多媒體project2報告

408410102 資工三 楊力行

A.方法描述

a.說明：

以下依照原題目進行程式說明：

(a) Creates the Gaussian filter with hsize = 3x3, 7x7 and 13x13. Apply image convolution to image 柴犬飛飛.jpg by three Gaussian filters, and compute the PSNR with the original image. [3 image]

根據程式碼，我使用自定義函式 my_conv_hsize(test, h)，輸入是轉成double的圖片，以及hsize。
function裡面先根據hsize創建對應大小的Gaussian filter，之後對輸入的原圖邊緣做補零，方便等一下計算卷積。
下面的迴圈計算了rgb各層的卷積，其中按照卷積的概念上下移動filter對圖片做卷積運算，並把值放到跟原圖大小一樣的新的三維矩陣中對應的位置，最後，圖片輸出成jpg檔。

(b) Creates the Gaussian filter with sigma = 1, 30 and 100 with hsize = 3x3. Apply image convolution to image 柴犬飛飛.jpg by three Gaussian filters, and compute the PSNR with the original image. [3 image]

基本上作法跟(a)相同。使用自定義函式 my_conv_sigma(test, s)，輸入是轉成double的圖片，以及sigma。
function裡面先根據sigma創建對應的Gaussian filter。剩下的動作皆與(a)相同，就不再贅述。

(c) Compare and discuss the results from the above three methods and give the meaning of PSNR values to these results.

此部分會在 B, C 部分討論。

(d) Apply Unsharp mask、Edge detection mask to the 柴犬飛飛.jpg and show the results.

1. Unsharp mask
   基本上作法跟(a)相同。定義一個 G = [0 -1 0; -1 5 -1; 0 -1 0] 作為 unsharp mask，接著利用迴圈做卷積運算，最後輸出圖片。
2. Edge detection mask
   基本上作法跟(a)相同。定義一個 G = [-1 -1 -1; -1 8 -1; -1 -1 -1] 作為 edge detection mask，接著利用迴圈做卷積運算，最後輸出圖片。

b.程式碼：

hw2.m

```
function hw2()
orig = imread('./pic/柴犬飛飛.jpg');
test = im2double(orig);

% hsize 3, 7, 13
my_conv_hsize(test, 3);
my_conv_hsize(test, 7);
my_conv_hsize(test, 13);

% sigma 1, 30 100
my_conv_sigma(test, 1);
my_conv_sigma(test, 30);
my_conv_sigma(test, 100);

% unsharp mask
my_unsharp_mask(test);

% edge detection mask
my_edge_detection(test);

end

function my_conv_hsize(test, h)
G = fspecial('gaussian', [h h], 1);
test1 = padarray(test, [(h-1)/2 (h-1)/2], 0, 'both');
img = zeros(640, 640, 3);

for i = 1:640
    for j = 1:640
        for layer = 1:3
            value = sum( test1(i:i+h-1, j:j+h-1, layer) .* G(:,:), 'all');
            if(value > 1)
                value = 1;
            end
            if(value < 0)
                value = 0;
            end
            img(i, j, layer) = value;
            %fprintf("%d\n",img(i, j, layer));
        end
    end
end

%psnr
[peaksnr, snr] = psnr(img, test);
```

```
fprintf('The PSNR in hsize %d value is %0.4f\n', h, peaksnr);

% imshow(img);
if(h == 3)
    file_name = './pic/a_1.jpg';
end
if(h == 7)
    file_name = './pic/a_2.jpg';
end
if(h == 13)
    file_name = './pic/a_3.jpg';
end
imwrite(img, file_name);
end

function my_conv_sigma(test, s)
G = fspecial('gaussian', [3 3], s);
test1 = padarray(test, [1 1], 0, 'both');
img = zeros(640, 640, 3);

for i = 1:640
    for j = 1:640
        for layer = 1:3
            value = sum( test1(i:i+2, j:j+2, layer) .* G(:,:), 'all');
            if(value > 1)
                value = 1;
            end
            if(value < 0)
                value = 0;
            end
            img(i, j, layer) = value;
            %fprintf("%d\n",img(i, j, layer));
        end
    end
end

%psnr
[peaksnr, snr] = psnr(img, test);
fprintf('The PSNR in sigma %d value is %0.4f\n', s, peaksnr);

% imshow(img);
if(s == 1)
    file_name = './pic/b_1.jpg';
end
if(s == 30)
    file_name = './pic/b_2.jpg';
end
if(s == 100)
    file_name = './pic/b_3.jpg';
```

```
end
imwrite(img, file_name);
end

function my_unsharp_mask(test)
G = [0 -1 0; -1 5 -1; 0 -1 0];
test1 = padarray(test, [1 1], 0, 'both');
img = zeros(640, 640, 3);

for i = 1:640
    for j = 1:640
        for layer = 1:3
            value = sum( test1(i:i+2, j:j+2, layer) .* G(:,:), 'all');
            if(value > 1)
                value = 1;
            end
            if(value < 0)
                value = 0;
            end
            img(i, j, layer) = value;
            %fprintf("%d\n",img(i, j, layer));
        end
    end
end

% imshow(img);
imwrite(img, './pic/d_unsharp.jpg');
end

function my_edge_detection(test)
G = [-1 -1 -1; -1 8 -1; -1 -1 -1];
test1 = padarray(test, [1 1], 0, 'both');
img = zeros(640, 640, 3);

for i = 1:640
    for j = 1:640
        for layer = 1:3
            value = sum( test1(i:i+2, j:j+2, layer) .* G(:,:), 'all');
            if(value > 1)
                value = 1;
            end
            if(value < 0)
                value = 0;
            end
            img(i, j, layer) = value;
            %fprintf("%d\n",img(i, j, layer));
        end
    end
end
```

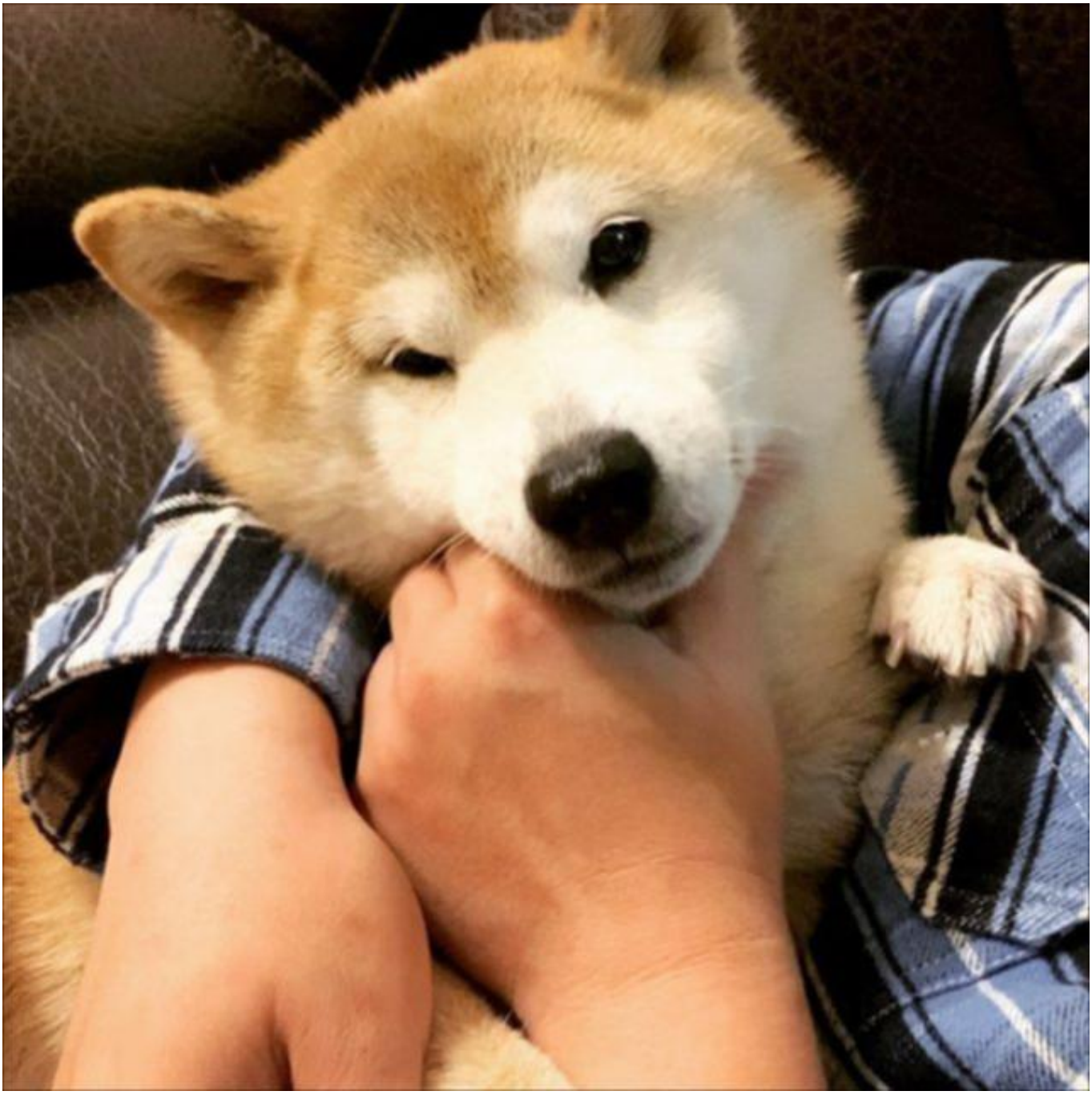
```
% imshow(img);
imwrite(img, './pic/d_edge.jpg');
end
```

