Programming Project #1: Hybrid Images

CS445: Computational Photography - Spring 2020

Part I: Hybrid Images

In [1]:

```
import cv2

import numpy as np
from matplotlib.colors import LogNorm
from scipy import signal

import utils
```

In [2]:

```
%matplotlib notebook
import matplotlib.pyplot as plt
```

In [ ]:

```
'''
Lion: https://ichef.bbci.co.uk/news/410/cpsprodpb/1CE8/production/_
John Wick: https://i.dawn.com/large/2019/05/5ce52d8edc02a.jpg

'''
```

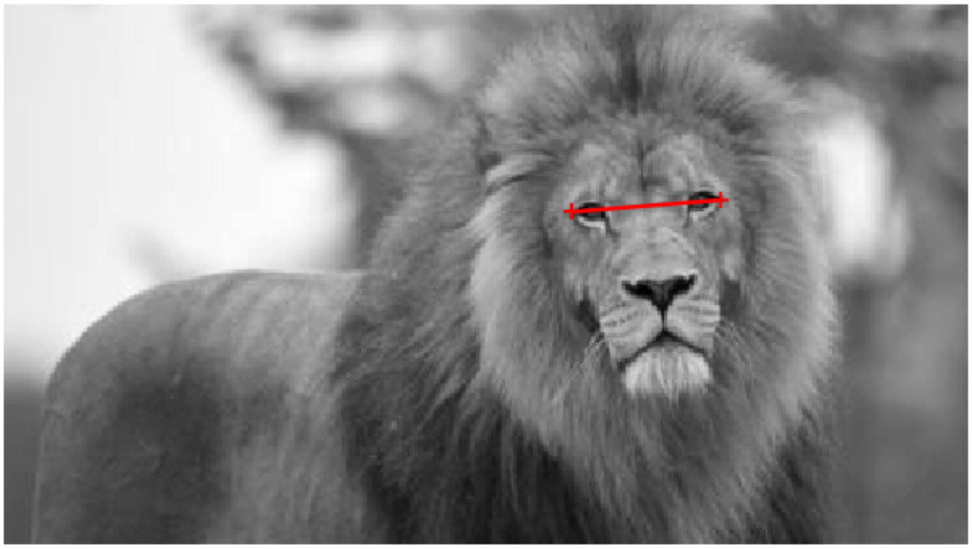
In [52]:

```
im1_file = 'lion.jpg'
im2_file = 'john_wick.jpg'

im1 = cv2.imread(im1_file, cv2.IMREAD_GRAYSCALE)
im2 = cv2.imread(im2_file, cv2.IMREAD_GRAYSCALE)
```

In [53]:

```
pts_im1 = utils.prompt_eye_selection(im1)
```

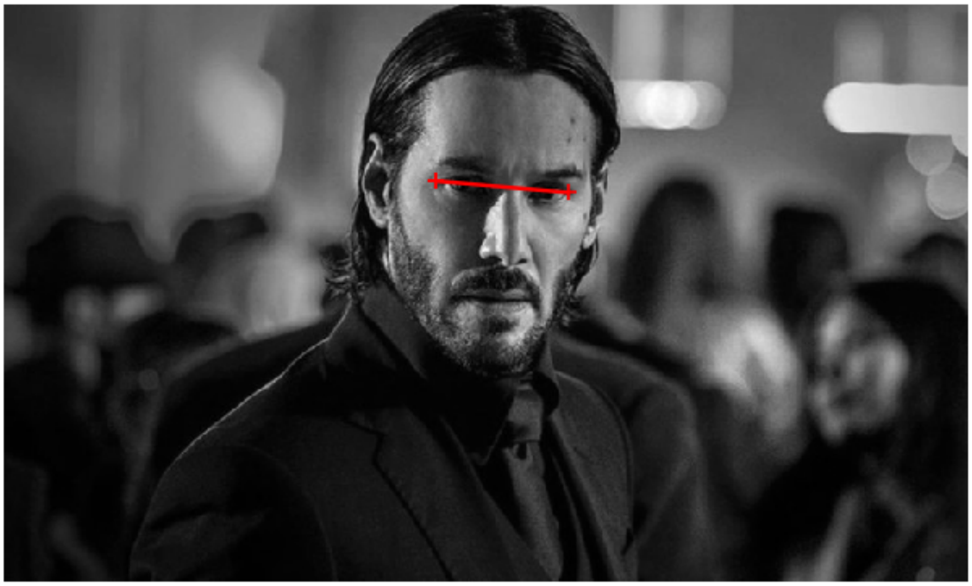
```
<IPython.core.display.Javascript object>
```



In [54]:

```
pts_im2 = utils.prompt_eye_selection(im2)
```

```
<IPython.core.display.Javascript object>
```



