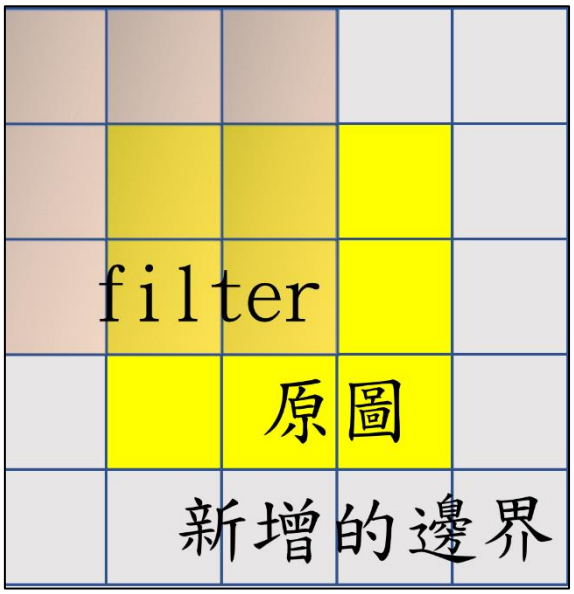
Homework #1 Hybrid Images

1 Implementation (45%)

1.1 Image filtering (20%)

在 my_filter.py 中，首先讀取圖片，再將原圖的四個邊各擴展 int(imfilter.shape[0]/2-0.5)個零，也就是為之後 filter (or convolution)預先增加邊界的 pixel，如（圖一），如此一來，filter 過後的圖才會原圖大小相同。



（圖一）原圖、邊界與 filter

再逐原圖的 pixel 將每塊 filter 大小的 pixel 數值與 filter 的數值相乘並總和，依序填入原圖大小的新圖中，即為 filter 過後的圖，我是使用 for 迴圈來完成，所以當 filter 的 shape 較大時，會需要更多的計算，也就為更久的運算時間。

1.3 Others (5%)

以下是我所使用的函式庫與其對應的版本和用途：

I、 numpy = 1.25.2
用來產生指定形狀的零矩陣與處理 float。

II、 cv2 = 4.6.0
用來讀取 image。

III、 matplotlib = 3.7.2
用來輸出結果圖片。

程式運行的方法是直接執行 hw1 即可，但可以在 if __name__ == '__main__': 中選擇要運行哪些 image pairs，我共嘗試了 7 對，列表如下：

I、 main()：
貓和狗，cutoff_frequency 預設為 7。

II、 celebrities(cutoff_frequency)：
愛因斯坦和瑪麗蓮夢露，cutoff_frequency 我使用 3。

III、 flying_objects(cutoff_frequency)：
鳥和戰鬥機，cutoff_frequency 我使用 5。

IV、 ocean(cutoff_frequency)：
魚和潛水艇，cutoff_frequency 我使用 5。

V、 two_wheels(cutoff_frequency)：
腳踏車和機車，cutoff_frequency 我使用 5。

VI、 dragonball (cutoff_frequency)：
特南克斯和悟天，cutoff_frequency 我使用 8。

VII、 classmate(cutoff_frequency)：
王清翰和林政旭，cutoff_frequency 我使用 4。

2 Experiments (30%)

2.1 Hybrid Image (10%)

（圖二）high_frequenciy cat

（圖三）low_frequenciy dog

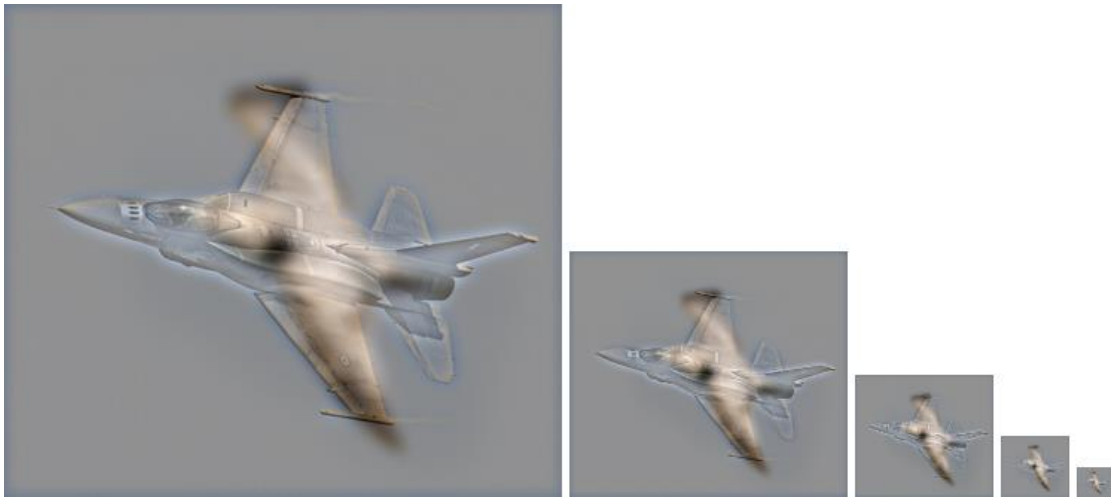
（圖四）hybrid_image

（圖五）hybrid_image_scales

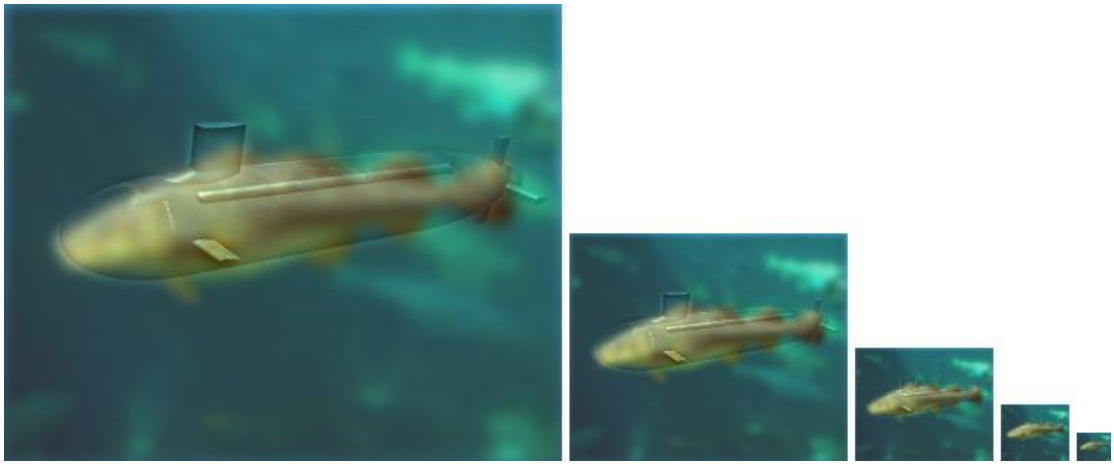
這部分將貓的圖擷取高頻，狗的圖則是擷取低頻，再將兩圖混合在一起，如此便創造出近看是貓，但遠看(或瞇眼看、縮小看)是狗的效果。

2.2 Other hybrid images (10%)

（圖六）hybrid_celebrities_scales



（圖七）hybrid_flying_scales



（圖八）hybrid_ocean_scales

（圖九）hybrid_two_wheels_scales

在對/data 中的其他 image pairs 測試時，會發現如果也和 2.1 的貓和狗一樣使用 cutoff_frequency 為 7，那會有其中幾對的混合效果非常難以辨識，所以 cutoff_frequency 需要針對不同的 image pair 做微調。

2.3 Customized hybrid images (10%)

（圖十）hybrid_dragonball_scales

（圖十一）hybrid_classmate_scales

上兩圖為我自己蒐集測試的 hybrid image，（圖十）是動畫七龍珠中的兩位角色，（圖十一）則是我的兩位同學。在測試時發現，混合的高低頻圖片需要找到一樣的解析度，並且確保輪廓位置大致相同，才會得到較佳的混合效果。

3 Discussion (25%)

I、 經過幾次調整 cutoff_frequency 的實驗後，我發覺同一對 image pairs 使用不同的 cutoff_frequency 會造成不一樣的視覺效果，而各對 image pairs 所適合的 cutoff_frequency 也不盡相同。

II、 這項技術與現今主流社群軟體的濾鏡效果類似，但因為這次作業是透過 for 迴圈做計算，所以運算速度實在太慢，不符合社群軟體中隨拍即 filter 的效果，可能使用其他運算方式才可達成。

III、 在做作業時，除了運算速度的問題以外，還發現 hybrid_image 的圖因為需要將高低頻相加，就可能導致矩陣中的某些值超過 1，所以需要先 normalize 才能使用 visualize_hybrid_image.py 輸出圖片。