B.實驗結果

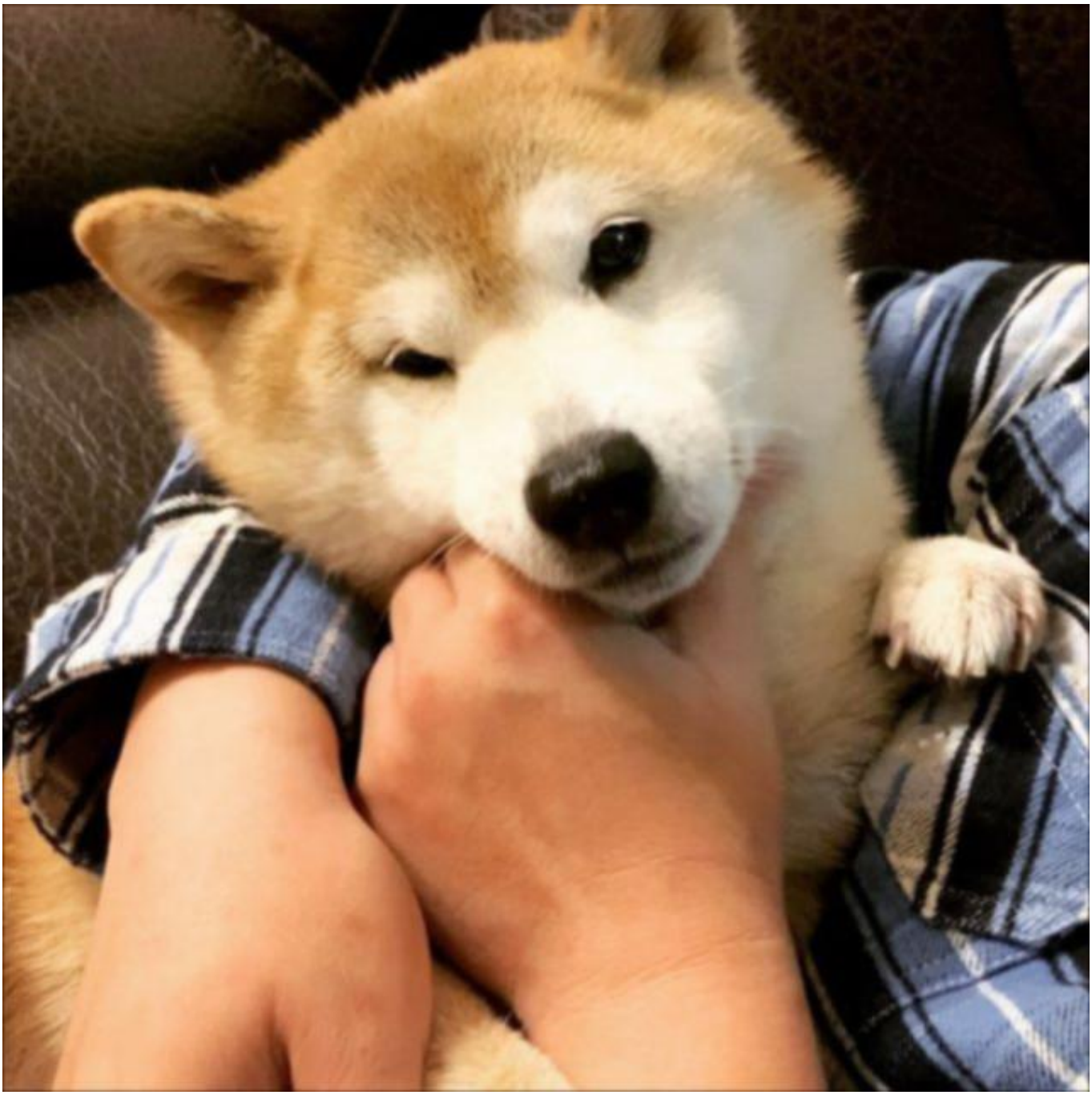
a.圖片

(a)

