Lab5

April 24, 2022

0.1 Lab5

0.1.1 Submitted By: Manav Doda

0.1.2 Roll No.: 195057

0.2 Importing Necessary modules

[1]:
```
from PIL import Image
import matplotlib.pyplot as plt
import numpy as np
import cv2
```

0.3 Objective 1

0.3.1 To understand and implement the geometrical operation on an image:

1. Translation

[20]:
```
img = cv2.imread("testImage.jpeg")
plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
```

[20]:
```
<matplotlib.image.AxesImage at 0x161385310>
```

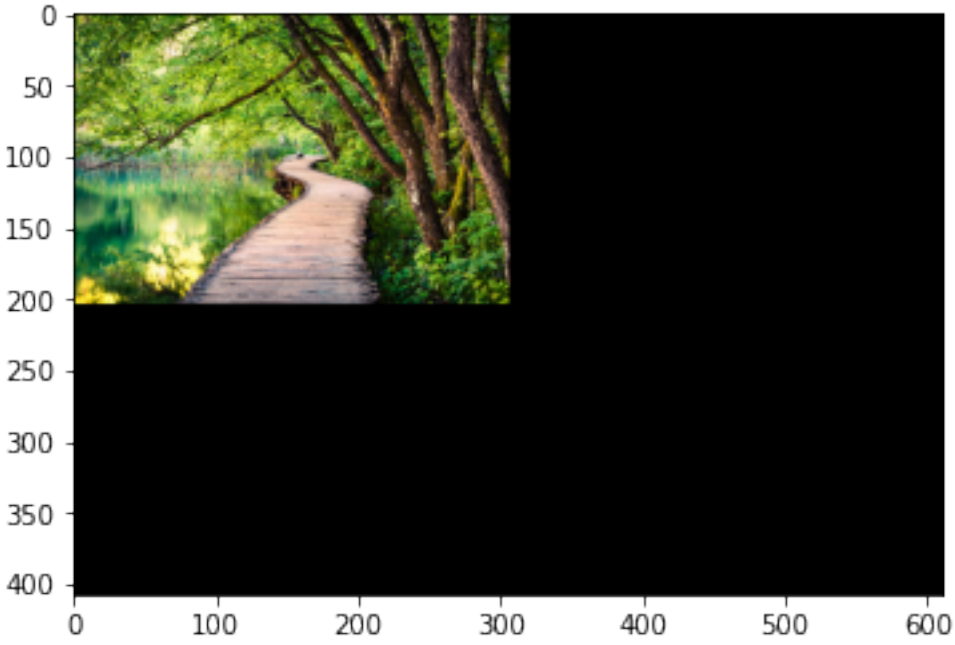
[21]:
```
shape = img.shape
rows = shape[0]
cols = shape[1]
translateX = 30
translateY = 40
```

[22]:
```
# For translating the image
for i in range(rows-translateY-1, -1, -1):
    for j in range(cols-translateX-1, -1, -1):
        img[i+translateY][j+translateX] = img[i][j]
        img[i][j] = [0,0,0]
# For removing extra pixels on bottom left
for i in range(rows-translateY, rows):
    for j in range(translateX):
        img[i][j]=[0,0,0]
# For removing extra pixels on top right
for i in range(translateY):
    for j in range(cols-translateX, cols):
        img[i][j] = [0,0,0]
plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
```

[22]: <matplotlib.image.AxesImage at 0x161071e80>

2. Scaling

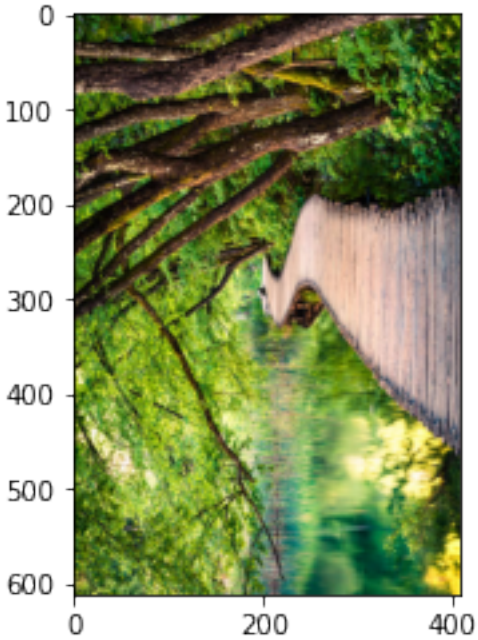
[27]:
```
img = cv2.imread("testImage.jpeg")
plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
plt.show()
for i in range(rows):
    for j in range(cols):
        if i<rows/2 and j<cols/2:
            img[i][j] = img[i*2][j*2]
        else:
            img[i][j] = [0,0,0]
plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
plt.show()
```



3. Rotation

[31]:
```
img = cv2.imread("testImage.jpeg")
plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
```

```
plt.show()
img = np.rot90(img)
plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
plt.show()
```



