INT3404E 20 - Image Processing: Homeworks 1

Nguyen Tien Dat 20021327

1 Basic Python (done)

2 Google Colaboratory (done)

3 OpenCV, Numpy, Matplotlib, and Latex Report

3.1 grayscale_image

Since OpenCV uses BGR as its default colour, matplotlib uses RGB. When displaying, we will need to convertfrom BGR to RGB

```python
def display_image(image, title="Image"):
    """
    Display an image using matplotlib. Remember to use plt.show() to display the image
    """
    image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
    plt.imshow(image)
    plt.title(title)
    plt.show()
```

Then we can convert to grey scale by running 2 loops.

```python
def grayscale_image(image):
    """
    Convert an image to grayscale. Convert the original image to a grayscale image. In a grayscale image, the pix
    3 channels will be the same for a particular X, Y coordinate. The equation for the pixel value
    [1] is given by:
        p = 0.299R + 0.587G + 0.114B
    Where the R, G, B are the values for each of the corresponding channels. We will do this by
    creating an array called img_gray with the same shape as img
    """
    # Get the shape of the image
    height, width = image.shape[:2]

    # Create an array to store the grayscale image
    img_gray = np.zeros((height, width), dtype=np.uint8)

    # Loop over the image and calculate the grayscale value for each pixel
    for i in range(height):
        for j in range(width):
            # Get the pixel value
            pixel = image[i, j]

            # Calculate the grayscale value
            gray = 0.299 * pixel[0] + 0.587 * pixel[1] + 0.114 * pixel[2]

            # Store the grayscale value in the img_gray array
            img_gray[i, j] = gray

    return img_gray
```

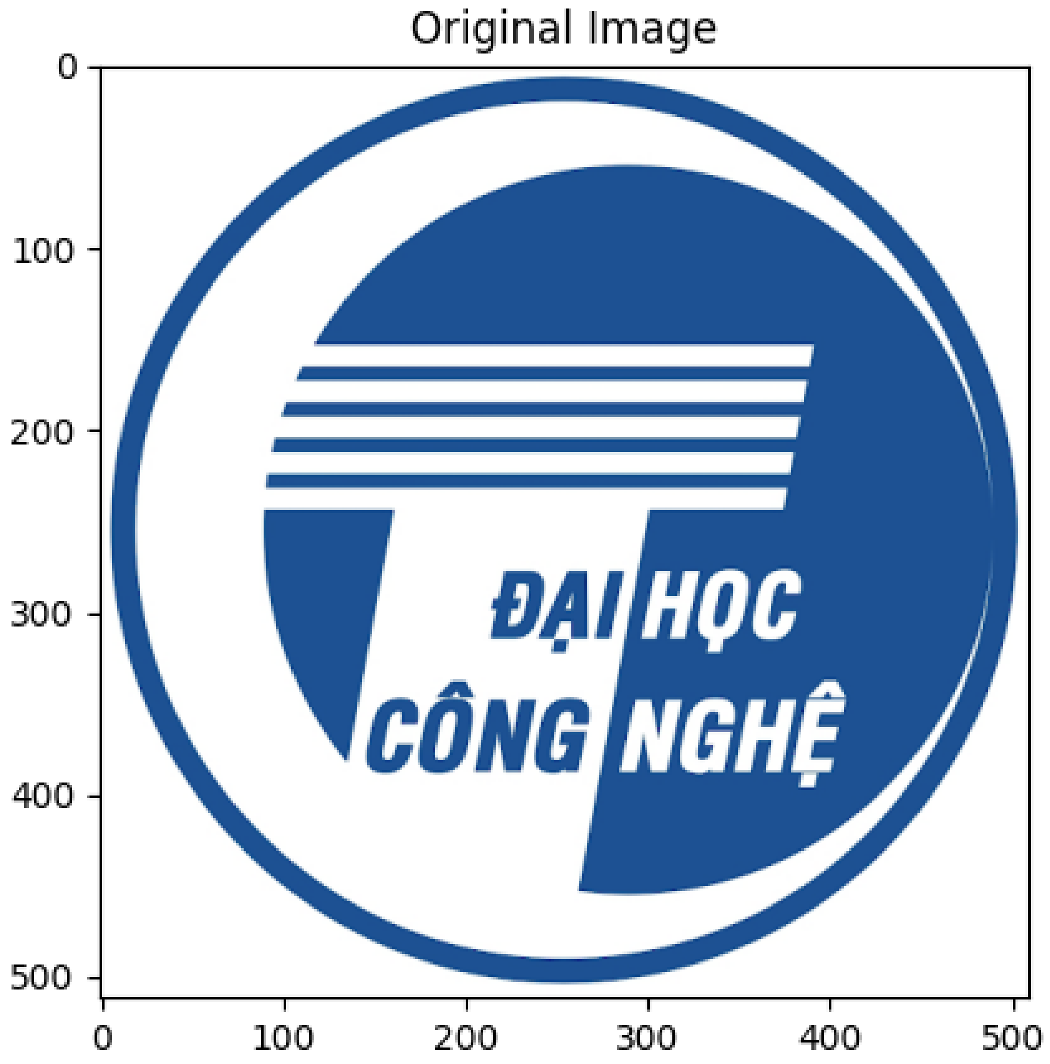

Figure 1: Original Image

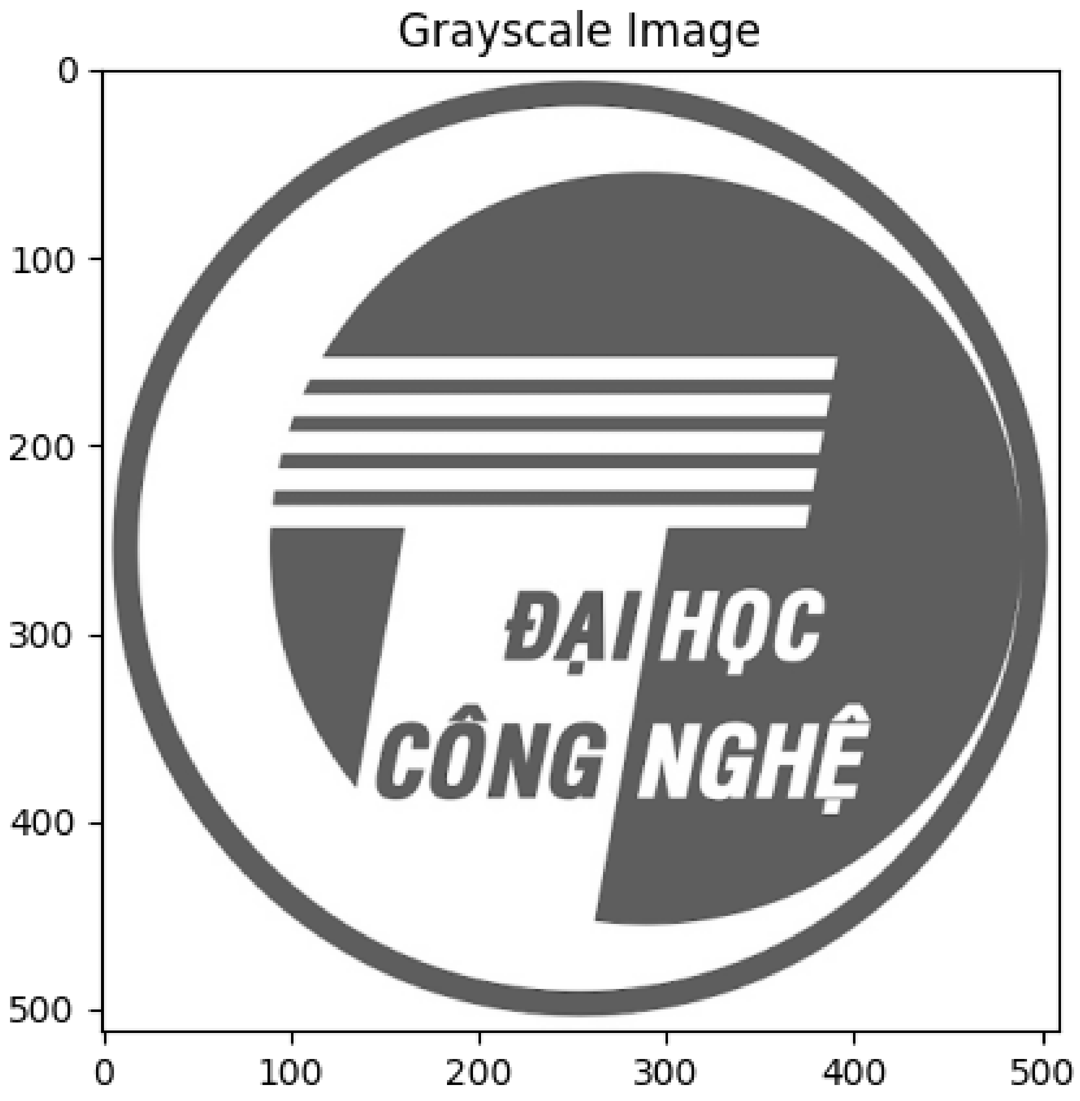

Figure 2: Grey Scale Image

3.2 flip_image

Image flipping using OpenCV

```python
def flip_image(image):
    """
    Flip an image horizontally using OpenCV
    """
    # Use the flip function from OpenCV to flip the image horizontally
    flipped_image = cv2.flip(image, 1)

    return flipped_image
```

OpenCV flip function take flipCode: This parameter determines the direction of the flip. It should be an integer value that specifies the flip code. A value of 1 indicates a horizontal flip, 0 indicates a vertical flip, and a negative value indicates both horizontal and vertical flip.

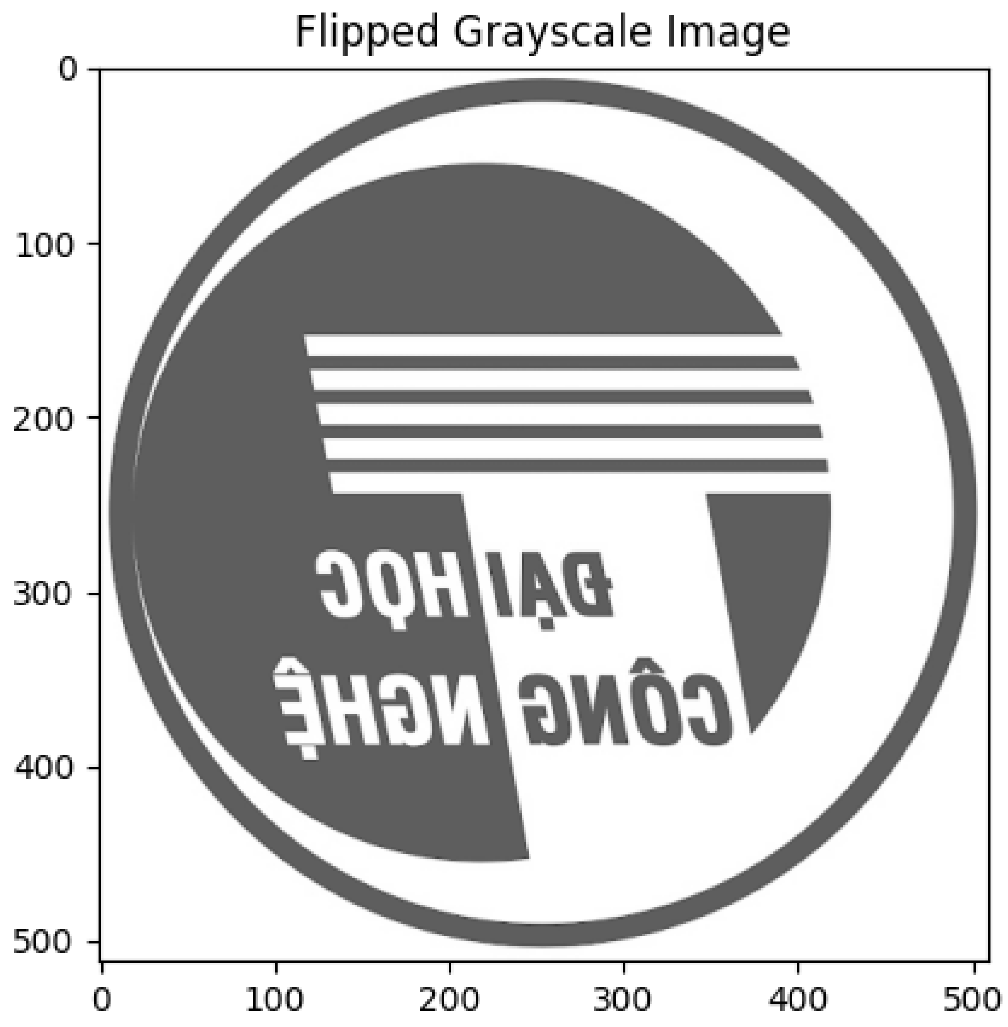

Figure 3: Enter Caption

3.3 rotate_image

Image Rotation using OpenCV

```python
def rotate_image(image, angle):
    """
    Rotate an image using OpenCV. The angle is in degrees
    """
    # Get the shape of the image
    height, width = image.shape[:2]

    # Calculate the center of the image
    center = (width // 2, height // 2)

    # Define the rotation matrix
    rotation_matrix = cv2.getRotationMatrix2D(center, angle, 1.0)

    # Apply the rotation to the image
    rotated_image = cv2.warpAffine(image, rotation_matrix, (width, height))

    return rotated_image
```

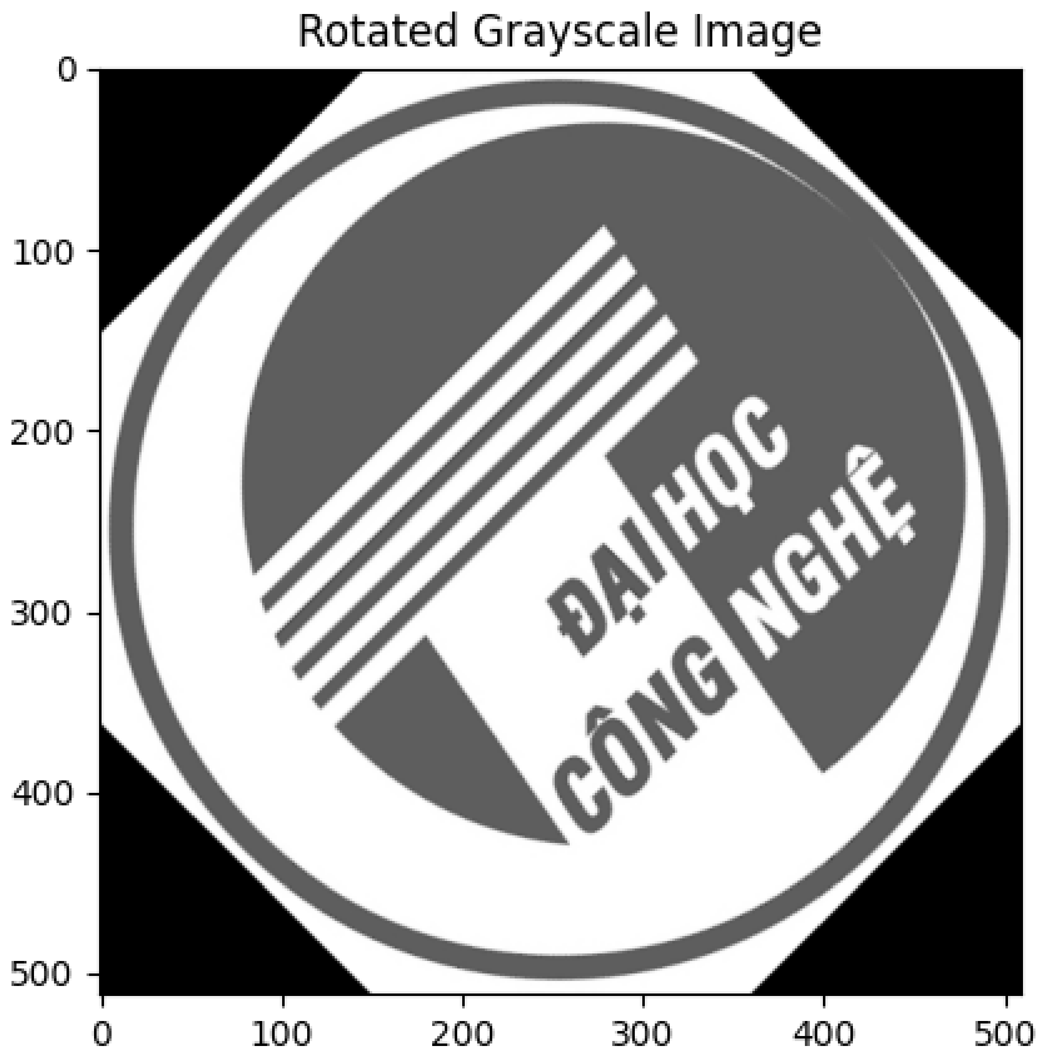

Figure 4: Rotated Image

The getRotationMatrix2D function takes three parameters:

1. center: This parameter specifies the center point around which the rotation will be performed. It is of type cv2.typing.Point2f, which represents a 2D point with floating-point coordinates.
2. angle: This parameter specifies the angle of rotation in degrees. It is of type float.
3. scale: This parameter specifies the scaling factor applied to the rotated image or points. It is of type float and is optional. By default, it is set to 1.0, which means no scaling is applied.

The result is an image rotated 45 degree counter-clockwise when called
img_gray_rotated = rotate_image(img_gray, 45)