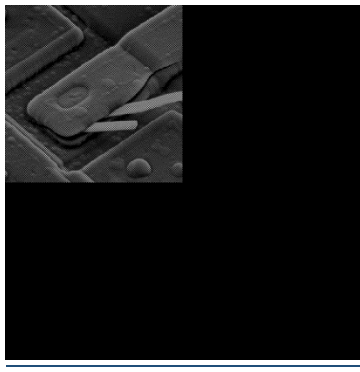
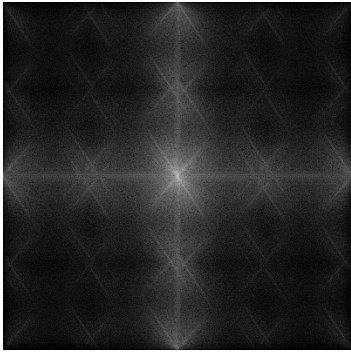
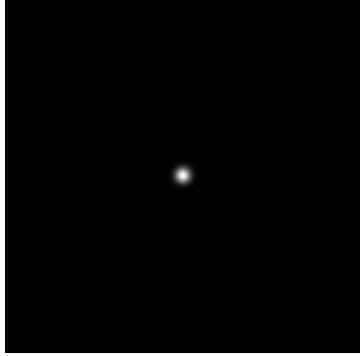
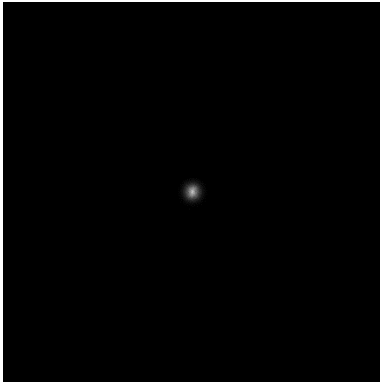
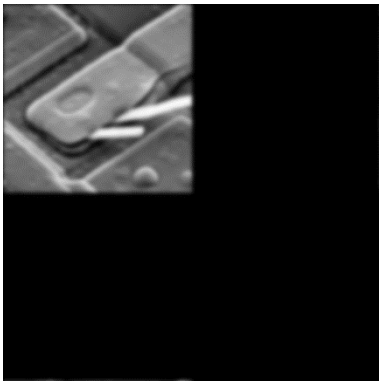
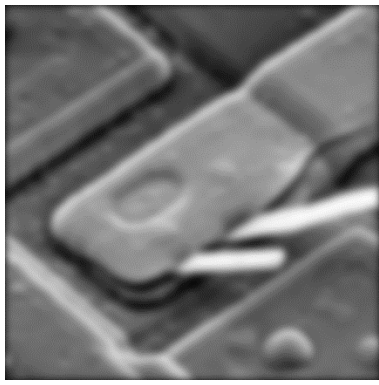
我將 $x(u)$, $y(v)$ 分開做 exponential 最後再利用 matrix multiplication (*) 將 x , input, y 相乘，得到 Output。

