Name :

Awais Ahmad

Roll-No:

SU92-BSAIM-F23-132

Section:

BSAI-4C

Task:

(5)

Open-CV

1) Importing libraries and installing Open CV:

```
!pip install opencv-python
!pip install opencv-python-headless
!pip install opencv-python

import cv2
import matplotlib.pyplot as plt
import numpy as np
```

2) Loading Pictures:

```
import cv2
import matplotlib.pyplot as plt

# Correct the file path
img = cv2.imread(r"C:\Users\digit\Downloads\cat.jpg")
if img is None:
    print("Image not loaded. Check the file path.")
else:
    # Convert BGR to RGB for correct color display
    img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    plt.imshow(img_rgb)
```

```
 plt.axis('off')
plt.show()
```

Output:

3) Image Details:

```
img.shape
```

Output:

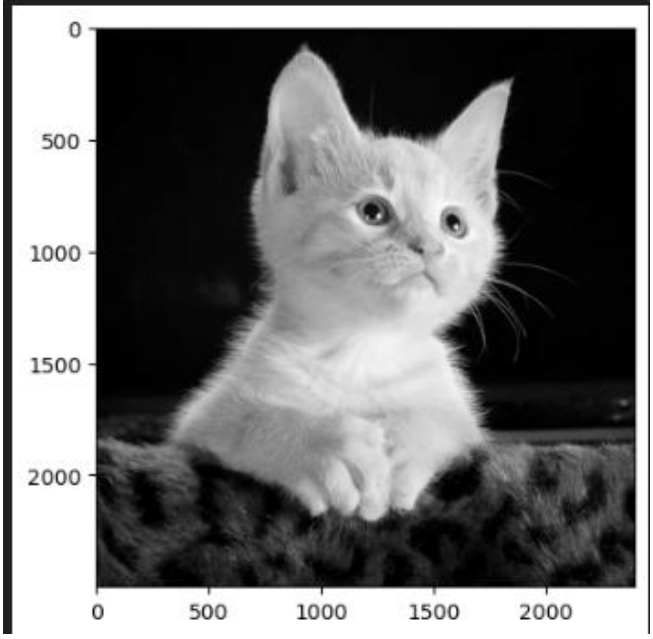
(2500, 2392, 3)

4) Read image in greystyle:

```
img_gray = cv2.imread(r'C:\Users\digit\Downloads\cat.jpg')
img_gray = cv2.cvtColor(img_gray, cv2.COLOR_BGR2GRAY)
plt.imshow(img_gray, cmap='gray')
```

Output:

```
<matplotlib.image.AxesImage at 0x16809a72210>
```



5)

```
# Shape of Grayscale image
img_gray.shape
```

```
(2500, 2392)
```

Write/Save Image

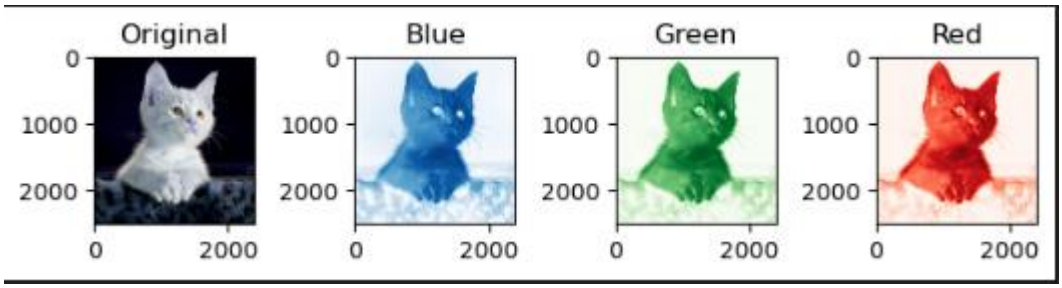
```
cv2.imwrite("owl_gray.jpg", img_gray)
```

```
True
```

