Arthur Liou
NetID: arthurl3
CS445

MP1 - Hybrid Images

https://www.coursera.org/learn/cs-445/irt/ai9Ix/bells-and-whistles-claim-form-all-projects-1-5/step-1/submit/

Report Prompt

Use words and images to show us what you've done (the report doesn't need to be fancy). Please:

- Show us your favorite hybrid image result. Include:
  - 1) the original and filtered input images;
  - 2) the hybrid image; and
  - 3) the FFT images.
  - Briefly (a few sentences) explain how it works, using the included images as illustrations. Explain any clever ideas that you've incorporated and any parameters. This should be with your own images (not the included samples).
- Next, show us at least two more hybrid image results, including one that doesn't work so well (failure example). Briefly explain how you got the good results (e.g., chosen cut-off frequencies, alignment tricks, other techniques), as well as any difficulties and the possible reasons for the bad results. If you are so fortunate that everything that you try works well, try to figure out what shouldn't work.
- Do at least two of the image enhancement tasks, showing the original image, the resulting image(s), and explaining how the enhancement was done.
- Describe and bells and whistles under a separate heading.

Part 1: Hybrid Images

- Explanation:
  - Process
    - Using the “hybridImage” function, I used the cv2.GaussianBlur() function, which would have images passed in, a predefined size of (35, 35) as the numbers must be odd and positive, and the passed in standard deviation
    - The low pass image was created with the above function with the low cutoff as the standard deviation
    - I duplicated the same line for image 2 as in the following line I would generate the high pass image by subtracting image 2’s original image from its corresponding low pass image.
    - Hybrid image generated by adding the low pass image 1 and high pass image 2.

        - Done
    - Results: After defining the logic for this function, the results came from playing around with the cuff off frequencies (standard deviations) for both low and high. More details on the frequencies I used below.

<u>Images Descriptions:</u>

Image Pair 1 - Successful (But not Favorite ie FFT images)

For this image pair I set both the high- and low-frequency cutoff at 10.

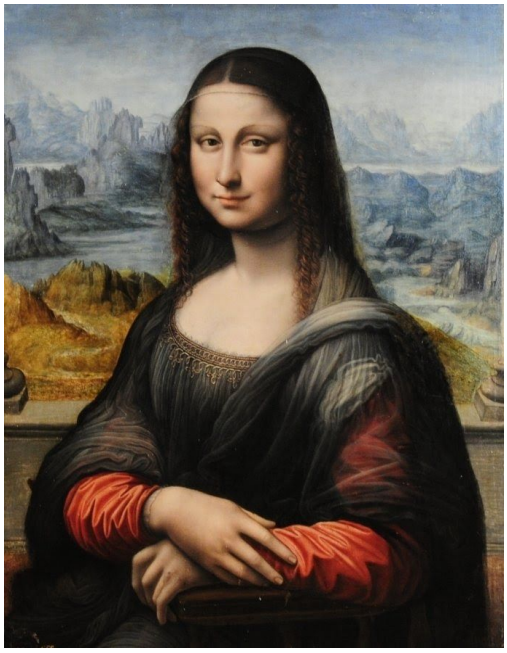
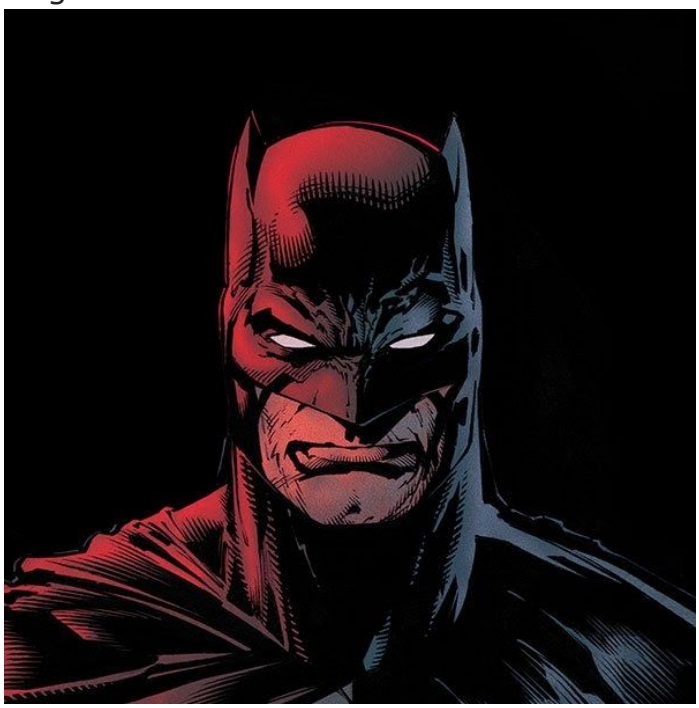
<table>
<tr>
<td>Original
  </td>
<td>Original
  </td>
</tr>
<tr>
<td>Filtered
Sample Image - Low Pass
 
Sample Image - High Pass
  </td>
<td>Hybrid
  </td>
</tr>
</table>

Image Pair 2 - Favorite, FFT Included

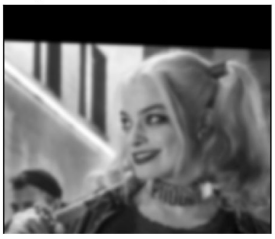
For this image pair I set both the low-frequency cutoff at 5, and high at 15.

Original

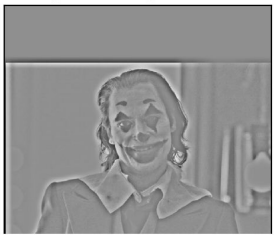
Original

Filtered

Harley Quinn Image - Low Pass



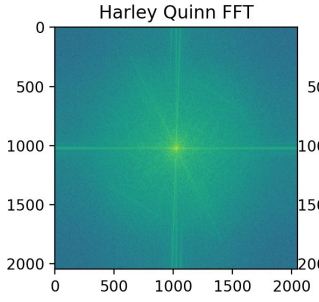
Joker - High Pass



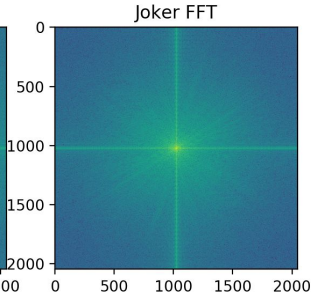
Hybrid

FFT for Original Images

Harley Quinn FFT

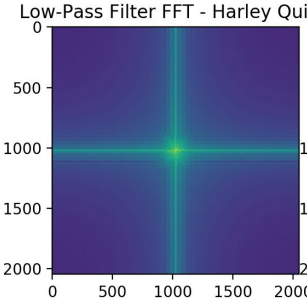


Joker FFT

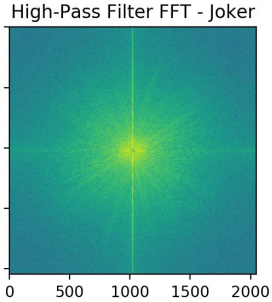


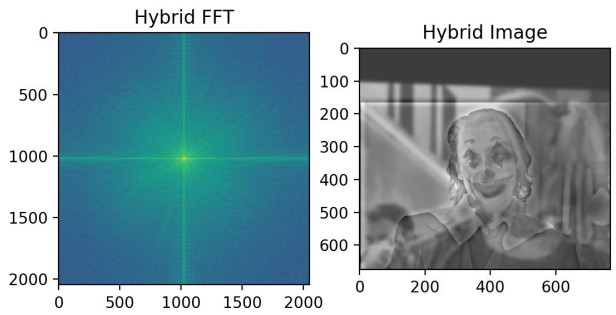
FFT for Low Pass and High Pass

Low-Pass Filter FFT - Harley Quinn



High-Pass Filter FFT - Joker





FFT for Hybrid Image & Hybrid Image

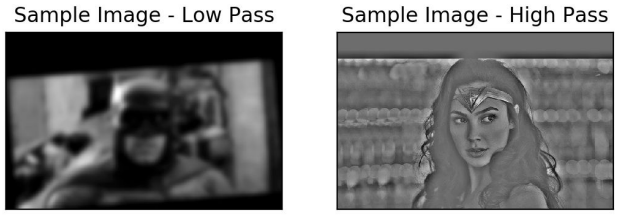
Image Pair 3 - Failure

For this image pair I set both the low- and high-frequency cutoff at 30

Original

Original

Filtered



Hybrid

Image Pair 4 - BounceDown

For this image pair I set both the low-frequency cutoff at 15, and high at 25.