其中，exp(i)，是內建的函數，虛數的部分我用 1i 表示(matlab 建議)。

分開算第一個是可以避免看起來複雜的 4 層 for loop，再者可以利用 matlab 的矩陣相乘，而非傳統的點對點矩陣相乘。

myIDFT2: 幾乎與 myDFT2 相同，唯二的差別在於 exponential 係數少了一個負號，還有最後除以 $(M \times N)$ 。

結果圖片

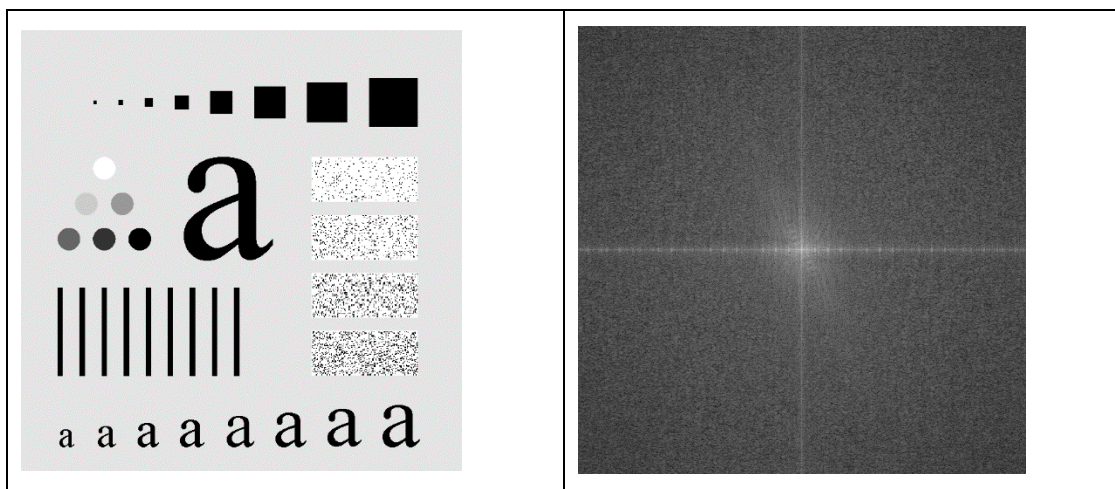
		
4.36(a) Original	4.36(b) padding	4.36(c) shift center
		
	4.36(d) Spectrum	4.36(e) Gaussian lowpass Filtering
		
4.36(f) filtered in fourier domain	4.36(g) back to spatial domain	4.36(h) remove padding

分析以及討論

觀察 Fourier Transform 我們可以發現運算結果具有週期性，也就是一切的根本，或說 frequency。這是個十分常用的方法，美中不足的部分就在於，速度。理論上而言，DFT 的時間是 $O(n^4)$ ，在實作上是 $O(n^2)$ 加上係數 (In matlab)。因此有了所謂的 Fast Fourier Transform (FFT)，速度為 $O(n \lg n)$ 。實作的方法理論上為 divide- and-conquer (recursive)，將 problem size 縮小，然後可以得到 $O(n \lg n)$ 。

4-2 Fourier Spectrum and Average Value

結果圖片



分析以及討論

Spectrum 的原點為 $(0,0)$ ，帶入公式會得到 $\text{exponential}^0 = 1$ 。因此這個公式就會變成，把原圖的每一點加起來 (sum)。而 $\text{mean} = \text{sum} / \text{image_size}$ ，所以 sum 就會和 mean 差 MN 倍。Mean 乘上 image size (MN) = centered image on Fourier domain 的 Real part (Spectrum)。

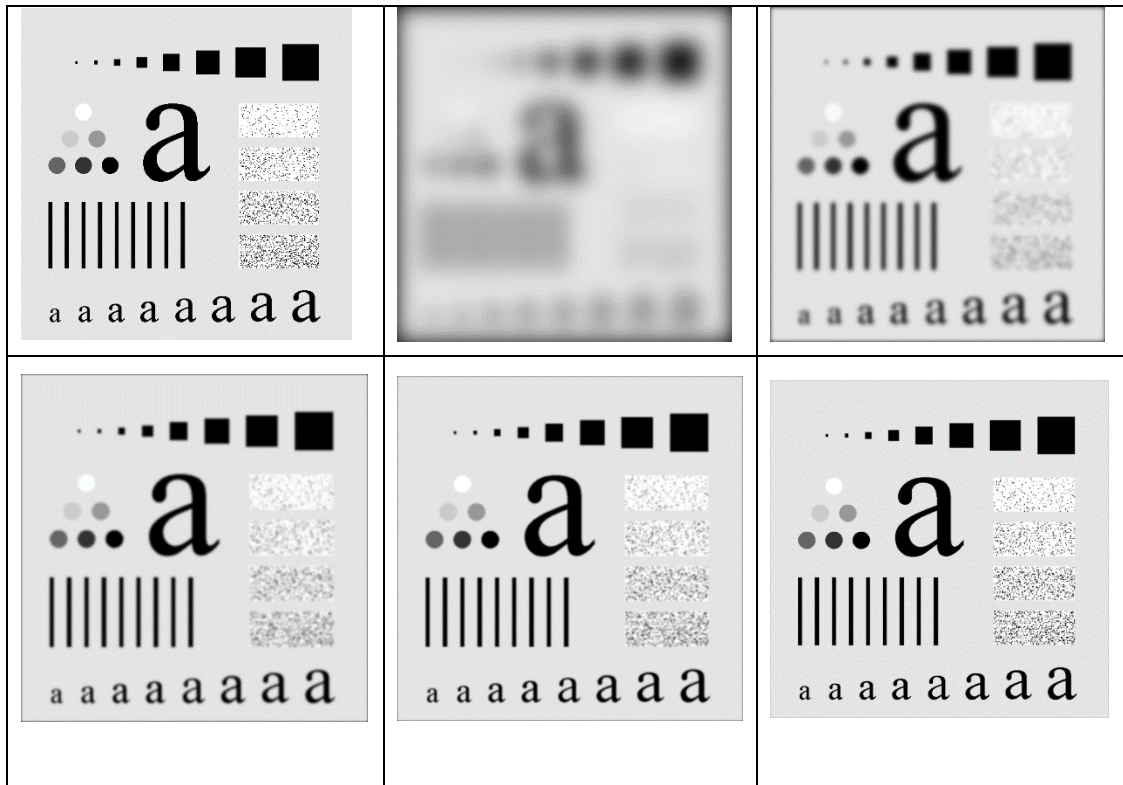
```
mean of Fig 4.41
0.8130
```

```
mean * MN
3.8483e+05
```

```
center of F
3.8483e+05 +4.3462e-08i
```