6) Color spaces:

```
img = cv2.imread(r'C:\Users\digit\Downloads\cat.jpg')
red, green, blue = cv2.split(img)
fig, axes = plt.subplots(nrows=1, ncols=4, facecolor = 'white')
axes[0].imshow(img, )
axes[1].imshow(blue, cmap='Blues')
axes[2].imshow(green, cmap='Greens')
axes[3].imshow(red, cmap='Reds')
axes[0].set_title('Original')
axes[1].set_title('Blue')
axes[2].set_title('Green')
axes[3].set_title('Red')
fig.tight_layout()
plt.show()
 Matplotlib color Scheme = Red, Green, Blue.
 Opencv color scheme = Blue, Green, Red.
```

Output:



7) Arithmetic Operations on Images Addition of Images:

```
import cv2
import matplotlib.pyplot as plt
img = cv2.imread(r'C:\Users\digit\Downloads\cat.jpg')
img2 = cv2.imread(r'C:\Users\digit\Downloads\forest.jpeg')
# Resize both images to the same size
target_size = (436, 612)
img = cv2.resize(img, target_size)
img2 = cv2.resize(img2, target_size)
weighted_sum = cv2.addWeighted(img, 0.5, img2, 0.5, 0)
img_rgb = cv2.cvtColor(weighted_sum, cv2.COLOR_BGR2RGB)
plt.imshow(img_rgb)
plt.axis('off')
```

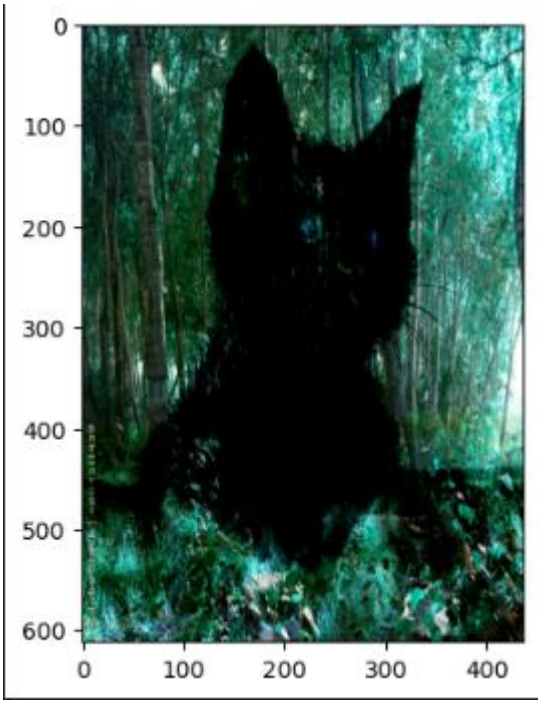
```
plt.show()
```

Output:

8) Subtraction of image:

```
sub = cv2.subtract(img2, img)
plt.imshow(sub)
```

Output:



9) Operations:

```
import cv2
import matplotlib.pyplot as plt
img = cv2.imread(r'C:\Users\digit\Downloads\sander-traa-5ldh94XzU4I-
unsplash.jpg')
img2 = cv2.imread(r'C:\Users\digit\Downloads\raymond-petrik-GEjnfY995yc-
unsplash.jpg')

if img is None:
 raise ValueError("First image not loaded. Check the file path.")
if img2 is None:
 raise ValueError("Second image not loaded. Check the file path.")
img2 = cv2.resize(img2, (img.shape[1], img.shape[0]))
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
img2 = cv2.cvtColor(img2, cv2.COLOR_BGR2RGB)
dest_AND = cv2.bitwise_and(img2, img)
dest_OR = cv2.bitwise_or(img2, img)
dest_XOR = cv2.bitwise_xor(img2, img)
dest_NOT = cv2.bitwise_not(img2)
fig, axes = plt.subplots(nrows=2, ncols=3, facecolor='white', figsize=(12,
8))
  axes = axes.flatten()
fig.tight_layout()
axes[0].imshow(img)
```

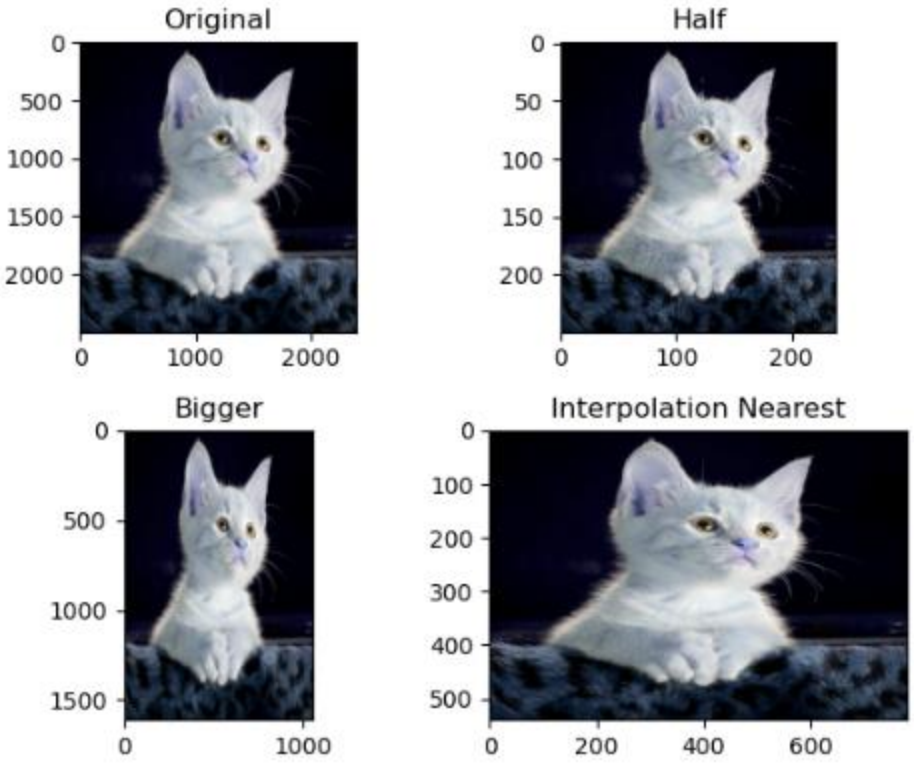
```
axes[1].imshow(img2)
axes[2].imshow(dest_AND)
axes[3].imshow(dest_OR)
axes[4].imshow(dest_XOR)
axes[5].imshow(dest_NOT)
axes[0].set_title('Image 1')
axes[1].set_title('Image 2')
axes[2].set_title('AND Operation')
axes[3].set_title('OR Operation')
axes[4].set_title('XOR Operation')
axes[5].set_title('NOT Operation')
for ax in axes:
    ax.axis('off')
plt.show()
```

OUTPUT:

10) Image resizing:

```
image = cv2.imread(r'C:\Users\digit\Downloads\cat.jpg')
half = cv2.resize(image, (0, 0), fx = 0.1, fy = 0.1)
bigger = cv2.resize(image, (1050, 1610))
stretch_near = cv2.resize(image, (780, 540),
        interpolation = cv2.INTER_LINEAR)
Titles =["Original", "Half", "Bigger", "Interpolation Nearest"]
images =[image, half, bigger, stretch_near]
count = 4
for i in range(count):
   plt.subplot(2, 2, i + 1)
   plt.title(Titles[i])
   plt.tight_layout()
  plt.imshow(images[i])
```

Output:



