Lab Exercise 5: Geometric Transformations and Affine Transformations

• **Objective:** Apply geometric transformations and affine transformations to images.

• **Task:** Perform image scaling, rotation, translation, and affine transformations. Visualize the effect of each transformation on an image.

Image Scaling

```
import numpy as np
import cv2
from google.colab.patches import cv2_imshow

# Reading the original image before resizing
img = cv2.imread('Effiel Tower.jpg')
resized_img = cv2.resize(img, (300, 300))
# original dimensions of the image
height, width = img.shape[:2]
height,width
```

```
(4032, 3024)
```

```
# Displaying the orginial Image
cv2_imshow(img)
```

```
# Resizing the image
resized_img = cv2.resize(img, (300, 300))
# Dimensions after resizing
# original dimensions of the image
height, width = resized_img.shape[:2]
height,width
```

```
(300, 300)
```

```
# Save the resized image
cv2.imwrite("Effiel Tower Resized.jpg", resized_img)
img_resized = cv2.imread('Effiel Tower Resized.jpg',0)

# displaying the image
cv2_imshow(img_resized)
```

Image Translation

```
# get tx and ty values for translation
# you can specify any value of your choice
tx, ty = width / 4, height / 4

# create the translation matrix using tx and ty, it is a NumPy array
translation_matrix = np.array([
    [1, 0, tx],
```

```
    [0, 1, ty]
], dtype=np.float32)

# apply the translation to the image
translated_image = cv2.warpAffine(src=resized_img,
M=translation_matrix, dsize=(width, height))

# display the original and the Translated images
cv2_imshow( translated_image)
# save the translated image to disk
cv2.imwrite('translated_image.jpg', translated_image)
```

```
True
```

Image Rotation

```
# Reading the image
resized_img = cv2.imread('Effiel Tower Resized.jpg')

# Displaying the image
cv2_imshow(resized_img)
```

```
# Dividing height and width by 2 to get the center of the image
height, width = resized_img.shape[:2]
center = (width/2, height/2)

# the above center is the center of rotation axis
# use cv2.getRotationMatrix2D() to get the rotation matrix
rotate_matrix = cv2.getRotationMatrix2D(center=center, angle=30,
scale=1)

# Rotate the image using cv2.warpAffine
rotated_image = cv2.warpAffine(src=resized_img, M=rotate_matrix,
dsize=(width, height))

# visualize the original and the rotated image
cv2_imshow(rotated_image)
```

```
# write the output, the rotated image to disk
cv2.imwrite('rotated_image.jpg', rotated_image)
```

```
True
```