INT3404E 20 - Image Processing: Homeworks 1

21021491 - Ngo Thuong Hieu

1 Original Image

Figure 1: Original image

This is the image which I have to process.

2 Flipping Image

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

$cv2.flip()$ is a function provided by the OpenCV library in Python. This function is used to flip images either horizontally, vertically, or both horizontally and vertically. It takes three parameters:

- *src*: The input image to be flipped.
- $flipCode$: An integer value that specifies how the image should be flipped. It can be one of the following:
  - 0: Flips the image horizontally around the y-axis.
  - 1: Flips the image vertically around the x-axis.
  - -1: Flips the image both horizontally and vertically.
- *dst*: An optional parameter that specifies the output array where the flipped image will be stored. If not provided, the function creates a new array to store the flipped image.

The result flipped image is Figure 2.

3 Rotating Image

Figure 2: Flipped image

```python
def rotate_image(image, angle):
    """
    Rotate an image using OpenCV. The angle is in degrees
    """
    (h, w) = image.shape[:2]
    center = (w // 2, h // 2)
    # perform the rotation
    M = cv2.getRotationMatrix2D(center, angle, scale=1.0)
    img = cv2.warpAffine(image, M, (w, h))

    return img
```

$cv2.getRotationMatrix2D$ is a function used for generating an affine transformation matrix for rotating an image around a specified center.

- *center*: A tuple (x, y) representing the center of rotation.
- *angle*: The rotation angle in degrees, measured clockwise.
- *scale*: An optional parameter representing the scale factor. By default, it's set to 1.0, meaning no scaling is applied.

$cv2.warpAffine$ is a function used for applying affine transformations to images. Affine transformations include operations such as rotation, translation, scaling, and shearing.

- *src*: The input image.
- M: The 2x3 transformation matrix representing the affine transformation to be applied. This matrix can be obtained
- *dsize*: The size of the output image, specified as a tuple $(width, height)$.

The result is Figure 3.

Figure 3: Rotated image

4 Grayscaling Image

```python
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
    """
    grayscale = image[:, :, 0] * 0.299 + image[:, :, 1] * 0.587 + image[:, :, 2] * 0.114

    return grayscale
```

Figure 4: Grayscale image