4-3 Lowpass Filtering

結果圖片



由上至下，左至右，分別為，

原圖， $D0 = 10$ ， $D0 = 30$

$D0 = 60$ ， $D0 = 160$ ， $D0 = 460$

分析以及討論

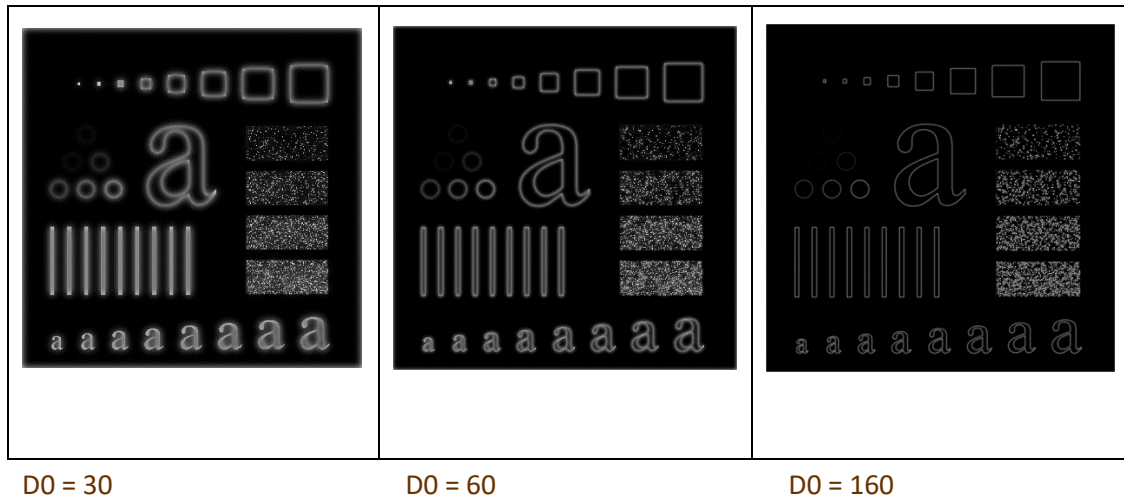
Lowpass Filter 顧名思義允許低頻的訊號通過，把高於一定頻率的訊號變成零。隨著 $D0$ 的變大，能通過原始 filter 的頻率就越多，模糊的效果就會越弱，如上圖顯示。在一般的圖片上做，也就是 spatial domain，會是週期性函數函數，實作上十分不容易，但如果是在 fourier domain 上做，經過公式的幫忙，就會是簡單的 array multiplication(.*)運算即可達成。也因此實作上才會將圖片先經 4-1 的步驟，轉至 Fourier domain。

GLPF 能將圖片變模糊(當 $D0$ 不大時)，與一般的 blur(average filter)差異性在於：

Average blur 是很粗糙的將圖片 blur (平均)，會產生非常多雜訊，看起來甚至會有一格一格的感覺，而 GLPF 處理 blur 上的效果較為理想。

4-4 Highpass Filtering

結果圖片



分析以及討論

相反於 GLPF，GHPF 就是允許高頻率的訊號通過，將低於一定頻率的訊號 eliminate。

當 $D0$ 越大，能通過的頻率就越多，效果越弱。

相反於 GLPF 的 blur，GHPF 就是 sharp，由上圖可得知，edge 被強化了，當 $D0$ 越大，效果越不明顯。

同樣是銳利化，相似的還有 unsharpFiltering(Laplacian)。

一個是直接對 image 做 filter，另一個是在 fourier domain 做文章。

我認為原理上，unsharp 效果不會比 GHPF 來的好，因為 Laplacian 會受周遭的影響多，在一些較為複雜的 edge 上比較難 detection。而 GHPF 是根據頻率上做調整，因此在整體的感覺上會較佳。

但實際的效果上，經測試(月球表面)，肉眼看起來差不多。



左圖為 GHPF。

右圖為 Laplacian
Mask。