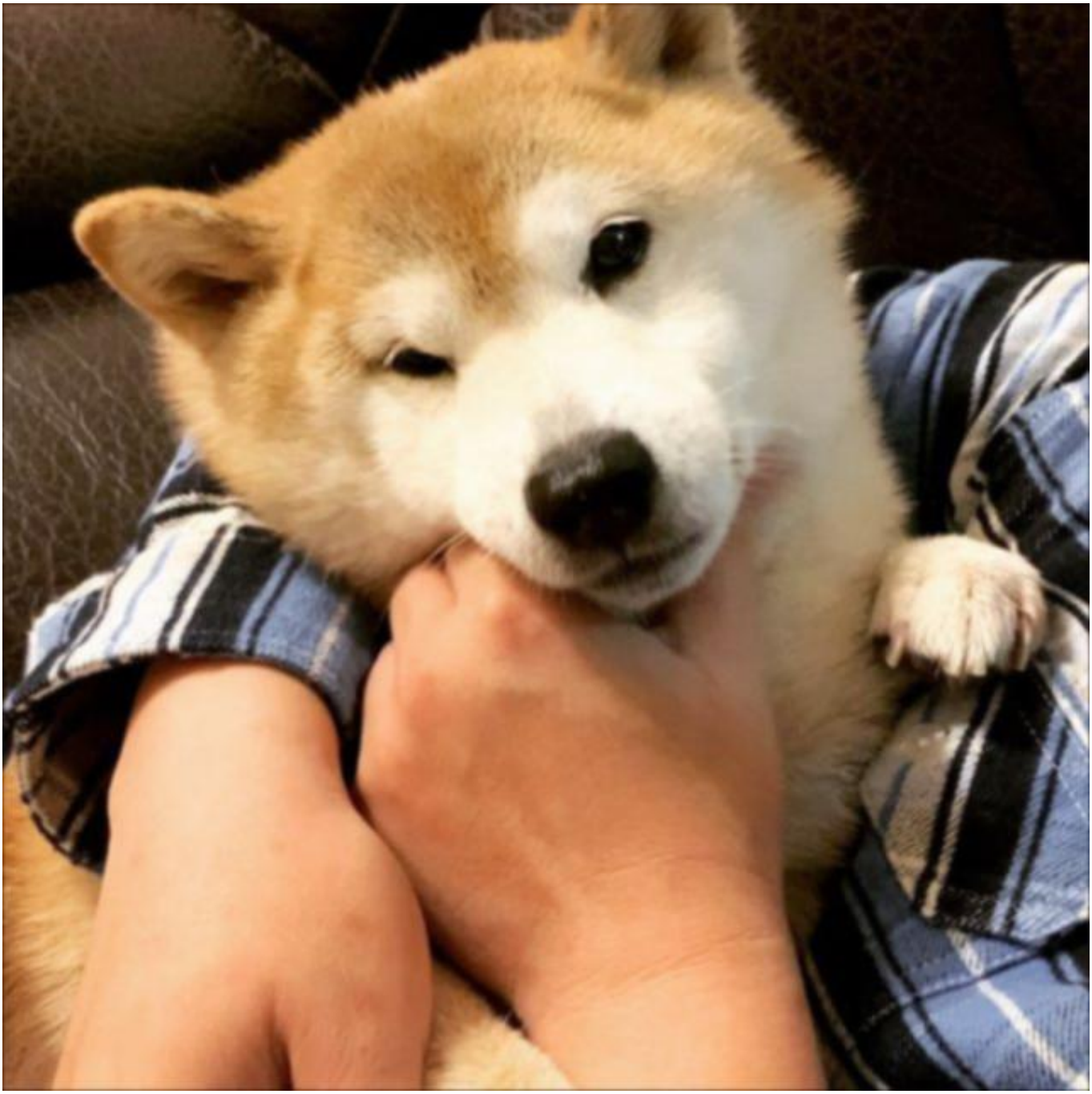
1. a_1.jpg (hsize 3)



2. a_2.jpg (hsize 7)

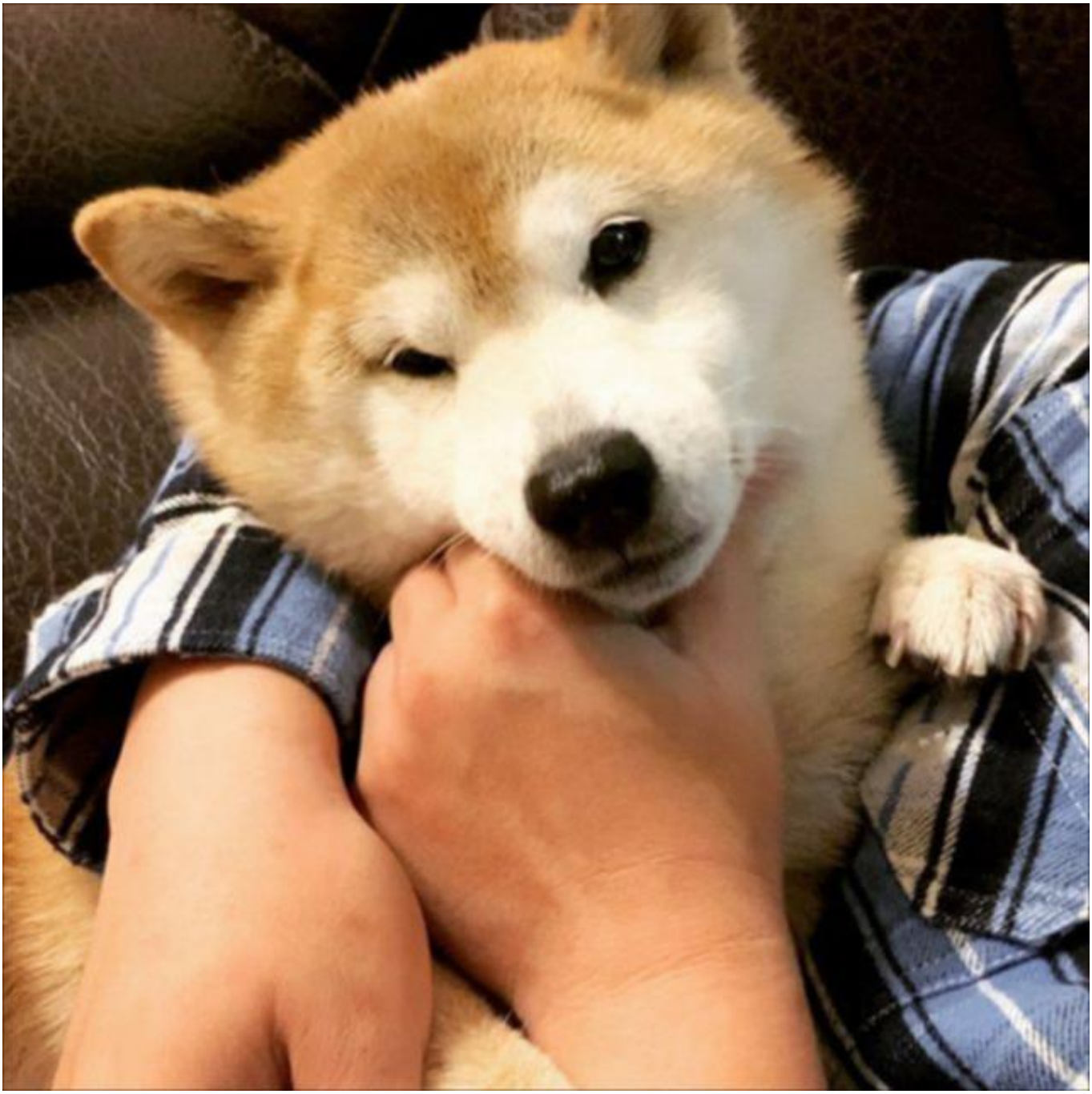


3. a_3.jpg (hsize 13)

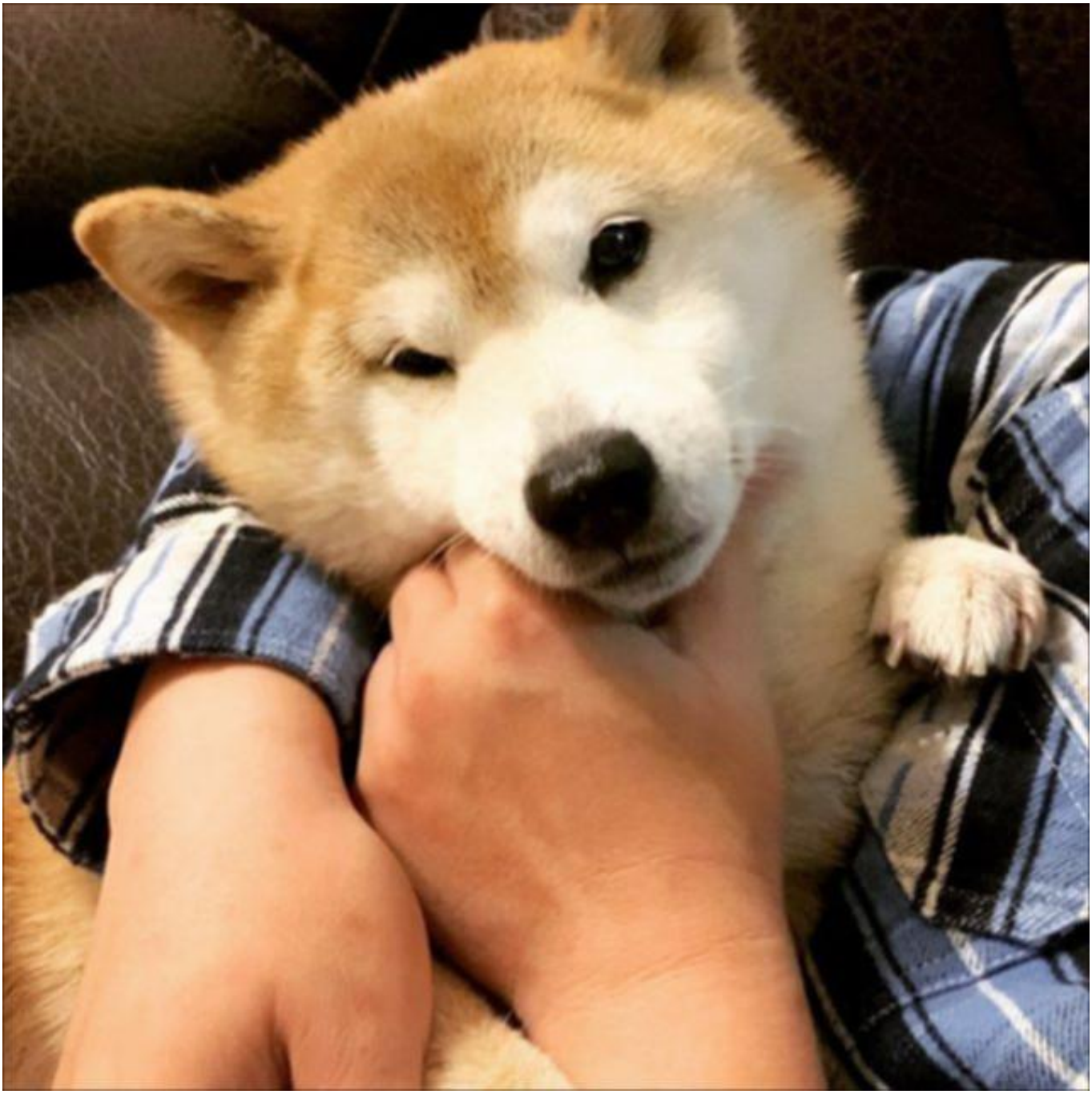


(b)

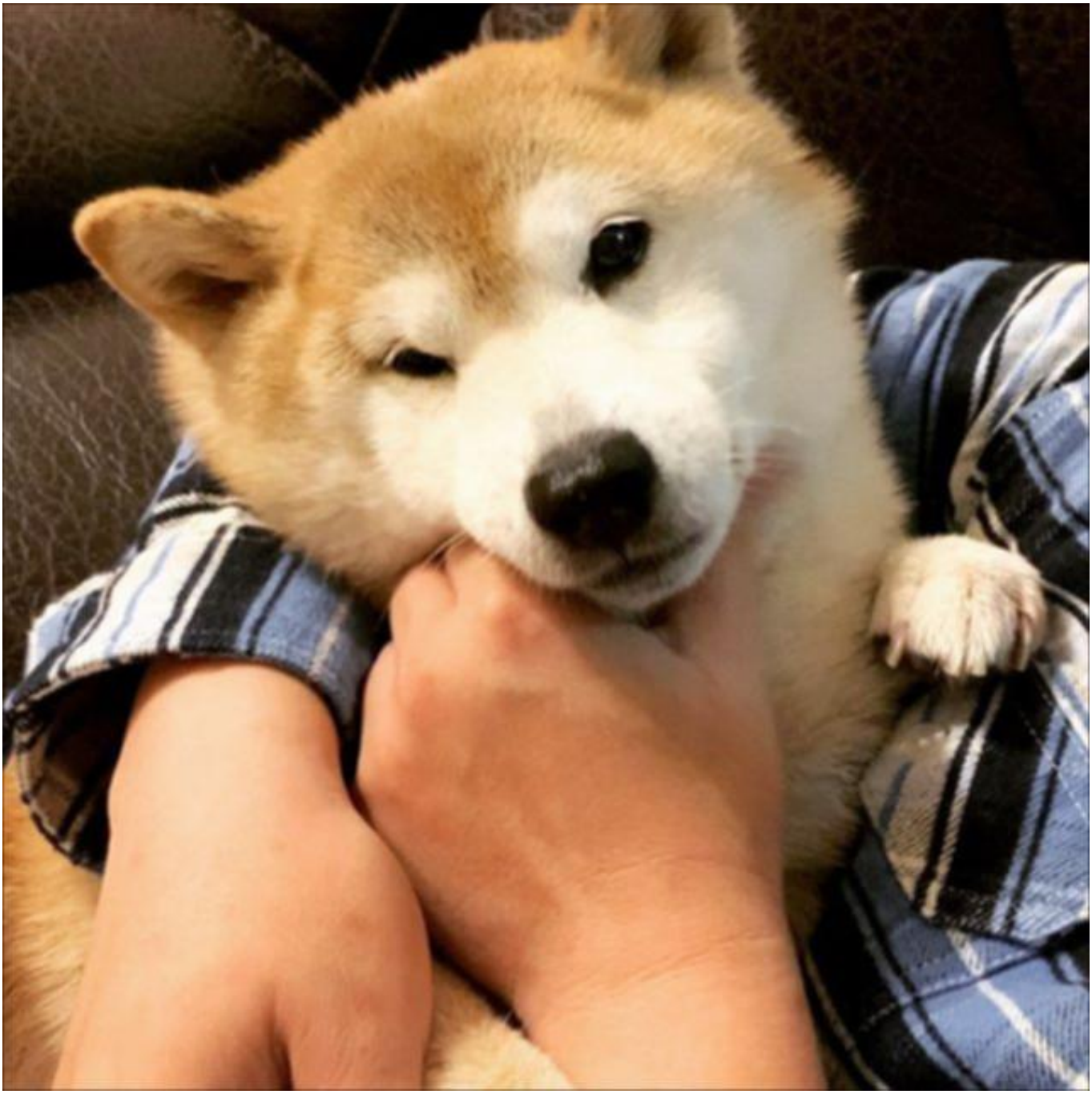
1. b_1.jpg (sigma 1)



2. b_2.jpg (sigma 30)

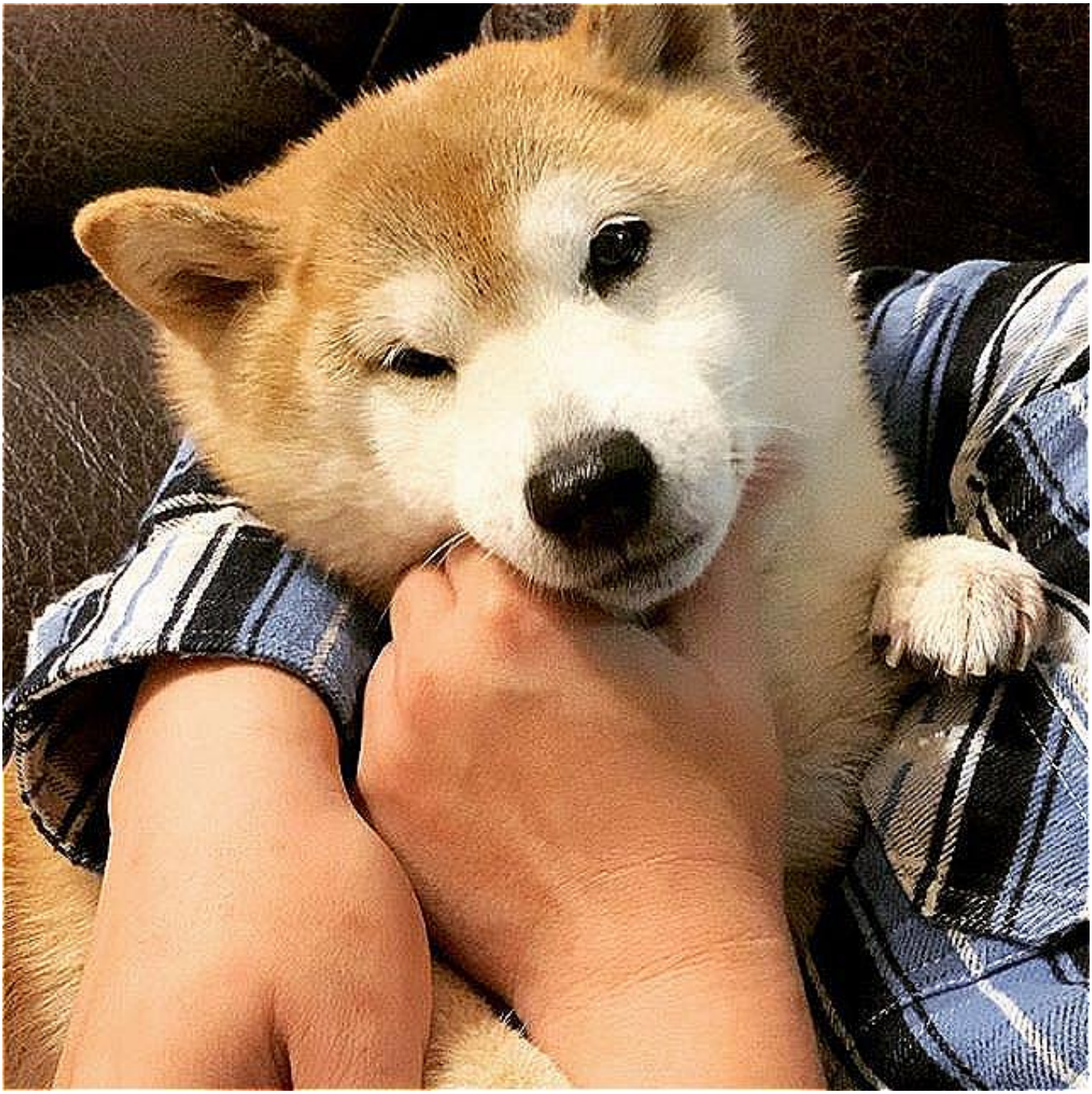


3. b_3.jpg (sigma 100)



(d)

1. d_unsharp.jpg



2. d_edge.jpg

b.PSNR

以下為程式執行後輸出的結果：
The PSNR in hsize 3 value is 32.7918
The PSNR in hsize 7 value is 31.0031
The PSNR in hsize 13 value is 30.9989
The PSNR in sigma 1 value is 32.7918
The PSNR in sigma 30 value is 31.3396
The PSNR in sigma 100 value is 31.3384

C.結果討論

a.圖片

(a)(b) 皆對原圖做了一定程度的模糊處理，一個是對filter大小做變化，一個是對sigma做變化，跟原圖做比較之後，肉眼可以明顯地發現圖片變模糊了。
(d)的unsharp mask對圖片做了銳利化，可以看得出與原圖的自然線條比起來，圖片多了一種難以形容的「銳利感」。而dege detection則把原圖的邊匡線條都抓了出來。

b.PSNR

PSNR (Peak Signal-to-Noise Ratio) 可以用來計算影像的失真。PSNR值越大，就代表失真越少。這是一個客觀的評比數據，但有時候並不能完全代表人的主觀感受。通常PSNR值越高表示品質越好，一般而言當PSNR的值<30db時，代表以人的肉眼看起來是不能容忍的範圍。因此大部分PSNR值皆要>30db。

1. **hsize**

   可以看出，當選擇的 filter size 越大時，PSNR 值有變小的趨勢，可以推測，如果卷積 filter 大小越大，失真的程度會比小的 filter 來得高一些。這是非常合理的，當 filter 變大，表示卷積需要處理的範圍變大，能影響原值的範圍也就越大，失真程度上升很正常。

2. **sigma**

   相同地，sigma 越大時，PSNR值會稍微變小一些。由於 sigma 越大會使圖片在更大的範圍內被模糊，所以失真程度在越大的 sigma 值上會更加明顯。

D.問題討論

做作業的時候最令人困擾的是：卷積運算怎麼做比較有效率？最後我使用一些數學運算，像是sum把矩陣的值直接相加；為了運算方便把邊匡依照filter size大小補0等，增加運算效率。不過如果之後可以完全不用迴圈那一定是最好了，可惜我還沒有那麼厲害。

後來發現圖片的值應該有一定的範圍，所以加上了值的邊緣檢測，為了運算正確性以及方便因素，我把原圖轉成0到1之間的double值，因此值的範圍是0到1之間。

剩下的其實都是一些粗心大意的小bug，這裡就不說了。

E.參考資料

資料來源：https://cg2010studio.com/2013/01/06/%E5%B3%B0%E5%80%BC%E4%BF%A1%E8%99%9F%E9%9B%9C%E8%A8%8A%E6%AF%94-peak-signal-to-noise-ratio/