soc-24-a1

June 20, 2024

1 Image Restoration

[2]:
```
import cv2 as cv
import numpy as np
import matplotlib.pyplot as plt
```

[3]:
```
import cv2 as cv
import os

# Image paths
img_path1 = r'd:\22B2492@iitb.ac.in\SOC_24\image1.jpg'
img_path2 = r'd:\22B2492@iitb.ac.in\SOC_24\image2.jpg'
img_path3 = r'd:\22B2492@iitb.ac.in\SOC_24\image3.jpg'
img_path4 = r'd:\22B2492@iitb.ac.in\SOC_24\image4.jpg'

# Load and check first image
if not os.path.isfile(img_path1):
    print(f"Error: File does not exist at {img_path1}")
else:
    img1 = cv.imread(img_path1)
    if img1 is None:
        print(f"Error: Unable to load image at {img_path1}")
    else:
        print(f"Image 1 loaded successfully. Image shape: {img1.shape}")
        # Apply Gaussian Blur
        g_blur1 = cv.GaussianBlur(img1, (7, 7), 0)
        cv.imwrite('g_blur1.jpg', g_blur1)
        print("Gaussian Blur applied and saved as 'g_blur1.jpg'")

# Load and check second image
if not os.path.isfile(img_path2):
    print(f"Error: File does not exist at {img_path2}")
else:
    img2 = cv.imread(img_path2)
    if img2 is None:
        print(f"Error: Unable to load image at {img_path2}")
    else:
        print(f"Image 2 loaded successfully. Image shape: {img2.shape}")
        # Apply Gaussian Blur
        g_blur2 = cv.GaussianBlur(img2, (7, 7), 0)
        cv.imwrite('g_blur2.jpg', g_blur2)
        print("Gaussian Blur applied and saved as 'g_blur2.jpg'")

# Load and check third image
if not os.path.isfile(img_path3):
    print(f"Error: File does not exist at {img_path3}")
else:
    img3 = cv.imread(img_path3)
    if img3 is None:
        print(f"Error: Unable to load image at {img_path3}")
    else:
        print(f"Image 3 loaded successfully. Image shape: {img3.shape}")
        # Apply Gaussian Blur
        g_blur3 = cv.GaussianBlur(img3, (7, 7), 0)
        cv.imwrite('g_blur3.jpg', g_blur3)
        print("Gaussian Blur applied and saved as 'g_blur3.jpg'")

# Load and check second image
if not os.path.isfile(img_path4):
    print(f"Error: File does not exist at {img_path4}")
else:
    img4 = cv.imread(img_path4)
    if img4 is None:
        print(f"Error: Unable to load image at {img_path4}")
    else:
        print(f"Image 4 loaded successfully. Image shape: {img4.shape}")
        # Apply Gaussian Blur
        g_blur4 = cv.GaussianBlur(img4, (7, 7), 0)
        cv.imwrite('g_blur4.jpg', g_blur4)
        print("Gaussian Blur applied and saved as 'g_blur4.jpg'")
```

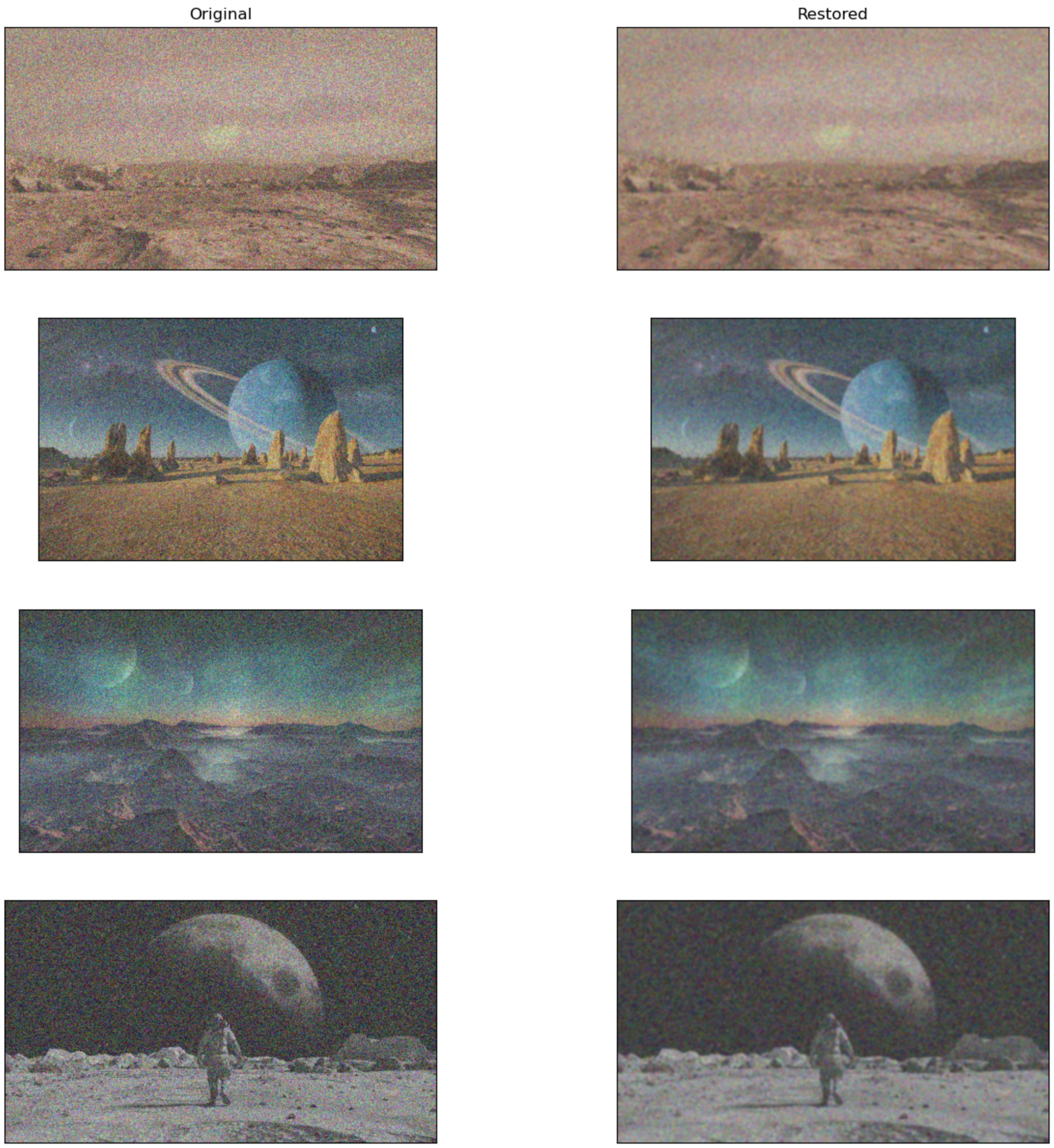
```
Image 1 loaded successfully. Image shape: (344, 612, 3)
Gaussian Blur applied and saved as 'g_blur1.jpg'
Image 2 loaded successfully. Image shape: (408, 612, 3)
Gaussian Blur applied and saved as 'g_blur2.jpg'
Image 3 loaded successfully. Image shape: (368, 612, 3)
Gaussian Blur applied and saved as 'g_blur3.jpg'
Image 4 loaded successfully. Image shape: (344, 612, 3)
Gaussian Blur applied and saved as 'g_blur4.jpg'
```

[4]:
```
#Inpainting method
mask1 = cv.imread('g_blur1.jpg',0)
imgnew1 = cv.inpaint(g_blur1,mask1,11,cv.INPAINT_NS)
```

```
mask2 = cv.imread('g_blur2.jpg',0)
imgnew2 = cv.inpaint(g_blur2,mask2,11,cv.INPAINT_NS)

mask3 = cv.imread('g_blur3.jpg',0)
imgnew3 = cv.inpaint(g_blur3,mask3,11,cv.INPAINT_NS)

mask4 = cv.imread('g_blur4.jpg',0)
imgnew4 = cv.inpaint(g_blur4,mask4,11,cv.INPAINT_NS)
```

[5]:

```
#plotting the images
plt.figure(figsize=(15,15))

plt.subplot(4,2,1)
plt.imshow(cv.cvtColor(img1,cv.COLOR_BGR2RGB)),plt.title('Original'),plt.
 ↪xticks([]),plt.yticks([])

plt.subplot(4,2,2)
plt.imshow(cv.cvtColor(imgnew1,cv.COLOR_BGR2RGB)),plt.title('Restored'),plt.
 ↪xticks([]),plt.yticks([])

plt.subplot(4,2,3)
plt.imshow(cv.cvtColor(img2,cv.COLOR_BGR2RGB)),plt.xticks([]),plt.yticks([])

plt.subplot(4,2,4)
plt.imshow(cv.cvtColor(imgnew2,cv.COLOR_BGR2RGB)),plt.xticks([]),plt.yticks([])

plt.subplot(4,2,5)
plt.imshow(cv.cvtColor(img3,cv.COLOR_BGR2RGB)),plt.xticks([]),plt.yticks([])

plt.subplot(4,2,6)
plt.imshow(cv.cvtColor(imgnew3,cv.COLOR_BGR2RGB)),plt.xticks([]),plt.yticks([])

plt.subplot(4,2,7)
plt.imshow(cv.cvtColor(img4,cv.COLOR_BGR2RGB)),plt.xticks([]),plt.yticks([])

plt.subplot(4,2,8)
plt.imshow(cv.cvtColor(imgnew4,cv.COLOR_BGR2RGB)),plt.xticks([]),plt.yticks([])

plt.show()
```

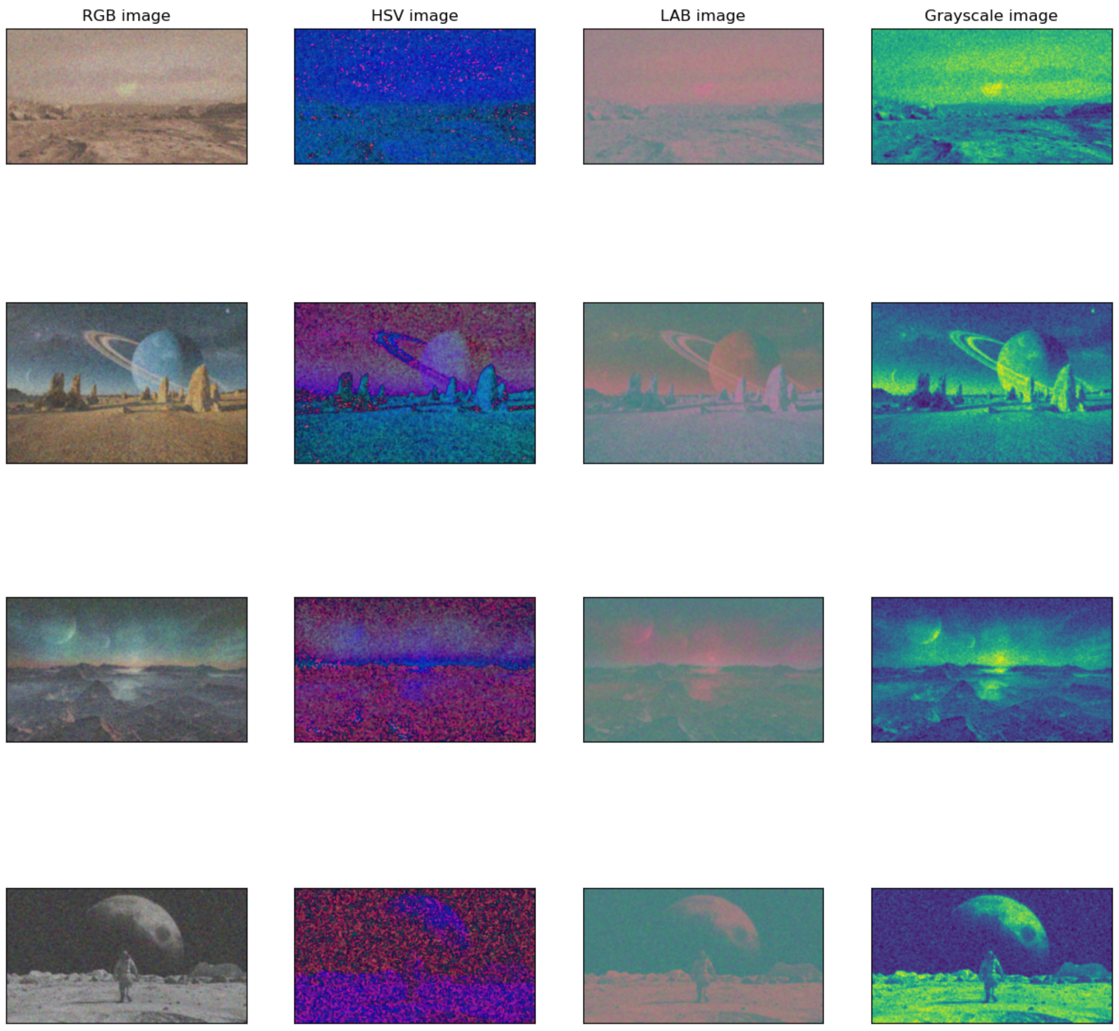


2 Color Space Exploration

[18]:
```python
#importing restored images
imagenew1 = cv.imread("D:/22B2492@iitb.ac.in/SOC_24/imgnew1.jpg",-1)
imagenew2 = cv.imread("D:/22B2492@iitb.ac.in/SOC_24/imgnew2.jpg",-1)
imagenew3 = cv.imread("D:/22B2492@iitb.ac.in/SOC_24/imgnew3.jpg",-1)
imagenew4 = cv.imread("D:/22B2492@iitb.ac.in/SOC_24/imgnew4.jpg",-1)
```

```
[22]: #plotting different images on different color modules
plt.figure(figsize=(15,15))

plt.subplot(4,4,1)
plt.imshow(cv.cvtColor(imagenew1,cv.COLOR_BGR2RGB)),plt.title('RGB image'),plt.
  ↪xticks([]),plt.yticks([])

plt.subplot(4,4,2)
plt.imshow(cv.cvtColor(imagenew1,cv.COLOR_BGR2HSV)),plt.title('HSV image'),plt.
  ↪xticks([]),plt.yticks([])

plt.subplot(4,4,3)
plt.imshow(cv.cvtColor(imagenew1,cv.COLOR_BGR2LAB)),plt.title('LAB image'),plt.
  ↪xticks([]),plt.yticks([])

plt.subplot(4,4,4)
plt.imshow(cv.cvtColor(imagenew1,cv.COLOR_BGR2GRAY)),plt.title('Grayscale␣
  ↪image'),plt.xticks([]),plt.yticks([])

plt.subplot(4,4,5)
plt.imshow(cv.cvtColor(imagenew2,cv.COLOR_BGR2RGB)),plt.xticks([]),plt.
  ↪yticks([])

plt.subplot(4,4,6)
plt.imshow(cv.cvtColor(imagenew2,cv.COLOR_BGR2HSV)),plt.xticks([]),plt.
  ↪yticks([])

plt.subplot(4,4,7)
plt.imshow(cv.cvtColor(imagenew2,cv.COLOR_BGR2LAB)),plt.xticks([]),plt.
  ↪yticks([])

plt.subplot(4,4,8)
plt.imshow(cv.cvtColor(imagenew2,cv.COLOR_BGR2GRAY)),plt.xticks([]),plt.
  ↪yticks([])

plt.subplot(4,4,9)
plt.imshow(cv.cvtColor(imagenew3,cv.COLOR_BGR2RGB)),plt.xticks([]),plt.
  ↪yticks([])

plt.subplot(4,4,10)
plt.imshow(cv.cvtColor(imagenew3,cv.COLOR_BGR2HSV)),plt.xticks([]),plt.
  ↪yticks([])

plt.subplot(4,4,11)
plt.imshow(cv.cvtColor(imagenew3,cv.COLOR_BGR2LAB)),plt.xticks([]),plt.
  ↪yticks([])
```

```
plt.subplot(4,4,12)
plt.imshow(cv.cvtColor(imagenew3,cv.COLOR_BGR2GRAY)),plt.xticks([]),plt.
 ↪yticks([])

plt.subplot(4,4,13)
plt.imshow(cv.cvtColor(imagenew4,cv.COLOR_BGR2RGB)),plt.xticks([]),plt.
 ↪yticks([])

plt.subplot(4,4,14)
plt.imshow(cv.cvtColor(imagenew4,cv.COLOR_BGR2HSV)),plt.xticks([]),plt.
 ↪yticks([])

plt.subplot(4,4,15)
plt.imshow(cv.cvtColor(imagenew4,cv.COLOR_BGR2LAB)),plt.xticks([]),plt.
 ↪yticks([])

plt.subplot(4,4,16)
plt.imshow(cv.cvtColor(imagenew4,cv.COLOR_BGR2GRAY)),plt.xticks([]),plt.
 ↪yticks([])

plt.show()
```



RGB(Red,Green, Blue) :- Most common way to represent color in digital formats. Each pixel is a combination of these three colors. Ideal for tasks involving color display and basic collor-based operations.

HSV(Hue, Saturation, Value) :- It separates color info(hue) from intensity(value) and colorfulness(saturation). Ideal for tasks that require robust color segmentation and lighting -invariant object detection.

LAB(Lightness(L*), *a*(green-red component), b*(blue-yellow component)) :- Designed for human vision, it separates lightness from color information. Useful for advanced color correction and accurate color-based segmentation.

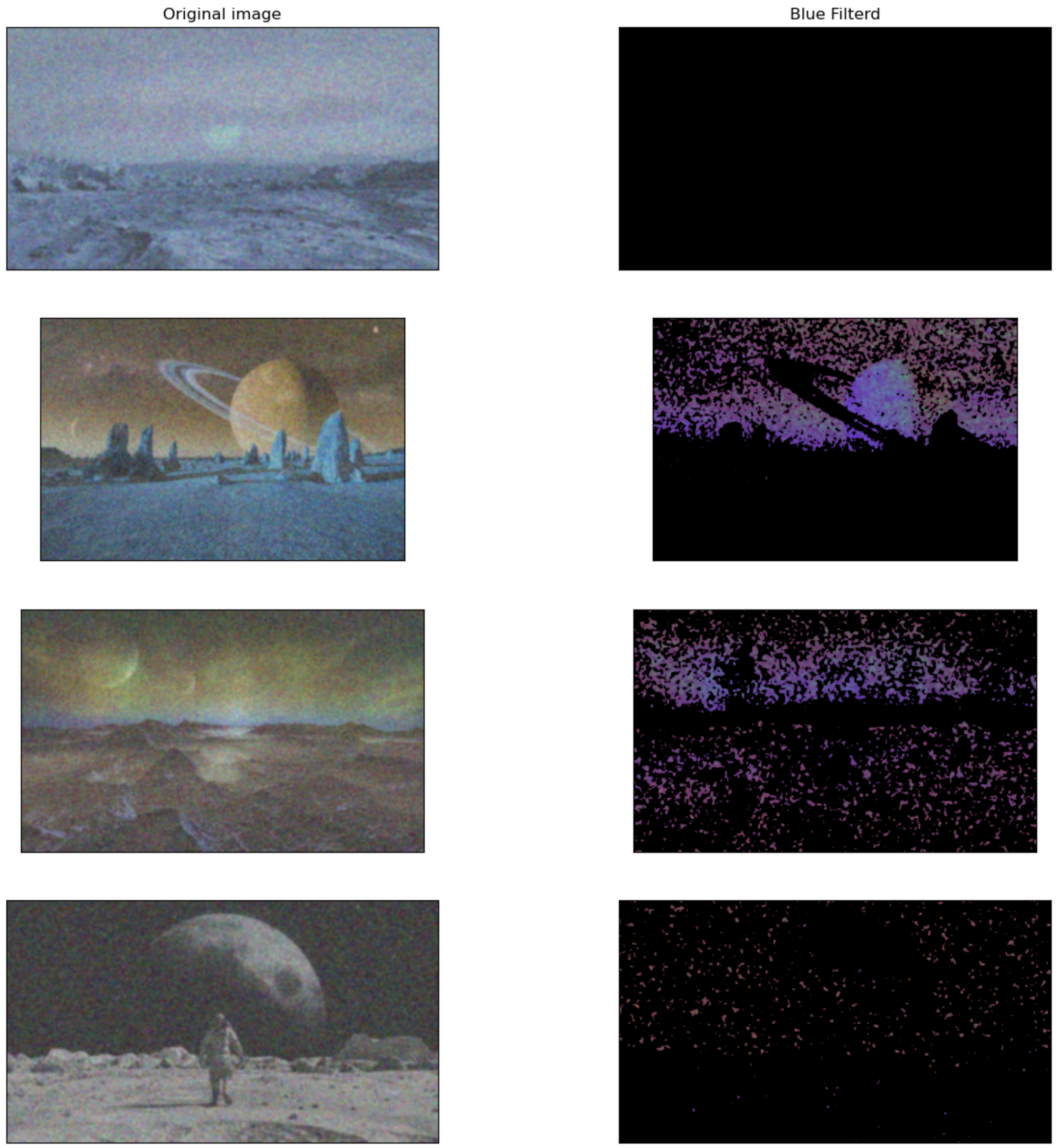
GRAY(Grayscale) :- Represent intensity variation without color. Useful for tasks like edge detection, texture analysis and simplified processing.

2.1 Highlighting Specific Color

[26]:
```
#importing all restored images in HSV format
hsv1 = cv.cvtColor(imagenew1, cv.COLOR_BGR2HSV)
hsv2 = cv.cvtColor(imagenew2, cv.COLOR_BGR2HSV)
hsv3 = cv.cvtColor(imagenew3, cv.COLOR_BGR2HSV)
hsv4 = cv.cvtColor(imagenew4, cv.COLOR_BGR2HSV)
```

[27]:
```
#using mask of blue color to highlight it on each hsv images
#for hsv1
lower_blue1 = np.array([90,50,50])
upper_blue1 = np.array([130,255,255])
mask_blue1 = cv.inRange(hsv1, lower_blue1, upper_blue1)
result_blue1 = cv.bitwise_and(hsv1,hsv1, mask = mask_blue1)

#for hsv2
lower_blue2 = np.array([90,50,50])
upper_blue2 = np.array([130,255,255])
mask_blue2 = cv.inRange(hsv2, lower_blue2, upper_blue1)
result_blue2 = cv.bitwise_and(hsv2,hsv2, mask = mask_blue2)

#for hsv3
lower_blue3 = np.array([90,50,50])
upper_blue3 = np.array([130,255,255])
mask_blue3 = cv.inRange(hsv3, lower_blue3, upper_blue3)
result_blue3 = cv.bitwise_and(hsv3,hsv3, mask = mask_blue3)

#for hsv4
lower_blue4 = np.array([90,50,50])
upper_blue4 = np.array([130,255,255])
mask_blue4 = cv.inRange(hsv4, lower_blue4, upper_blue4)
result_blue4 = cv.bitwise_and(hsv4,hsv4, mask = mask_blue4)
```

[28]:
```
#plotting all images before and after blue filter
plt.figure(figsize=(15,15))

plt.subplot(4,2,1)
plt.imshow(imagenew1),plt.title('Original image'),plt.xticks([]),plt.yticks([])

plt.subplot(4,2,2)
plt.imshow(result_blue1),plt.title('Blue Filterd'),plt.xticks([]),plt.yticks([])

plt.subplot(4,2,3)
plt.imshow(imagenew2),plt.xticks([]),plt.yticks([])

plt.subplot(4,2,4)
plt.imshow(result_blue2),plt.xticks([]),plt.yticks([])
plt.subplot(4,2,5)
plt.imshow(imagenew3),plt.xticks([]),plt.yticks([])

plt.subplot(4,2,6)
plt.imshow(result_blue3),plt.xticks([]),plt.yticks([])

plt.subplot(4,2,7)
plt.imshow(imagenew4),plt.xticks([]),plt.yticks([])

plt.subplot(4,2,8)
plt.imshow(result_blue4),plt.xticks([]),plt.yticks([])

plt.show()
```


Original image
Blue Filterd