Original	Original

Filtered

Aquaman - Low Pass

Deadpool - High Pass

Hybrid

Part 2: Image Enhancement Tasks

1. Contrast Enhancement: Choose an image (ideally one of yours, but from web is ok) that has poor contrast and fix the problem. Potential fixes include Laplacian filtering, gamma correction, and histogram equalization. Explain why you chose your solution.
    a. Two fixes tested + how it was done
        i. Gamma Correction
            - Convert original image to RGB
            - Divide the image pixels by 255 to get a (0, 1) image and use a gamma factor for the gamma correction effect.
            - Done but pay with different gamma factors
        ii. Histogram Equalization
            - Convert original image to RGB and then grayscale

- Run cv2.equalizeHist() on the grayscaled image
- Done

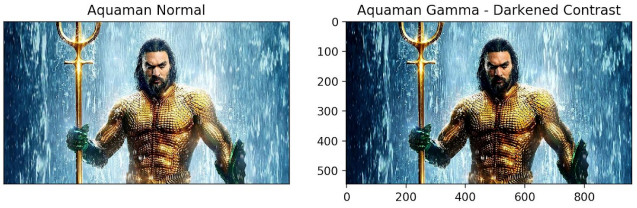
b. Explanation

i. I tried both gamma correction and histogram equalization. I much preferred gamma correction because, when attempting to fix the histogram equalization, I ran into a multitude of problems, which should've been easy to resolve, but many did not work, even with the help of the official documentation, Stack Overflow, and countlessly checking my logic for the cell. In my first try under Histogram Equalization, you can see that the output is not close to what one may expect. Thus, 9I finally resorted to just a conversion to YUV and changing the Y channel ("luma" component). In addition, I found the Gamma Correction contrast enhancement to be more effective, especially when I compared the original and enhanced image side by side for each

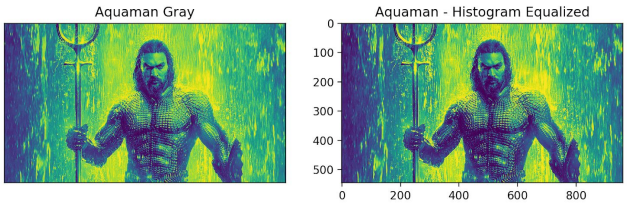
Images for Contrast Enhancement - both Gamma and Histogram

Original

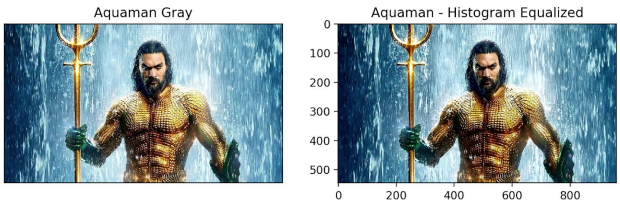
Gamma



Histogram Test 1



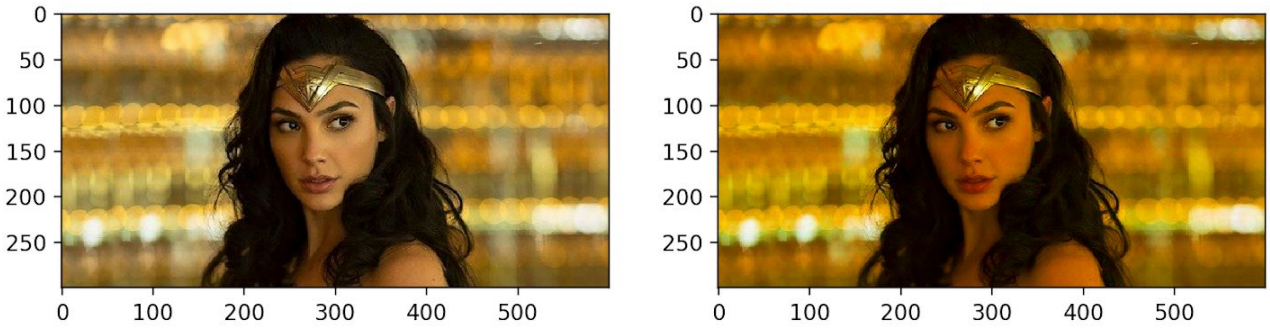
Histogram Test 2



2. Color Enhancement: Convert the images to HSV color space and divide into hue, saturation, and value channels (hsv = cv2.cvtColor(im,cv2.COLOR_BGR2HSV) in OpenCv). Then manipulate the appropriate channel(s) to make the colors (but not the intensity) brighter. Note that you want the values to map between the range defined by the imported library (in OpenCv 0-255), so you shouldn't just add or

multiply with some constant. Show this with at least one photograph. Show the original and enhanced images and explain your method.