In [55]:

```
im1, im2 = utils.align_images(im1_file, im2_file,pts_im1,pts_im2,s
```

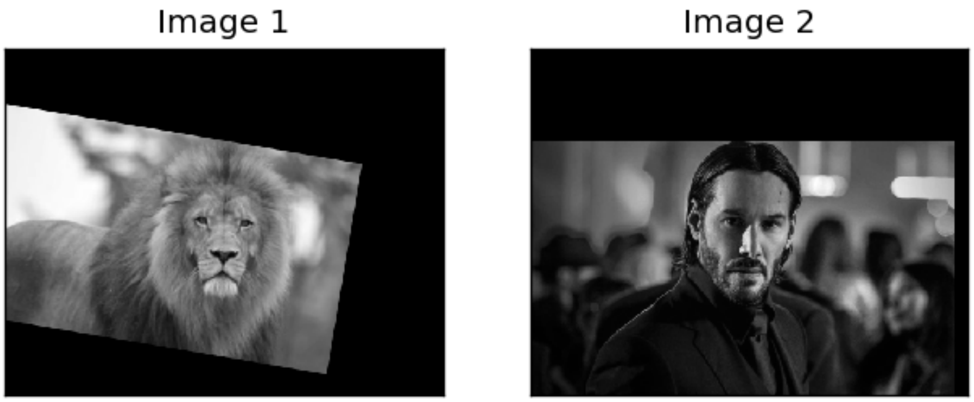
In [56]:

```
# convert to grayscale
im1 = cv2.cvtColor(im1, cv2.COLOR_BGR2GRAY) / 255.0
im2 = cv2.cvtColor(im2, cv2.COLOR_BGR2GRAY) / 255.0
```

In [57]:

```
#Images sanity check
fig, axes = plt.subplots(1, 2)
axes[0].imshow(im1,cmap='gray')
axes[0].set_title('Image 1'), axes[0].set_xticks([]), axes[0].set_
axes[1].imshow(im2,cmap='gray')
axes[1].set_title('Image 2'), axes[1].set_xticks([]), axes[1].set_
```

```
<IPython.core.display.Javascript object>
```



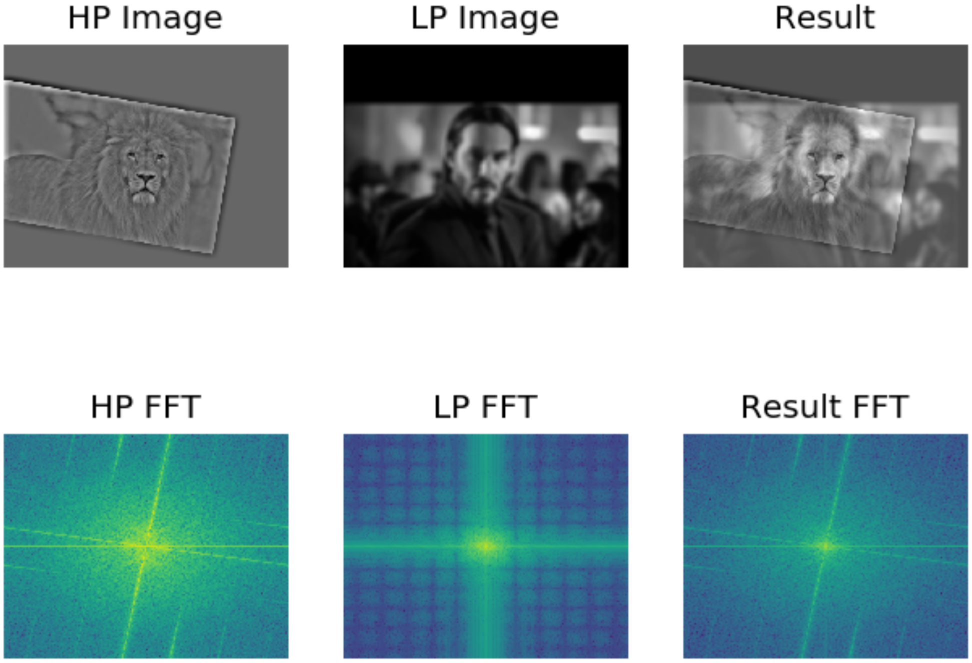
In [58]:

```
import numpy
def hybridImage(im1, im2, cutoff_low, cutoff_high):
    '''
    Inputs:
        im1:    RGB (height x width x 3) or a grayscale (height x
                as a numpy array.
        im2:    RGB (height x width x 3) or a grayscale (height x
                as a numpy array.
        cutoff_low: standard deviation for the low-pass filter
        cutoff_high: standard deviation for the high-pass filter

    Output:
        Return the combination of both images, one filtered with a
        and the other with a high-pass filter.
    '''

    filtImage1 = signal.convolve2d(im1, utils.gaussian_kernel(cuto
    filtImage2 = signal.convolve2d(im2, utils.gaussian_kernel(cuto

    croppingBounds1 = numpy.array([[cutoff_high, cutoff_high],[cut
    croppingBounds2 = numpy.array([[cutoff_low, cutoff_low],[cutof

    high_pass_filtered = cv2.subtract(im1, utils.crop_image(filtIm
    low_pass_filtered = utils.crop_image(filtImage2, croppingBound
    hybridImageResult = cv2.addWeighted(high_pass_filtered, 0.75,

    fig = plt.figure()

    plt.subplot(2, 3, 1)
    plt.imshow(high_pass_filtered, cmap='gray')
    plt.title("HP Image", fontsize=12)
    plt.axis('off')

    plt.subplot(2, 3, 2)
    plt.imshow(low_pass_filtered, cmap='gray')
    plt.title("LP Image", fontsize=12)
    plt.axis('off')

    plt.subplot(2, 3, 3)
    plt.imshow(hybridImageResult, cmap='gray')
    plt.title("Result", fontsize=12)
    plt.axis('off')

    plt.subplot(2, 3, 4)
    plt.imshow(numpy.log(numpy.abs(numpy.fft.fftshift(numpy.fft.ff
    plt.title("HP FFT", fontsize=12)
    plt.axis('off')

    plt.subplot(2, 3, 5)
    plt.imshow(numpy.log(numpy.abs(numpy.fft.fftshift(numpy.fft.ff
    plt.title("LP FFT", fontsize=12)
    plt.axis('off')

    plt.subplot(2, 3, 6)
    plt.imshow(numpy.log(numpy.abs(numpy.fft.fftshift(numpy.fft.ff
    plt.title("Result FFT", fontsize=12)
    plt.axis('off')

    plt.show()




    return hybridImageResult
```

In [69]:

```
arbitrary_value = 80  # you should choose meaningful values; you
cutoff_low = 5
cutoff_high = 10

im_hybrid = hybridImage(im1, im2, cutoff_low, cutoff_high)
```

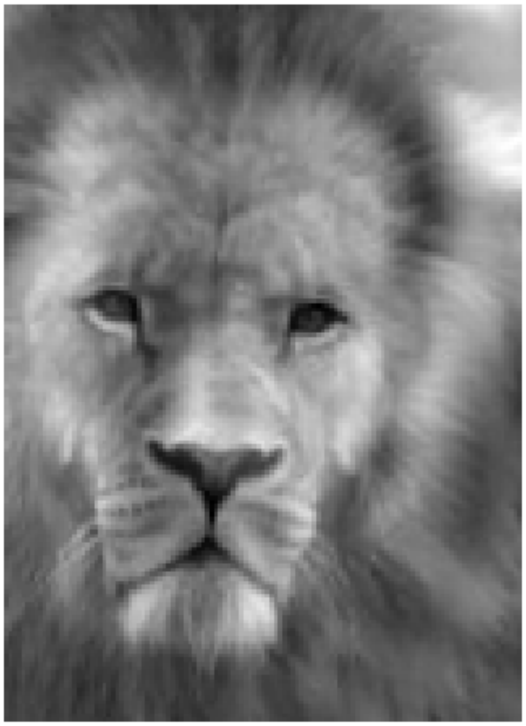
```
<IPython.core.display.Javascript object>
```



In [70]:

```
# Optional: Select top left corner and bottom right corner to crop
# the function returns dictionary of
# {
#   'cropped_image': np.ndarray of shape H x W
#   'crop_bound': np.ndarray of shape 2x2
# }
cropped_object = utils.interactive_crop(im_hybrid)

```

```
<IPython.core.display.Javascript object>
```



Part II: Image Enhancement

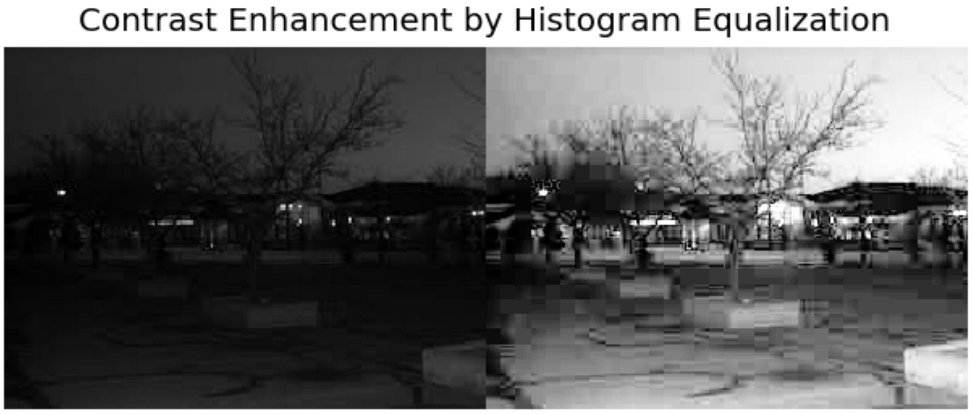
Two out of three types of image enhancement are required. Choose a good image to showcase each type and implement a method. This code doesn't rely on the hybrid image part.

Contrast enhancement

In [34]:

```
''';
Image location: data:image/jpeg;base64,/9j/4AAQSkZJRgABAQAAAQABAAD
'''

img = cv2.imread('part2_contrast.jpg', cv2.IMREAD_GRAYSCALE)
equ = cv2.equalizeHist(img)
res = np.hstack((img,equ)) #stacking images side-by-side
plt.imshow(res, cmap='gray')
plt.title("Contrast Enhancement by Histogram Equalization", fonts
plt.axis('off')
plt.show()
```

```
<IPython.core.display.Javascript object>
```

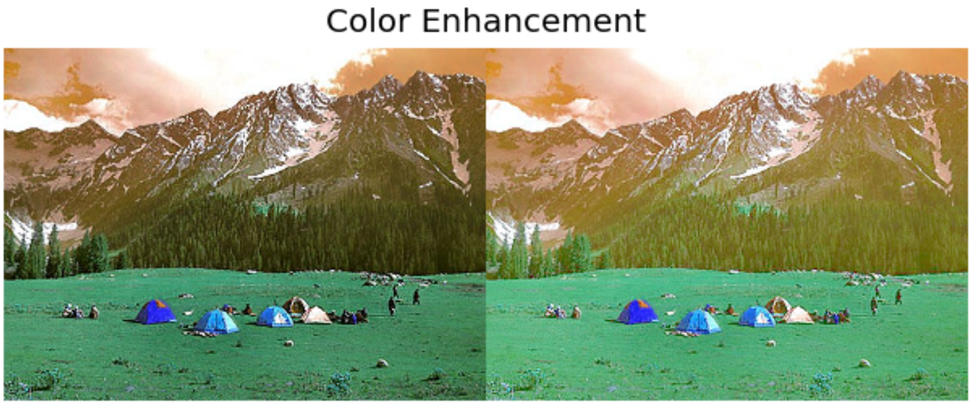


Color enhancement

In [49]:

```python
'''
Image Location:
https://pbs.twimg.com/media/Dh-MO9gWAAE-x-V.jpg
'''

enhancementFactor = 0.3

img = cv2.imread('part2_color_enhancement.jpg')
img_hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)

img_hsv[:,:,2] = img_hsv[:,:,2] + (255 - img_hsv[:,:,2]) * enhance
img_hsv = cv2.cvtColor(img_hsv, cv2.COLOR_HSV2BGR)


res = np.hstack((img,img_hsv)) #stacking images side-by-side
plt.imshow(res)
plt.title("Color Enhancement", fontsize=12)
plt.axis('off')
plt.show()
```

```
<IPython.core.display.Javascript object>
```

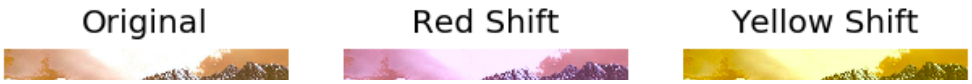


Color shift

In [51]:

```
shiftFactor = 0.2
img = cv2.imread('part2_color_enhancement.jpg')
img_lab = cv2.cvtColor(img, cv2.COLOR_BGR2Lab)
imgR = img_lab.copy()
imgY = img_lab.copy()


imgR[:,:,1] = img_lab[:,:,1] + (255 - img_lab[:,:,1]) * shiftFacto
imgY[:,:,1] = img_lab[:,:,1] - (img_lab[:,:,1]) * shiftFactor

imgR = cv2.cvtColor(imgR, cv2.COLOR_Lab2BGR)
imgY = cv2.cvtColor(imgY, cv2.COLOR_Lab2BGR)


fig = plt.figure()

plt.subplot(1, 3, 1)
plt.imshow(img)
plt.title("Original", fontsize=12)
plt.axis('off')

plt.subplot(1, 3, 2)
plt.imshow(imgR)
plt.title("Red Shift", fontsize=12)
plt.axis('off')

plt.subplot(1, 3, 3)
plt.imshow(imgY)
plt.title("Yellow Shift", fontsize=12)
plt.axis('off')


plt.show()
```

```
<IPython.core.display.Javascript object>
```



In [ ]: