INT3404E 20 - Image Processing: Homeworks 1

Ngô Tùng Lâm - 22028092

1 Task

A picture or image can be represented as a NumPy array of "pixels", with dimensions H × W × C, where H is the height, W is the width, and C is the number of color channels. Figure 1 illustrates the coordinate system. The origin is at the top left corner and the first dimension indicates the Y (row) direction, while the second dimension indicates the X (column) dimension. Typically we will use an image with channels that give the Red, Green, and Blue "level" of each pixel, which is referred to in the short form RGB. The value for each channel ranges from 0 (darkest) to 255 (lightest). However, when loading an image through Matplotlib, this range will be scaled from 0 (darkest) to 1 (brightest) instead, and will be a real number, rather than an integer.

You will write Python code to load an image perform several manipulations to the image and visualize their effects. You'll need to get the file uet.png from the same place you downloaded this assignment

Figure 1: Original image

2 Image processing functions

2.1 `grayscale_image`

- **Purpose:** Turn an image into a grayscale image
- **Implementation:**

```
def grayscale_image(image):
    """
    Convert an image to grayscale. Convert the original image to a grayscale image. In a
        grayscale image, the pixel value of the
    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
    """
    img_gray = image.copy()
    for i in range(img_gray.shape[0]):
        for j in range(img_gray.shape[1]):
            img_gray[i, j, :] = [0.299*img_gray[i, j, 2] + 0.587*img_gray[i, j, 1] + 0.114*
                img_gray[i, j, 0]]
    return img_gray
```

- **Result:**

Figure 2: Grayscale image

2.2 `flip_image`

- **Purpose:** Flip the image horizontally
- **Implement:**

```
def flip_image(image):
    """
    Flip an image horizontally using OpenCV
    """
    return cv2.flip(image, 1)
```

- **Result:**

Figure 3: Grayscale image after flipping on horizontal

2.3 `rotated_image`

- **Purpose:** Rotate the image by an angle (*The angle is in degrees*).
- **Implentation:**

```python
def rotate_image(image, angle):
    """
    Rotate an image using OpenCV. The angle is in degrees
    """
    rows, cols = image.shape[:2]
    M = cv2.getRotationMatrix2D((cols / 2, rows / 2), angle, 1)
    return cv2.warpAffine(image, M, (cols, rows))
```

- **Result:**

Figure 4: Grayscale image after rotated 45 degrees