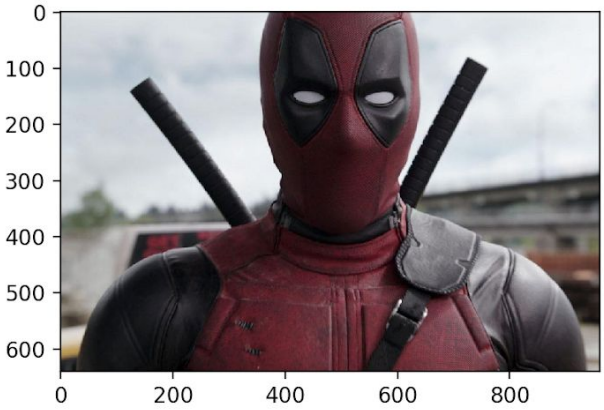
a. Color Enhancement
    i. Convert original image to HSV
    ii. Take out the saturation channel and multiply the values in that channel by a constant
    iii. Use a combination of np.clip() and astype to clip the values between 0-255 and perform pixel wrapping.
    iv. Convert back to RGB
    v. Done. Image color is enhanced.
        - A way to confirm is compare the skin tones
        - Original on the left & enhanced on the right



3. Color Shift: In OpenCv use cv2.cvtColor(image, cv2.COLOR_BGR2Lab) for converting between RGB and LAB spaces, in case you want to use LAB space.
    a. More Red
    b. Less Yellow
    c. Process for both
        i. Convert original image to LAB.
        ii. Depending on the request, more red or less yellow, grab the channel for that color, multiply those values for that channel with a constant.
        iii. Use a combination of np.clip() and astype to clip the values between 0-255 and perform pixel wrapping.
        iv. Convert LAB back to RGB
        v. Done
            - Red: Constant of 1.5 makes the image more pink
            - Yellow: Constant of 0.5 makes the image more blue

More Red

Less Yellow



Bells and Whistles

Points Claimed are in Bold

- **Try using color to enhance the effect of hybrid images. Does it work better to use color for the high-frequency component, the low-frequency component, or both? (5 pts)**
  - See below
- Illustrate the hybrid image process by implementing Gaussian and Laplacian pyramids and displaying them for your favorite result. This should look similar to Figure 7 in the Oliva et al. paper. (15 pts)
- **Do all three image enhancement tasks. (10 pts)**
  - See above for all three done

Color Enhanced Hybrid Image

- Color on High Frequency: Wow, works very well!
- Color on Low Frequency: Colors from the low pass pop and the high passed image really lurks underneath. Better than high frequency
- Color on Both: Great colors and apparent, but not a great as low frequency.
- Conclusion: While I find that all three are improvements over gray scale, out of the three colors, I find color on low frequency to be the best.

Color - High Frequency 	Color - Low Frequency 
Both Frequency 	

Expected Points Outline

The core assignment is worth 100 points, as follows:

- 45 points for implementation to create hybrid images from two aligned input images.
    - Completed. hybridImage and 4 sets of hybrid images created from two align input images
    - 45
- 25 points for illustration and additional results: 15 points for FFT images; 10 points for including at least two examples beyond the first (including at least one failure).
    - Completed
        - FFT Images on Image Set 2 - 15
        - 4 Sets total, including Image Set 3 being a failure - 10
    - 25
- 10 points for quality of results (e.g., 0=poor 5=average 10=great 15=amazing)
    - Completed. Each set was clearly done and had a well expected amazing success or a failure, like Image Set 3
    - 15
- 20 points for two image enhancement tasks (10 pts each), including explanation and display of results.
    - Completed. All three image enhancement tasks done, along with images displayed and explanations provided.
    - 20
- You can also earn up to 30 extra points for the bells & whistles mentioned above (5 for experimenting with color; 15 for Gaussian/Laplacian pyramids; 10 for third task of color enhancement) or suggest your own extensions (check with prof first).
    - Color experimenting - 5
    - 3rd Color Enhancement - 10

Regular: 45+25+15+20 = 100
Bells and Whistles: 15

Sources

- https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_filtering/py_filtering.html
- https://stackoverflow.com/questions/17190649/how-to-obtain-a-gaussian-filter-in-python
- https://docs.opencv.org/master/d1/dc5/tutorial_background_subtraction.html
- https://stackoverflow.com/questions/21425992/how-to-subtract-two-images-using-python-opencv2-to-get-the-foreground-object
- https://www.geeksforgeeks.org/addition-blending-images-using-opencv-python/
- https://docs.opencv.org/3.4/d3/dc1/tutorial_basic_linear_transform.html
- https://docs.opencv.org/master/d5/daf/tutorial_py_histogram_equalization.html

- https://www.programcreek.com/python/example/2716/numpy.uint8
- https://stackoverflow.com/questions/7547557/numpy-uint8-pixel-wrapping-solution
- https://www.packtpub.com/packtlib/book/Application-Development/9781785283932/2/ch02lvl1sec26/Enhancing%20the%20contrast%20in%20an%20image
- https://code.tutsplus.com/tutorials/histogram-equalization-in-python--cms-30202