0.4 Objective 2

0.4.1 To understand and implement the following filter on the given image

1. Mean Filter

[55]:
```
img = cv2.imread('testImage.jpeg')
plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
plt.show()
for i in range(rows):
    for j in range(cols):
        sumA = int(img[i][j][0])
        sumB = int(img[i][j][1])
        sumC = int(img[i][j][2])
        if i>0 and j>0:
            sumA+=img[i-1][j-1][0]
            sumB+=img[i-1][j-1][1]
            sumC+=img[i-1][j-1][2]
        if i>0:
            sumA+=img[i-1][j][0]
            sumB+=img[i-1][j][1]
            sumC+=img[i-1][j][2]
        if i>0 and j<cols-1:
            sumA+=img[i-1][1][0]
            sumB+=img[i-1][1][1]
            sumC+=img[i-1][1][2]
        if j>0:
            sumA+=img[i][j-1][0]
            sumB+=img[i][j-1][1]
            sumC+=img[i][j-1][2]
        if j<cols-1:
            sumA+=img[i][j+1][0]
            sumB+=img[i][j+1][1]
            sumC+=img[i][j+1][2]
        if i<rows-1 and j>0:
            sumA+=img[i+1][j-1][0]
            sumB+=img[i+1][j-1][1]
            sumC+=img[i+1][j-1][2]
        if i<rows-1:
            sumA+=img[i+1][j][0]
            sumB+=img[i+1][j][1]
            sumC+=img[i+1][j][2]
        if i<rows-1 and j<cols-1:
            sumA+=img[i+1][j+1][0]
            sumB+=img[i+1][j+1][1]
            sumC+=img[i+1][j+1][2]
        img[i][j] = [int(sumA/9), int(sumB/9), int(sumC/9)]
```

```
plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
plt.show()
```

2. Median Filter

[60]:
```
img = cv2.imread('testImage.jpeg')
plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
plt.show()
imgNew = img.copy()
for i in range(rows):
    for j in range(cols):
        midA = [img[i][j][0]]
        midB = [img[i][j][1]]
        midC = [img[i][j][2]]
        if i>0 and j>0:
            midA.append(img[i-1][j-1][0])
            midB.append(img[i-1][j-1][1])
            midC.append(img[i-1][j-1][2])
        if i>0:
            midA.append(img[i-1][j][0])
            midB.append(img[i-1][j][1])
            midC.append(img[i-1][j][2])
        if i>0 and j<cols-1:
            midA.append(img[i-1][1][0])
            midB.append(img[i-1][1][1])
            midC.append(img[i-1][1][2])
        if j>0:
            midA.append(img[i][j-1][0])
            midB.append(img[i][j-1][1])
            midC.append(img[i][j-1][2])
        if j<cols-1:
            midA.append(img[i][j+1][0])
            midB.append(img[i][j+1][1])
            midC.append(img[i][j+1][2])
        if i<rows-1 and j>0:
            midA.append(img[i+1][j-1][0])
            midB.append(img[i+1][j-1][1])
            midC.append(img[i+1][j-1][2])
        if i<rows-1:
            midA.append(img[i+1][j][0])
            midB.append(img[i+1][j][1])
            midC.append(img[i+1][j][2])
        if i<rows-1 and j<cols-1:
            midA.append(img[i+1][j+1][0])
            midB.append(img[i+1][j+1][1])
            midC.append(img[i+1][j+1][2])
        if len(midA)%2:
            imgNew[i][j] = [midA[int(len(midA)/2)], midB[int(len(midA)/2)],␣
↪midC[int(len(midA)/2)]]
        else:
            imgNew[i][j] = [(midA[int(len(midA)/2)] + midA[int(len(midA)/2-1)])/
↪2,
```

```
                                (midB[int(len(midA)/2)]+midB[int(len(midA)/2-1)])/2,
                                (midC[int(len(midA)/2)]+midC[int(len(midA)/2-1)])]
plt.imshow(cv2.cvtColor(imgNew, cv2.COLOR_BGR2RGB))
plt.show()
```


```
/var/folders/6r/c_0pyh_s5sl09dgwcds_qyk40000gn/T/ipykernel_89788/3272025033.py:4
6: RuntimeWarning: overflow encountered in ubyte_scalars
  (midB[int(len(midA)/2)]+midB[int(len(midA)/2-1)])/2,
/var/folders/6r/c_0pyh_s5sl09dgwcds_qyk40000gn/T/ipykernel_89788/3272025033.py:4
7: RuntimeWarning: overflow encountered in ubyte_scalars
  (midC[int(len(midA)/2)]+midC[int(len(midA)/2-1)])]
/var/folders/6r/c_0pyh_s5sl09dgwcds_qyk40000gn/T/ipykernel_89788/3272025033.py:4
5: RuntimeWarning: overflow encountered in ubyte_scalars
  imgNew[i][j] = [(midA[int(len(midA)/2)] + midA[int(len(midA)/2-1)])/2,
```


0
50
100
150
200
250
300
350
400
0
100
200
300
400
500
600

[ ]: