HOMEWORK 1A
ECE/CS 8690 2302 Computer Vision

Hybrid Images in Python

Xuanbo Miao

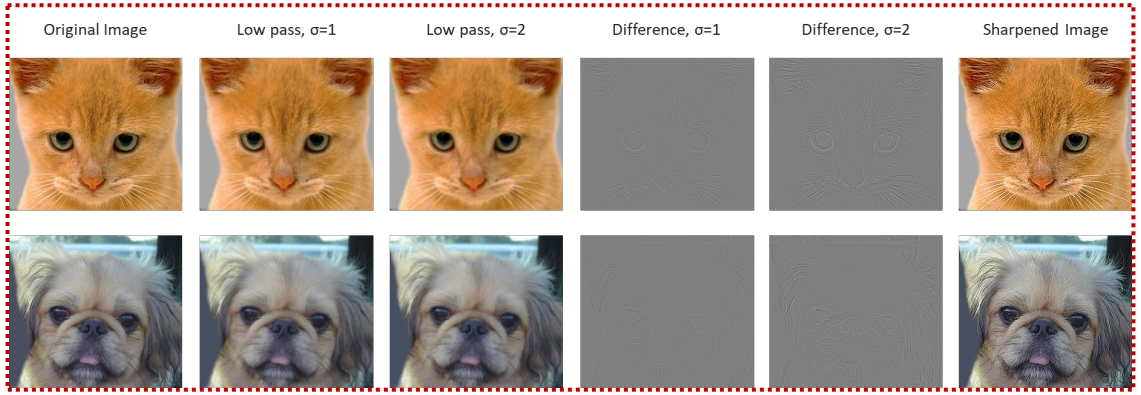
14422044

xmiao@mail.missouri.edu

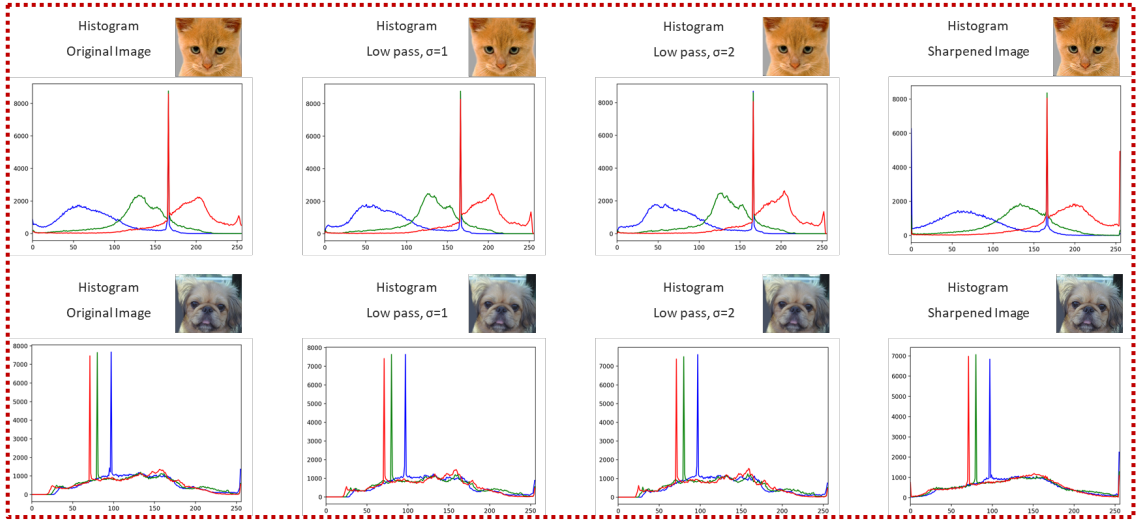
Abstract-what I made:

- Coding with python and cv2;
- Visualizing a couple of images;
- Low pass filter, difference, and sharpened images;
- Implementing of two Gaussian filter with different σ;
- Histogram of those images (R, G, and B);
- Image hybrid with different σ selection and images.

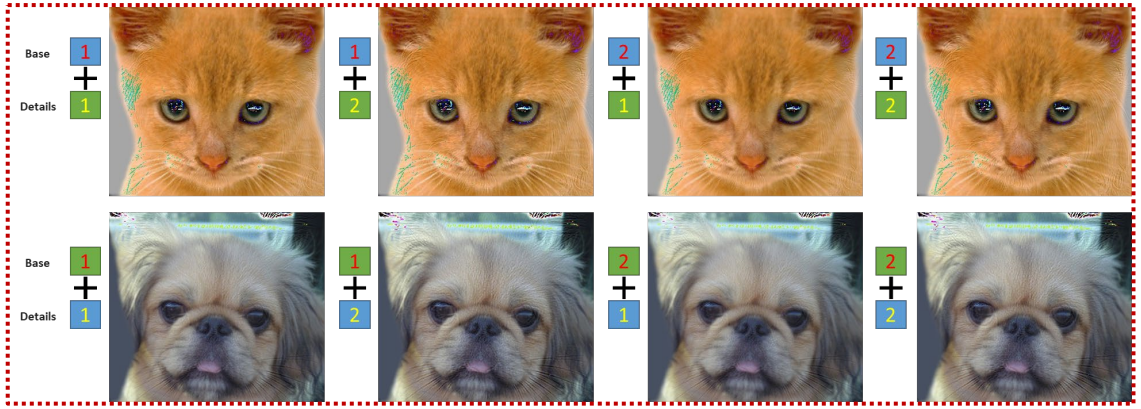
Section 1



Section 2



Section 3



Introduction & My Code

Here we get two input images. ①We are going to implement low pass filter, show difference between original image and filtered image, and show sharpened image. ②After that, RGB-channels histograms are drawn corresponding to all these input images and their varieties. ③Then we need to compute and display hybrid images for two sigma values. ④ All those images are needed to visualize.

ECE_8759_assignment1.py > ...

```python
import numpy as np
from datetime import datetime
import time
import matplotlib.pyplot as plt
import cv2

current_time = datetime.now().strftime("%H:%M:%S")

print("Current Time =", current_time)

img_cat = cv2.imread('./cat.jpg')
img_dog = cv2.imread('./dog.jpg')

if 1:
    kernel_gs_sig1_ = np.array([[0.0038,    0.0150, 0.0238, 0.0150, 0.0038],[0.0150,    0.0599, 0.0949, 0.0599, 0.0150],[0.0238,
0.0949, 0.1503, 0.0949, 0.0238],[0.0150,    0.0599, 0.0949, 0.0599, 0.0150],[0.0038,    0.0150, 0.0238, 0.0150, 0.0038]])
    kernel_gs_sig1 = kernel_gs_sig1_/sum(sum(kernel_gs_sig1_))
    kernel_gs_sig2_ = np.array([[0.0235,    0.0340, 0.0384, 0.0340, 0.0235],[0.0340,    0.0490, 0.0554, 0.0490, 0.0340],[0.0384,
0.0554, 0.0627, 0.0554, 0.0384],[0.0340,    0.0490, 0.0554, 0.0490, 0.0340],[0.0235,    0.0340, 0.0384, 0.0340, 0.0235]])
    kernel_gs_sig2 = kernel_gs_sig2_/sum(sum(kernel_gs_sig2_))

kernel_shar_ =  np.array([[0, -1, 0],[-1, 5,-1],[0, -1, 0]])
kernel_shar = kernel_shar_/sum(sum(kernel_shar_))

img_cat_soft1 = cv2.filter2D(img_cat,-1, kernel_gs_sig1)
img_cat_soft2 = cv2.filter2D(img_cat,-1, kernel_gs_sig2)
img_dog_soft1 = cv2.filter2D(img_dog,-1, kernel_gs_sig1)
img_dog_soft2 = cv2.filter2D(img_dog,-1, kernel_gs_sig2)

img_cat_diff1 = 128+img_cat_soft1-img_cat
img_cat_diff2 = 128+img_cat_soft2-img_cat
img_dog_diff1 = 128+img_dog_soft1-img_dog
img_dog_diff2 = 128+img_dog_soft2-img_dog

img_cat_shar = cv2.filter2D(img_cat,-1, kernel_shar)
img_dog_shar = cv2.filter2D(img_dog,-1, kernel_shar)

img_cat_dog11=cv2.normalize(img_cat_soft1+img_dog_diff1-128, None, 0, 255, cv2.NORM_MINMAX)
img_cat_dog12=cv2.normalize(img_cat_soft1+img_dog_diff2-128, None, 0, 255, cv2.NORM_MINMAX)
img_cat_dog21=cv2.normalize(img_cat_soft2+img_dog_diff1-128, None, 0, 255, cv2.NORM_MINMAX)
img_cat_dog22=cv2.normalize(img_cat_soft2+img_dog_diff2-128, None, 0, 255, cv2.NORM_MINMAX)

img_dog_cat11=cv2.normalize(img_dog_soft1+img_cat_diff1-128, None, 0, 255, cv2.NORM_MINMAX)
img_dog_cat12=cv2.normalize(img_dog_soft1+img_cat_diff2-128, None, 0, 255, cv2.NORM_MINMAX)
img_dog_cat21=cv2.normalize(img_dog_soft2+img_cat_diff1-128, None, 0, 255, cv2.NORM_MINMAX)
img_dog_cat22=cv2.normalize(img_dog_soft2+img_cat_diff2-128, None, 0, 255, cv2.NORM_MINMAX)
```

①

②

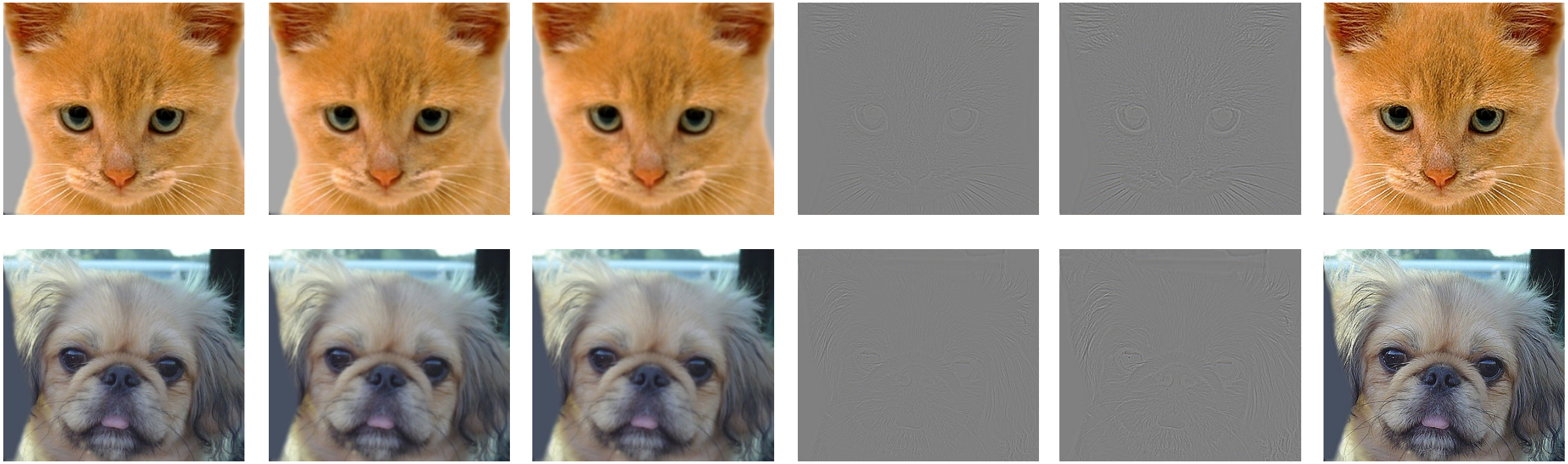
```python
for i, col in enumerate(['b', 'g', 'r']):
    hist_cat       = cv2.calcHist([img_cat],       [i], None, [256],  [0, 256])
    hist_dog       = cv2.calcHist([img_dog],       [i], None, [256],  [0, 256])

    hist_cat_soft1 = cv2.calcHist([img_cat_soft1], [i], None, [256],  [0, 256])
    hist_cat_soft2 = cv2.calcHist([img_cat_soft2], [i], None, [256],  [0, 256])
    hist_dog_soft1 = cv2.calcHist([img_dog_soft1], [i], None, [256],  [0, 256])
    hist_dog_soft2 = cv2.calcHist([img_dog_soft2], [i], None, [256],  [0, 256])

    hist_cat_shar  = cv2.calcHist([img_cat_shar],  [i], None, [256],  [0, 256])
    hist_dog_shar  = cv2.calcHist([img_dog_shar],  [i], None, [256],  [0, 256])

    plt.figure(1);   plt.plot(hist_cat     ,  color = col);  plt.xlim([0, 256])
    plt.figure(2);   plt.plot(hist_dog     ,  color = col);  plt.xlim([0, 256])
    plt.figure(3);   plt.plot(hist_cat_soft1, color = col);  plt.xlim([0, 256])
    plt.figure(4);   plt.plot(hist_cat_soft2, color = col);  plt.xlim([0, 256])
    plt.figure(5);   plt.plot(hist_dog_soft1, color = col);  plt.xlim([0, 256])
    plt.figure(6);   plt.plot(hist_dog_soft2, color = col);  plt.xlim([0, 256])
    plt.figure(7);   plt.plot(hist_cat_shar,  color = col);  plt.xlim([0, 256])
    plt.figure(8);   plt.plot(hist_dog_shar,  color = col);  plt.xlim([0, 256])

if 1:
    cv2.imshow('image_cat      ', img_cat);         cv2.resizeWindow('image_cat      ', 600, 600); cv2.imwrite('./img/image_cat
.jpg',img_cat      )
    cv2.imshow('image_dog      ', img_dog);         cv2.resizeWindow('image_dog      ', 600, 600); cv2.imwrite('./img/image_dog
.jpg',img_dog      )
    cv2.imshow('image_cat_soft1', img_cat_soft1);   cv2.resizeWindow('image_cat_soft1', 600, 600);
cv2.imwrite('./img/image_cat_soft1.jpg',img_cat_soft1)
    cv2.imshow('image_cat_soft2', img_cat_soft2);   cv2.resizeWindow('image_cat_soft2', 600, 600);
cv2.imwrite('./img/image_cat_soft2.jpg',img_cat_soft2)
    cv2.imshow('image_dog_soft1', img_dog_soft1);   cv2.resizeWindow('image_dog_soft1', 600, 600);
cv2.imwrite('./img/image_dog_soft1.jpg',img_dog_soft1)
    cv2.imshow('image_dog_soft2', img_dog_soft2);   cv2.resizeWindow('image_dog_soft2', 600, 600);
cv2.imwrite('./img/image_dog_soft2.jpg',img_dog_soft2)
    cv2.imshow('image_cat_diff1', img_cat_diff1);   cv2.resizeWindow('image_cat_diff1', 600, 600);
cv2.imwrite('./img/image_cat_diff1.jpg',img_cat_diff1)
    cv2.imshow('image_cat_diff2', img_cat_diff2);   cv2.resizeWindow('image_cat_diff2', 600, 600);
cv2.imwrite('./img/image_cat_diff2.jpg',img_cat_diff2)
    cv2.imshow('image_dog_diff1', img_dog_diff1);   cv2.resizeWindow('image_dog_diff1', 600, 600);
cv2.imwrite('./img/image_dog_diff1.jpg',img_dog_diff1)
    cv2.imshow('image_dog_diff2', img_dog_diff2);   cv2.resizeWindow('image_dog_diff2', 600, 600);
cv2.imwrite('./img/image_dog_diff2.jpg',img_dog_diff2)
    cv2.imshow('image_cat_shar ', img_cat_shar );   cv2.resizeWindow('image_cat_shar ', 600, 600); cv2.imwrite('./img/image_cat_shar
.jpg',img_cat_shar )
    cv2.imshow('image_dog_shar ', img_dog_shar );   cv2.resizeWindow('image_dog_shar ', 600, 600); cv2.imwrite('./img/image_dog_shar
.jpg',img_dog_shar )
    cv2.imshow('image_dog_cat11', img_dog_cat11);   cv2.resizeWindow('image_dog_cat11', 600, 600);
cv2.imwrite('./img/image_dog_cat11.jpg',img_dog_cat11)
    cv2.imshow('image_dog_cat12', img_dog_cat12);   cv2.resizeWindow('image_dog_cat12', 600, 600);
cv2.imwrite('./img/image_dog_cat12.jpg',img_dog_cat12)
    cv2.imshow('image_dog_cat21', img_dog_cat21);   cv2.resizeWindow('image_dog_cat21', 600, 600);
cv2.imwrite('./img/image_dog_cat21.jpg',img_dog_cat21)
    cv2.imshow('image_dog_cat22', img_dog_cat22);   cv2.resizeWindow('image_dog_cat22', 600, 600);
cv2.imwrite('./img/image_dog_cat22.jpg',img_dog_cat22)
    cv2.imshow('image_cat_dog11', img_cat_dog11);   cv2.resizeWindow('image_cat_dog11', 600, 600);
cv2.imwrite('./img/image_cat_dog11.jpg',img_cat_dog11)
    cv2.imshow('image_cat_dog12', img_cat_dog12);   cv2.resizeWindow('image_cat_dog12', 600, 600);
cv2.imwrite('./img/image_cat_dog12.jpg',img_cat_dog12)
    cv2.imshow('image_cat_dog21', img_cat_dog21);   cv2.resizeWindow('image_cat_dog21', 600, 600);
cv2.imwrite('./img/image_cat_dog21.jpg',img_cat_dog21)
    cv2.imshow('image_cat_dog22', img_cat_dog22);   cv2.resizeWindow('image_cat_dog22', 600, 600);
cv2.imwrite('./img/image_cat_dog22.jpg',img_cat_dog22)
    for i in range (1,8):   plt.figure(i);  plt.show()

cv2.waitKey(0)
```

③

④

Section 1. For both input images, generate and display: (a) low pass filtered (smooth) image; (b) difference between original image and smooth image; (c) sharpened image.

Original Image | Low pass, σ=1 | Low pass, σ=2 | Difference, σ=1 | Difference, σ=2 | Sharpened Image



```
img_cat_soft1 = cv2.filter2D(img_cat,-1, kernel_gs_sig1)
img_cat_soft2 = cv2.filter2D(img_cat,-1, kernel_gs_sig2)
img_dog_soft1 = cv2.filter2D(img_dog,-1, kernel_gs_sig1)
img_dog_soft2 = cv2.filter2D(img_dog,-1, kernel_gs_sig2)

img_cat_diff1 = 128+img_cat_soft1-img_cat
img_cat_diff2 = 128+img_cat_soft2-img_cat
img_dog_diff1 = 128+img_dog_soft1-img_dog
img_dog_diff2 = 128+img_dog_soft2-img_dog

img_cat_shar = cv2.filter2D(img_cat,-1, kernel_shar)
img_dog_shar = cv2.filter2D(img_dog,-1, kernel_shar)
```

Corresponding code for generating these images

Gaussian Kernel, σ=1

2D Kernel

0.0038,	0.0150,	0.0238,	0.0150,	0.0038,
0.0150,	0.0599,	0.0949,	0.0599,	0.0150,
0.0238,	0.0949,	0.1503,	0.0949,	0.0238,
0.0150,	0.0599,	0.0949,	0.0599,	0.0150,
0.0038,	0.0150,	0.0238,	0.0150,	0.0038,

Gaussian Kernel, σ=2

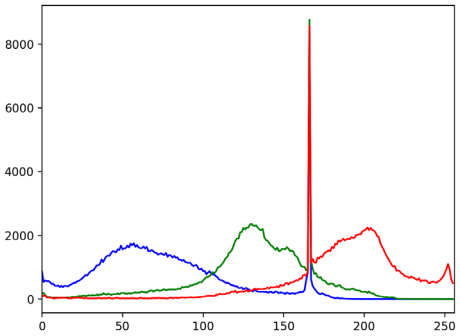
2D Kernel

0.0235,	0.0340,	0.0384,	0.0340,	0.0235,
0.0340,	0.0490,	0.0554,	0.0490,	0.0340,
0.0384,	0.0554,	0.0627,	0.0554,	0.0384,
0.0340,	0.0490,	0.0554,	0.0490,	0.0340,
0.0235,	0.0340,	0.0384,	0.0340,	0.0235,

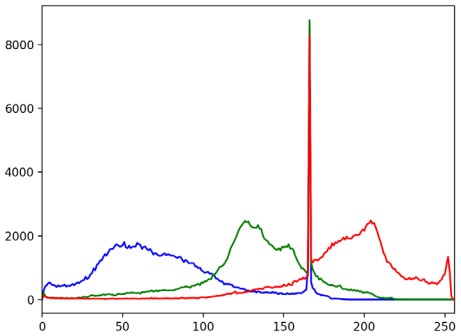
Section 2. For both input images and their smooth and sharpened versions, compute and plot red, green, blue channel histograms.

```
hist_cat       = cv2.calcHist([img_cat],       [i], None, [256], [0, 256])
hist_dog       = cv2.calcHist([img_dog],       [i], None, [256], [0, 256])

hist_cat_soft1 = cv2.calcHist([img_cat_soft1], [i], None, [256], [0, 256])
hist_cat_soft2 = cv2.calcHist([img_cat_soft2], [i], None, [256], [0, 256])
hist_dog_soft1 = cv2.calcHist([img_dog_soft1], [i], None, [256], [0, 256])
hist_dog_soft2 = cv2.calcHist([img_dog_soft2], [i], None, [256], [0, 256])

hist_cat_shar  = cv2.calcHist([img_cat_shar],  [i], None, [256], [0, 256])
hist_dog_shar  = cv2.calcHist([img_dog_shar],  [i], None, [256], [0, 256])
```

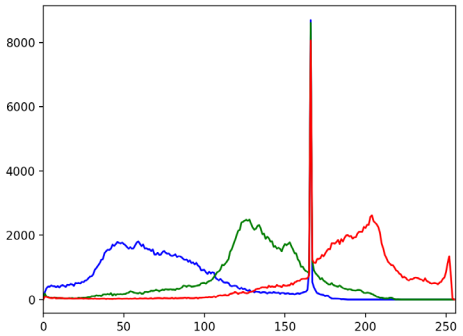
Histogram Original Image



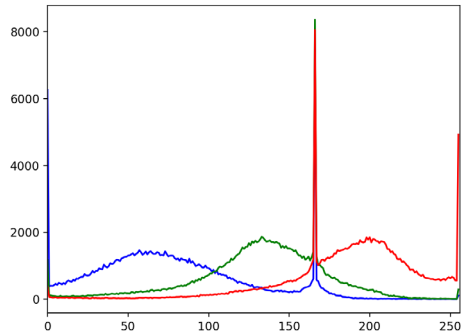
Histogram Low pass, σ=1



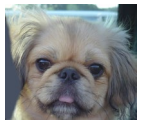
Histogram Low pass, σ=2

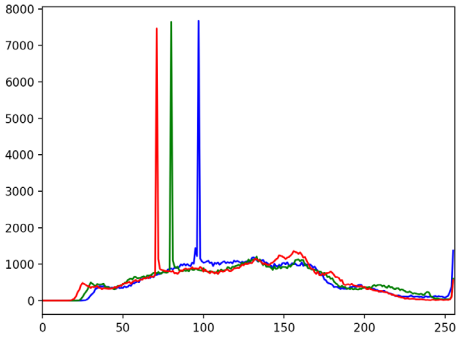


Histogram Sharpened Image

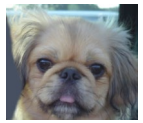


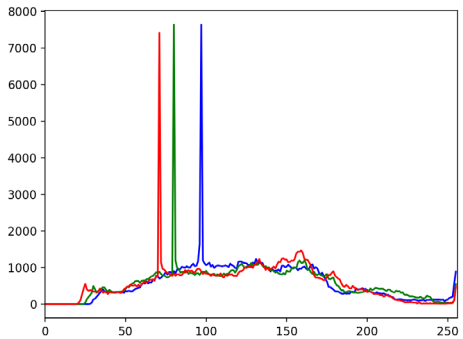
Histogram Original Image



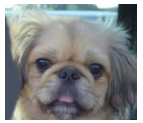


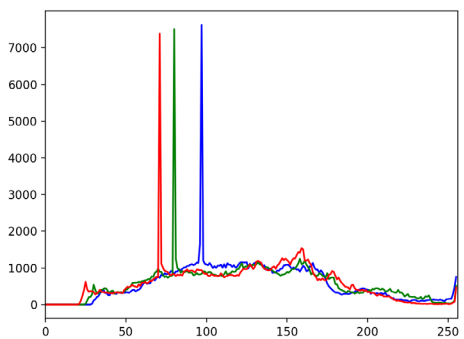
Histogram Low pass, σ=1



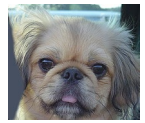


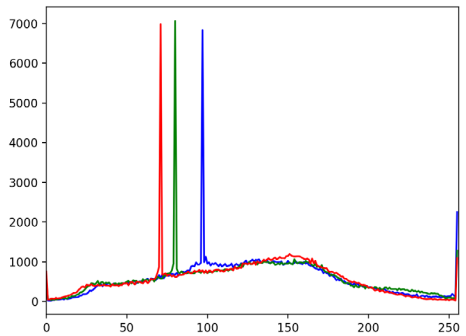
Histogram Low pass, σ=2



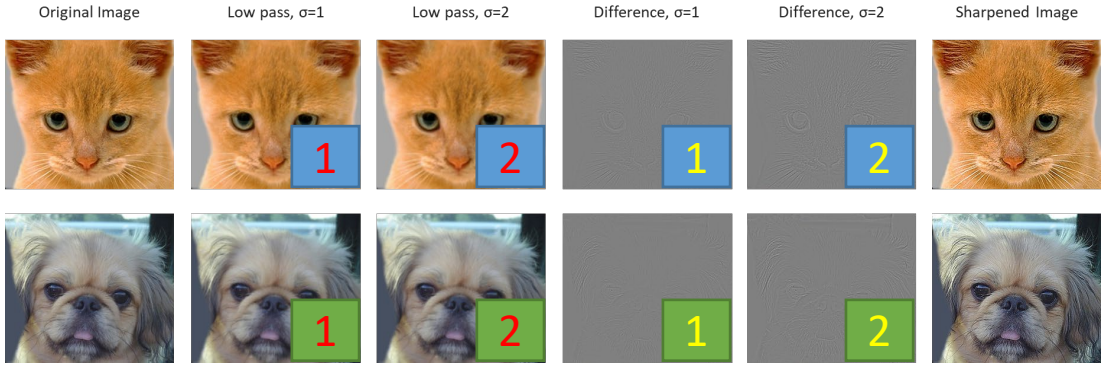


Histogram Sharpened Image

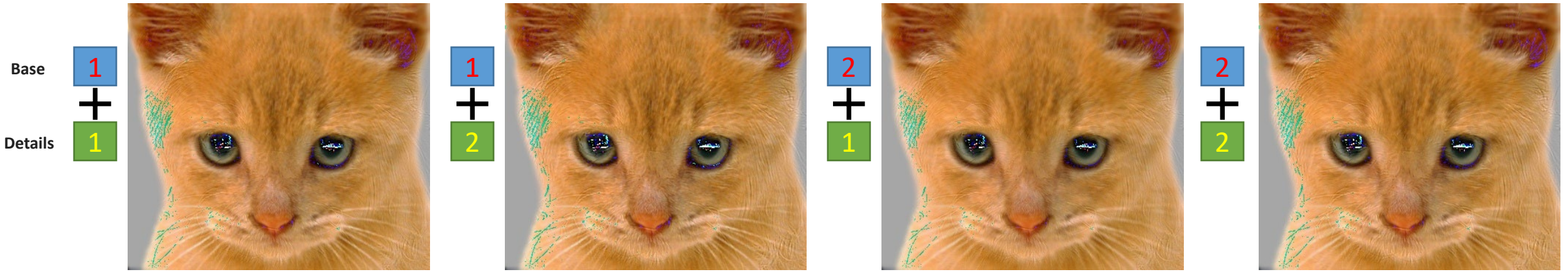


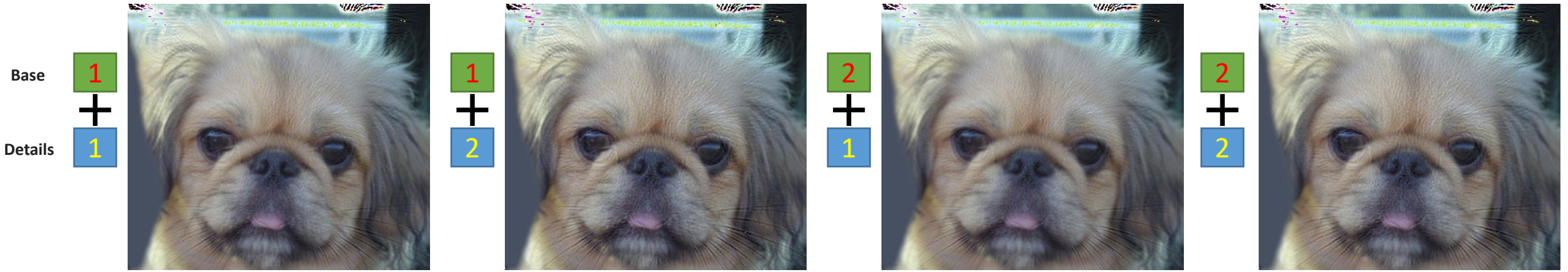


Section 3. Compute and display hybrid images for two sigma values.




```
img_cat_dog11=cv2.normalize(img_cat_soft1+img_dog_diff1-128, None, 0, 255, cv2.NORM_MINMAX)
img_cat_dog12=cv2.normalize(img_cat_soft1+img_dog_diff2-128, None, 0, 255, cv2.NORM_MINMAX)
img_cat_dog21=cv2.normalize(img_cat_soft2+img_dog_diff1-128, None, 0, 255, cv2.NORM_MINMAX)
img_cat_dog22=cv2.normalize(img_cat_soft2+img_dog_diff2-128, None, 0, 255, cv2.NORM_MINMAX)

img_dog_cat11=cv2.normalize(img_dog_soft1+img_cat_diff1-128, None, 0, 255, cv2.NORM_MINMAX)
img_dog_cat12=cv2.normalize(img_dog_soft1+img_cat_diff2-128, None, 0, 255, cv2.NORM_MINMAX)
img_dog_cat21=cv2.normalize(img_dog_soft2+img_cat_diff1-128, None, 0, 255, cv2.NORM_MINMAX)
img_dog_cat22=cv2.normalize(img_dog_soft2+img_cat_diff2-128, None, 0, 255, cv2.NORM_MINMAX)
```

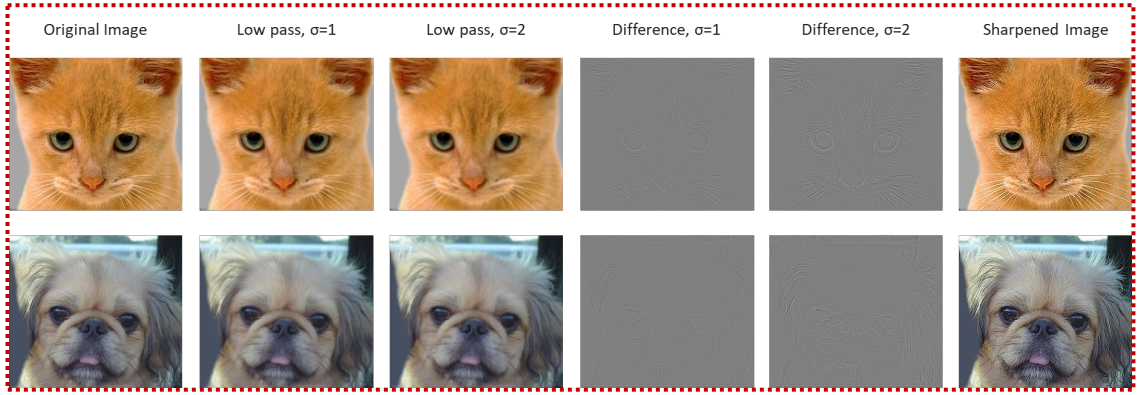




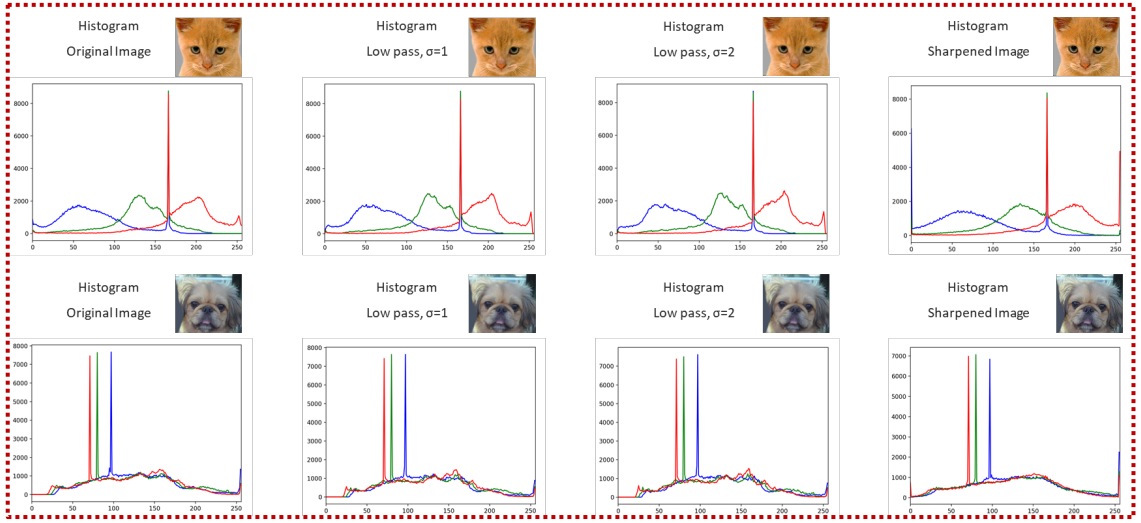
Discussion

① As shown in *Section 1*, we can see, low pass filter with higher σ will lead to more blurred images. This is also mean, if we calculate the difference of the original images and filtered images, more details will be shown on the difference images.
Actually, we get two ways to give a sharpened image. The first one is to add a difference map onto a original image. The second one is using a sharpening filter, like the one I used [0, -1, 0],[-1, 5,-1],[0, -1, 0]]. These two ways both can sharpen images.
② As shown in *Section 2*, the histogram is likely to be fitting well with the same corresponding original image. It worth to be noted, the details, are likely to be some kind of noise, and noise can actually flatten the histogram curves (a way to reduce color fault is by adding Gaussian noise). So, the histogram of low pass filtered images are not as flatten and smooth as the original images (and the detail added sharpened images).
③ As shown in *Section 2*, the hybrid images shows, the higher the σ is used on the difference image, more high frequency will be shown on the hybrid images; the lower the σ is used on the low pass base images, more overall graphic will be stayed on the hybrid images.

Section ①



Section ②



Section ③