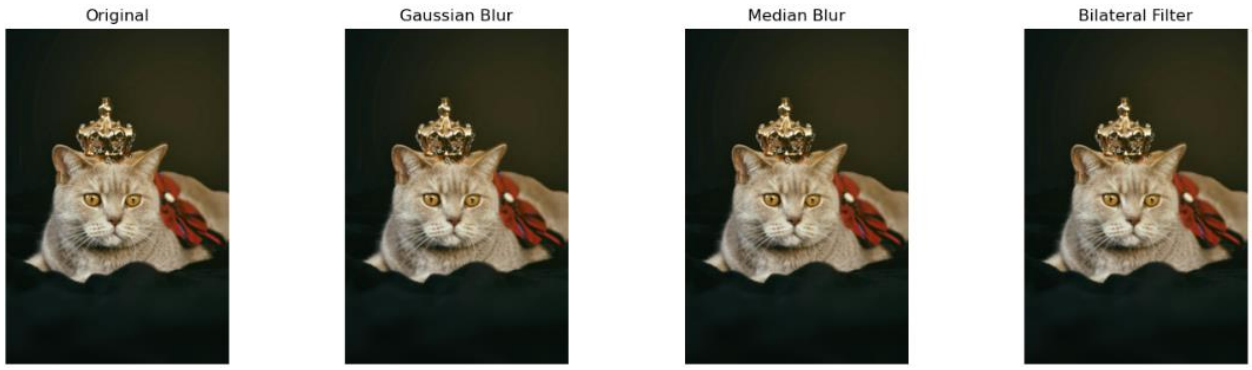
11) Image Erosion:

```
import cv2
import numpy as np
import matplotlib.pyplot as pl
img = cv2.imread(r"C:\Users\digit\Downloads\pexels-katarzyna-modrzejewska-
495044-1314550.jpg")
if img is None:
   raise ValueError("Image not loaded. Check the file path.")
img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
kernel = np.ones((5, 5), np.uint8)
eroded_img = cv2.erode(img_rgb, kernel, iterations=2)
fig, axes = plt.subplots(nrows=1, ncols=2, figsize=(14, 7))
fig.tight_layout()1
axes[0].imshow(img_rgb)
axes[0].set_title('Original')
axes[0].axis('off')
axes[1].imshow(eroded_img)
axes[1].set_title('Eroded')
axes[1].axis('off')
plt.show()
```

Output:

12) Blurring an image:

```
import cv2
import matplotlib.pyplot as plt
img = cv2.imread(r"C:\Users\digit\Downloads\pexels-katarzyna-modrzejewska-
495044-1314550.jpg")
if img is None:
   raise ValueError("Image not loaded. Check the file path.")
img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
gaussian = cv2.GaussianBlur(img_rgb, (15, 15), 0)
median = cv2.medianBlur(img_rgb, 11)
bilateral = cv2.bilateralFilter(img_rgb, 15, 150, 150)
fig, axes = plt.subplots(nrows=1, ncols=4, figsize=(14, 4))
fig.tight_layout()
axes[0].imshow(img_rgb)
axes[0].set_title('Original')
axes[0].axis('off')
axes[1].imshow(gaussian)
axes[1].set_title('Gaussian Blur')
axes[1].axis('off')
axes[2].imshow(median)
axes[2].set_title('Median Blur')
axes[2].axis('off')
axes[3].imshow(bilateral)
axes[3].set_title('Bilateral Filter')
axes[3].axis('off')
plt.show()
```



13) Edge Detection:

```
img = cv2.imread(r'C:\Users\digit\Downloads\cat.jpg', cv2.IMREAD_GRAYSCALE)
edges = cv2.Canny(img, 100, 200)
fig, axes = plt.subplots(nrows=1, ncols=2, figsize=(7, 6))
fig.tight_layout()
axes[0].imshow(img, cmap='gray')
axes[1].imshow(edges, cmap='gray')
axes[0].set_title('Orignal')
axes[1].set_title('Canny Edge Detection')
titles = ['Orignal', 'Canny Edge Detection']
for ax, title in zip(axes, titles):
    ax.set_title(title)
   ax.axis('Off')
plt.show()
```

output: