計算機視覺

Computer Vision

Homework I

國立清華大學

系所:電子所碩二

中文姓名:李聖謙

學號:111063517

授課老師:Min Sun, 孫民

1. Implementation

1.1 Image filtering

Please finish the function my_imfilter in the file my_imfilter.py and briefly describe your implementation ideas. Noted that you can not use convolution function from any python built-in libraries (eg. numpy, scipy).

```
#    # ============================== Start OF YOUR CODE ==============================
import numpy as np

def my_imfilter(image, imfilter):
    height, width, channels = image.shape
    filter_height, filter_width = imfilter.shape

    output = np.zeros_like(image)

    pad_height = int((filter_height - 1) / 2)
    pad_width = int((filter_width - 1) / 2)
    padded_image = np.pad(image, ((pad_height, pad_height), (pad_width, pad_width), (0, 0)), 'constant', constant_values=(0, 0))
    print('padded_image', padded_image.shape)

    for i in range(height):
        for j in range(width):
            for ch in range(channels):
                pad = padded_image[i:i+filter_height, j:j+filter_width, ch]
                convolution = np.sum(pad * imfilter)
                output[i, j, ch] = convolution

    return output
#    # ============================== END OF YOUR CODE ==============================
```

Fig.1 my_imfilter

Algorithm Design:

本題的要求是設計一個卷積在不使用已經存在的 library 指令而是用其他方法完成卷積的功能，首先，卷積在公式上其實就是做乘法再相加，所以這裡就選擇參考了老師建議的 np.sum 語法實現圖片與濾波器的卷積，然後因為濾波器的規格都是二維的矩陣形式，所以在 for 迴圈的設計上將代表色彩的 RGB 獨立成三層然後一次只處理一層如 Fig.2 所示，也就是照片長寬的二維矩陣與濾波器的二維矩陣做卷積，濾波器的規格確定後，就下來就是對輸入照片如何配合濾波器做卷積，這裡利用的 python 的 slice 語法達成框選出與濾波器相同大小的矩陣([i:i+filter_height, j:j+filter_width, ch])。

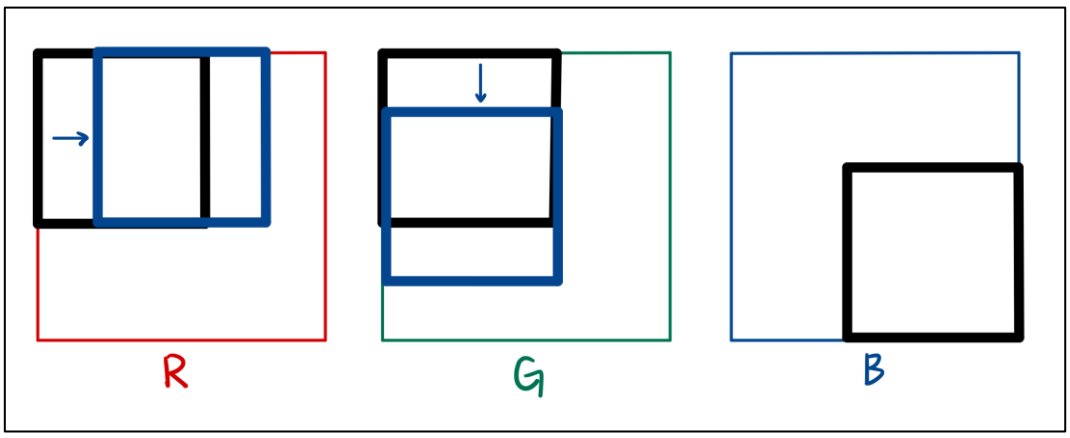

Fig.2 Convolution

第二個設計上會遇到的問題是當原始照片經過卷積後照片尺寸會變得比原本小張如 Fig.3 所示

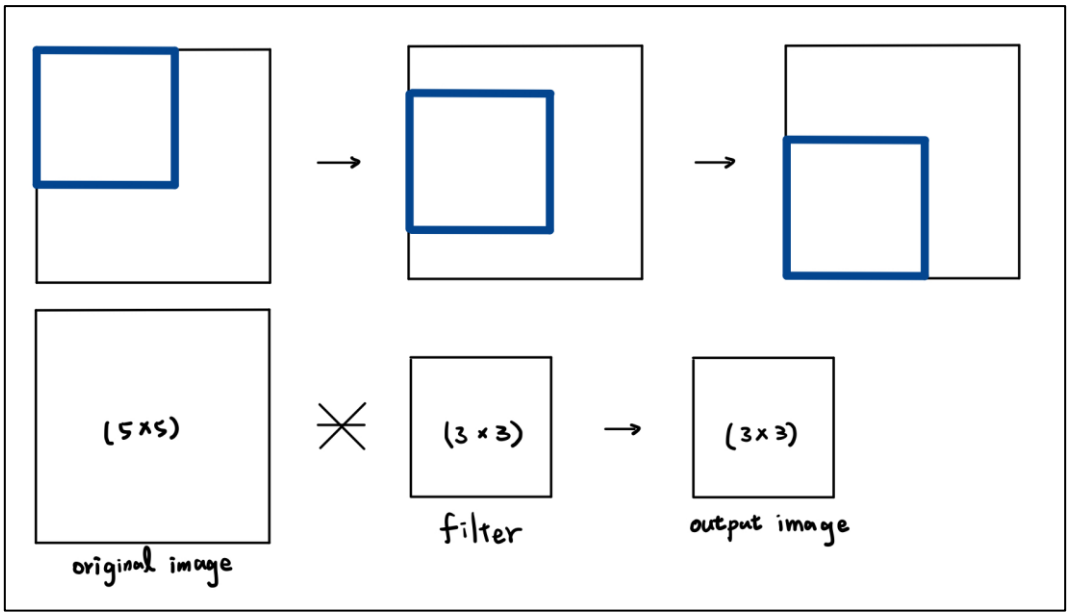

Fig.3 Image shrink

這裡就需要使用到 np.pad 的語法，透過先讓圖片填充多的長與寬，再做卷積如此便可達成卷積完的結果與原始照片尺寸相符，RGB 通道不需要填充的原因是因為他在卷積的過程不會因此尺寸縮水，但這會產生另一個問題那就是需要補多少的 0 才能不管濾波器的尺寸為何都能保持輸出結果不變，透過 Fig.3 的例子可以發現照片變小的原因是因為濾波器在掃描的過程因為本身濾波器的尺寸造成在原本應該掃五次的長寬只掃到三次，而這個是有規律的，取決於濾波器的矩陣長寬減一即為原始照片變小的長度與寬度，而這變小的數值即為 pad 需要多補上的長寬，我在程式裡的表示如下 Fig.4，我將濾波器的長寬減一後除二平均補再原圖片的上下左右(Fig.5)。

```
pad_height = int((filter_height - 1) / 2)
pad_width = int((filter_width - 1) / 2)
padded_image = np.pad(image, ((pad_height, pad_height), (pad_width, pad_width), (0, 0)), 'constant', constant_values=(0, 0))
```

Fig.4 pad height&width

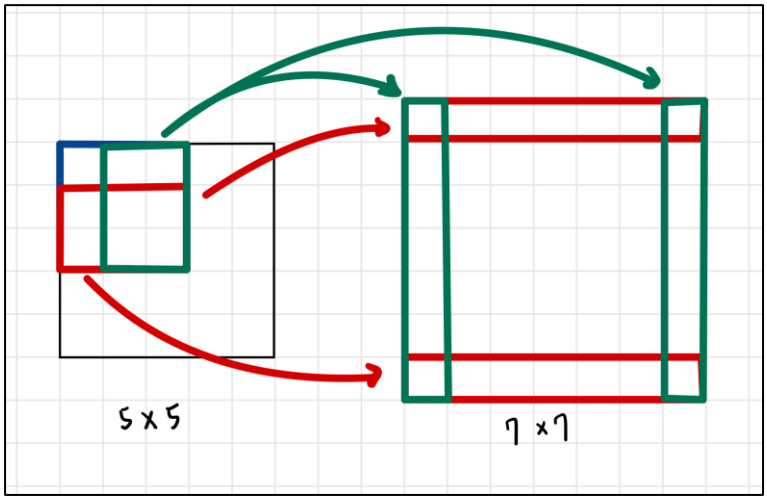

Fig.5 Add pad

1.2 Extract and combine the high-frequency and low-frequency signals.

Please finish the TODO in the file hw1.py.

```
# ============================================================================
# TODO: Use my_imfilter create 'low_frequencies' and
# 'high_frequencies' and then combine them to create 'hybrid_image'
# ============================================================================
# ============================================================================
# Remove the high frequencies from image1 by blurring it. The amount of
# blur that works best will vary with different image pairs
# ============================================================================
low_frequencies = my_imfilter(image1, gaussian_filter)
low_frequencies_2 = my_imfilter(image2, gaussian_filter)

# ============================================================================
# Remove the low frequencies from image2. The easiest way to do this is to
# subtract a blurred version of image2 from the original version of image2.
# This will give you an image centered at zero with negative values.
# ============================================================================
high_frequencies = image2 - low_frequencies_2

# ============================================================================
# Combine the high frequencies and low frequencies
# ============================================================================
hybrid_image = high_frequencies + low_frequencies
```

Fig.6 low frequencies / high frequencies / hybrid image

Algorithm Design:

本題的目標是取出 image1 的低頻和 image2 的高頻再混和成一張照片，透過調整濾波器的大小找出適合的尺寸混和照片。首先，低頻照片的我使用了老師提供的 gaussian filter 放進我第一題寫的 my imfilter 中，產生 image1 和 image2 的低頻照片；第二部分要取得高頻的照片我是利用原始照片減去低頻照片的方法取得，最後，混和照片將高頻與低頻相加。

1.3 Others

Please list the additional packages and versions required in your implementation and describe how to run your code. (make sure we can run your code)

1. Numpy 1.21.5
2. Opencv-python(cv2) 4.8.0.74
3. Matplotlib 3.5.3

附件只繳交其中一份 hybrid 的程式檔(111063517_hw1.py)，要執行其他照片直接改照片名稱和 cutoff frequency 即可。

2. Experiments

2.1 Hybrid Image

Put your hybrid result from the cat-dog pair and briefly explain your result.

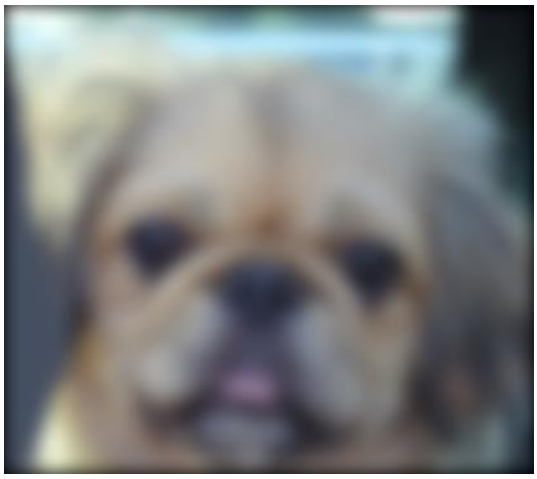

Fig.7 low frequency dog

Fig.8 High frequency cat

Fig.9 Hybrid image

Fig.7~9 是由低頻的狗照片和高頻的貓照片和 gaussian filter 的 cutoff frequency=7(filter size(29x29))混和出的照片，可以發現混和效果是不錯。如果將貓、狗的高低頻互換(Fig.10~12)，也可以在同樣的濾波器下得到不錯的混合效果。

Fig.10 low_frequency_dog

Fig.11 High_frequency_cat

Fig.12 Hybrid image

2.2 Other hybrid images

Try different pairs of pictures in the folder /data and put your results here.
Comparing the result of Problem 2.1, what's the difference?

在貓和狗的照片不管高低頻是哪一張都可以套用 Cutoff frequency=7 的濾波器，但在其他照片上就不一定適用，例如當高頻照片太突出的話，必須縮小濾波器的大小，降低 Cutoff frequency 的值，sigma 的的調整也會影響濾波器的效益，以下是其他 data 調整前與調整後的差異。

I. Einstein & Marilyn

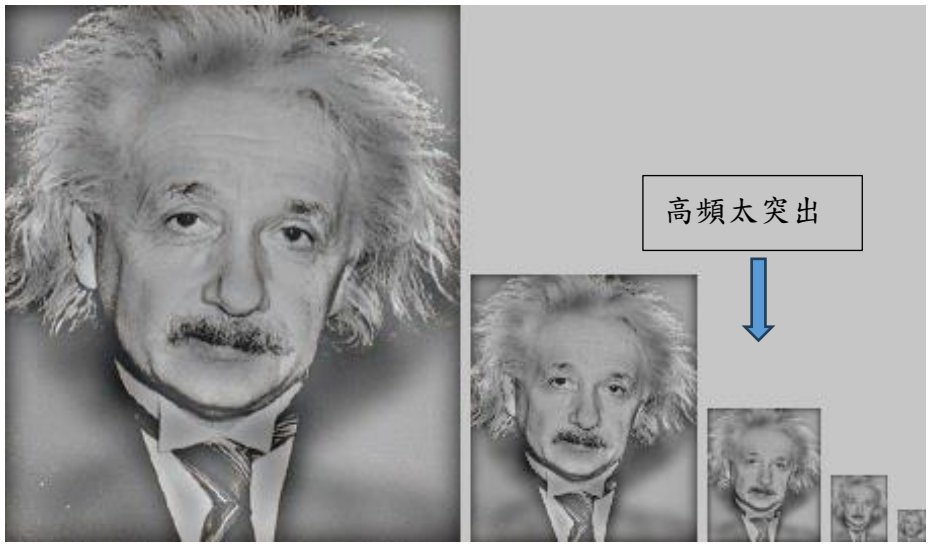

Fig.13 Cutoff frequency = 7

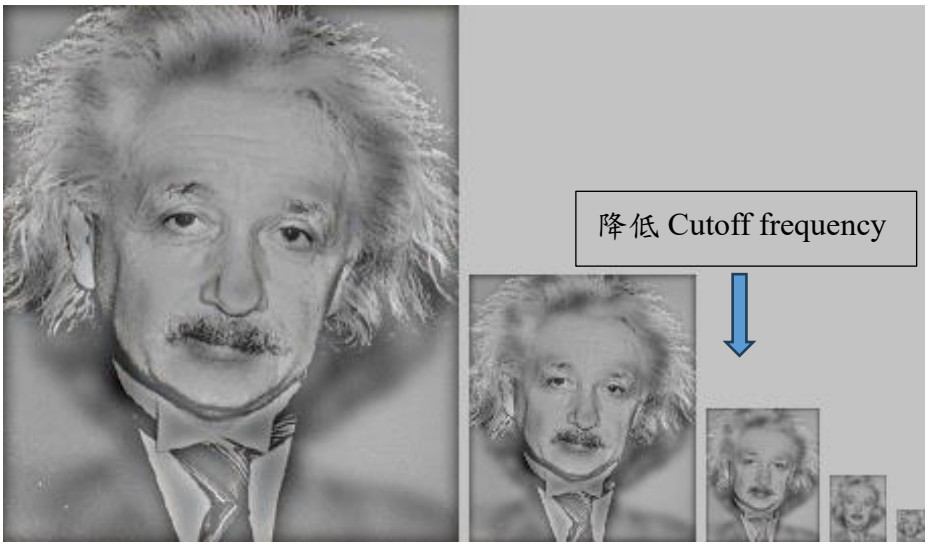

Fig.14 Cutoff frequency = 4.5

II. Bicycle & motorcycle

Fig.15 Cutoff frequency = 7

Fig.16 Cutoff frequency = 4

III. Bird & plain

Fig.17 Cutoff frequency = 7

Fig.18 Cutoff frequency = 5

IV. Fish & submarine

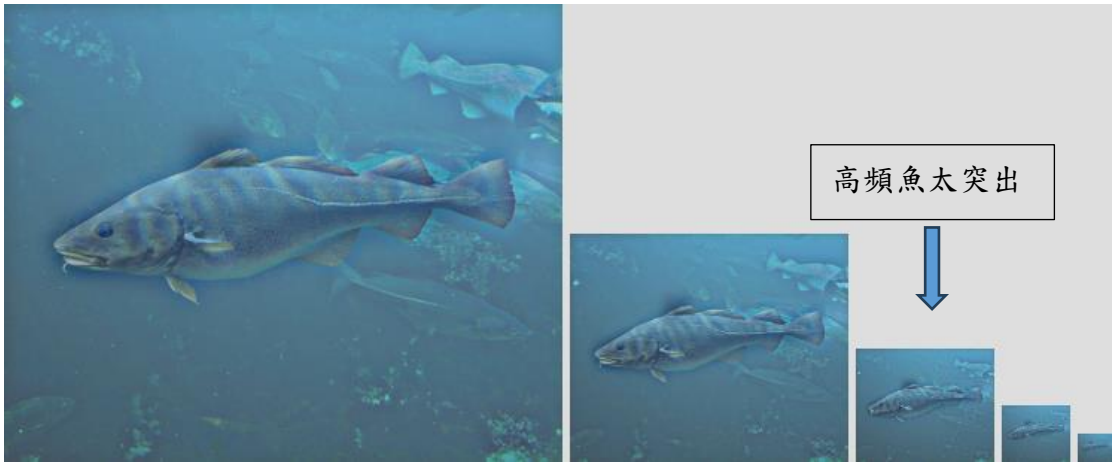

Fig.19 Cutoff frequency = 7

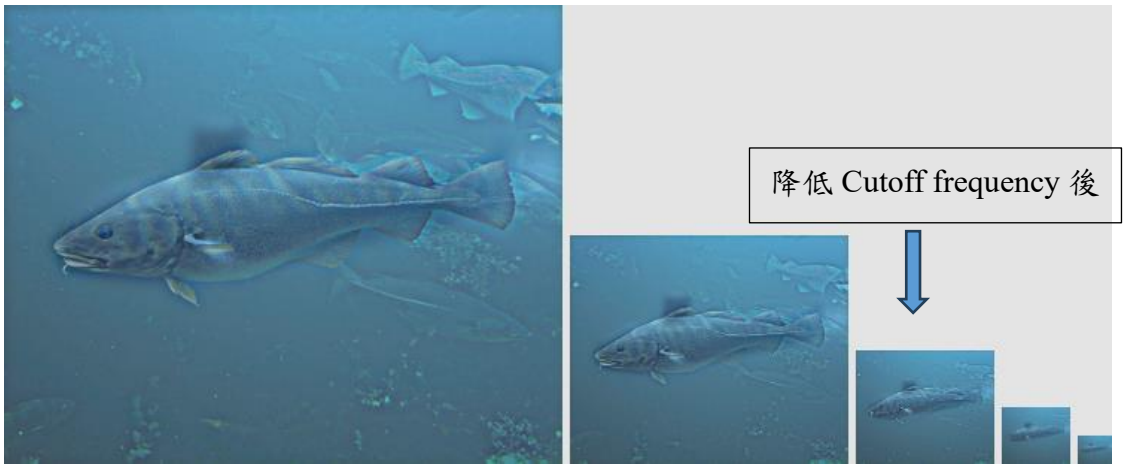

Fig.20 Cutoff frequency = 4

2.3 Customized hybrid images

Gather your own picture pairs and show your results of hybrid results. Briefly explain the difference between customized results and results from Problem 2.1 and 2.2.

Fig.23 Original Fig.24 Original

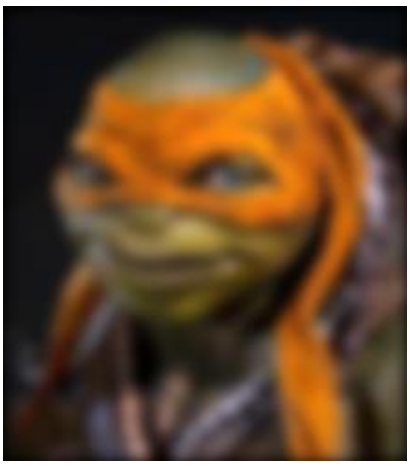

Fig.23 low frequency

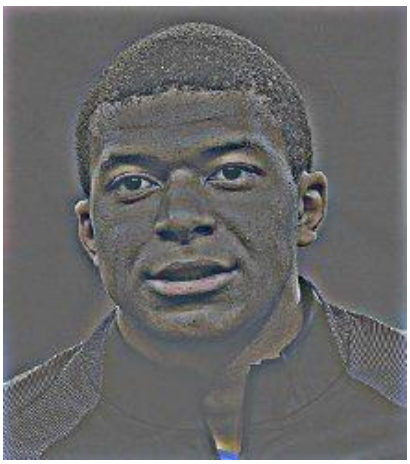

Fig.24 High frequency

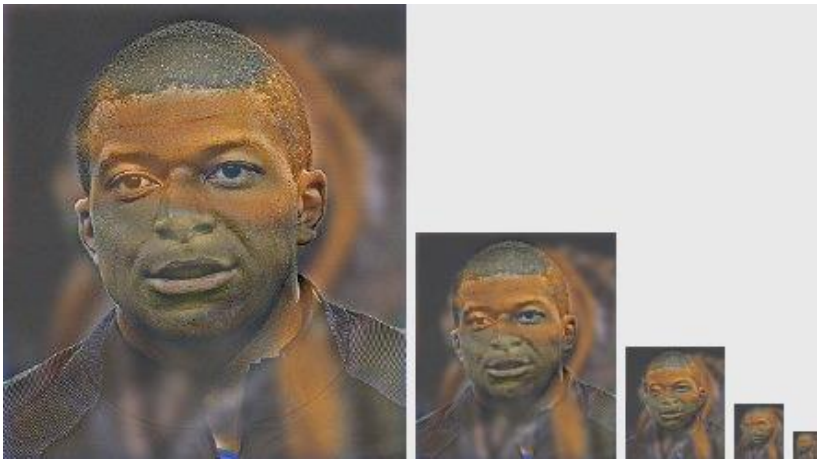

Fig.25 Hybrid image(cut off frequency = 4)

相比於 2.1 和 2.2 使用老師準備好的圖片合成，自己找的照片會遇到合成起來不是那麼契合的問題，例如人像的合成，必須找到眼睛可以對齊的照片，像是範例中的 Einstein & Marilyn，否則會出現 4 個眼睛的問題，如 Fig.26，另外我也發現了如果要處理人像通常濾波器的尺寸都要調小一點，否則高頻的線條會太明顯，在縮小後的照片上還是只有高頻。

Fig.26 尼克星與實驗室學長

3. Discussion

Do you discover anything special in your experimental results? What applications do you think this technology can be used for? Anything you discover while working on your homework.

這次的作業讓我深入了解了低頻和高頻影像合成的技術，並如何通過它們來創建混合影像。透過低頻濾波器處理，我們能夠使圖像變得模糊和更加柔和，這種技術的初步理解讓我知道到了許多現代圖像編輯軟件中的基本功能是如何實現的。舉例來說，要實現圖像的模糊效果，實際上就是轉換圖像為低頻成分的過程。

除此之外，高頻影像的概念也讓我深入探討了影像處理中的細節和輪廓。透過計算原始圖像減去低頻成分圖像，我們可以獲得高頻細節，這使得圖像的邊緣和細節更加突出。這個過程實際上運用了我之前在學習數位信號處理（DSP）時所掌握的理論知識，我能夠將這些理論應用到實際情境中。這些技術運用不只在人像修圖、電影特效，甚至是醫學影像都有使用的機會。

因為實驗室是進行有關 CNN 影像處理相關的研究，我們通常使用 TensorFlow 內建的 API 來訓練模型。這些 API 使得對圖片進行卷積等操作變得非常簡單，通常只需一行程式碼就可以完成。然而這個作業，要求我們不使用現有的庫(library)，而是自己實現圖像處理和卷積的功能。這使我們更深入地理解了圖片處理的過程，因為我們需要逐層處理圖像並進行卷積操作。特別是在寫"my_imfilter"時，我們更加深刻地理解了如何對圖像進行逐層處理以及如何實現卷積操作。