IMAGE AND VIDEO PROCESSING LABORATORY (EC69211) – AUTUMN 2024

Experiment 1: Image Scaling and Rotation Using Nearest Neighbour and Bilinear Interpolation

Submitted by: Unnati Singh (21EC39027)

Objective

To develop a Python program that reads an image and performs image scaling and rotation using custom-implemented functions.

Methodology

1. Image Acquisition: Utilize OpenCV to read the input image file into a suitable format.

2. Scaling: Implement custom functions for image scaling using both nearest neighbour and bilinear interpolation methods based on a given scaling factor.

3. Rotation: Develop custom functions for image rotation using both nearest neighbour and bilinear interpolation methods based on a given rotation angle (theta).

4. Output: Save the transformed images using OpenCV or other suitable libraries for both interpolation methods.

Files

1. `experiment1_iplab.py`: Contains the main code for scaling and rotating images.
2. `Input Image/cameraman.bmp`: The input image used for testing.
3. `Output Images/Scaling/`: Directory to save scaled images.
4. `Output Images/Rotation/`: Directory to save rotated images.
5. `README.pdf`: This README file.

Functions in the code:

I. Scaling:

1. Nearest Neighbour Interpolation:

```
scale_nn(scale_x, scale_y, img=cam)
```

- Scales the input image by the given factors using nearest neighbour interpolation.
- **Parameters:**
  a. `scale_x`: Scaling factor along the x-axis.
  b. `scale_y`: Scaling factor along the y-axis.
  c. `img`: The input image (default is `cam`).

2. Bilinear Interpolation:

```
scale_bl(scale_x, scale_y, img=cam)
```

- Scales the input image by the given factors using bilinear interpolation.
- **Parameters:**
  a. `scale_x`: Scaling factor along the x-axis.
  b. `scale_y`: Scaling factor along the y-axis.
  c. `img`: The input image (default is `cam`).

II. Rotation:

1. Nearest Neighbour Interpolation:

```
rotate_nn(theta, img=cam)
```

- Rotates the input image by the given angle using nearest neighbour interpolation.
- **Parameters:**
  a. `theta`: Rotation angle in degrees.
  b. `img`: The input image (default is `cam`).

2. Bilinear Interpolation:

```
rotate_bl(theta, img=cam)
```

- Rotates the input image by the given angle using bilinear interpolation.
- **Parameters:**
  a. `theta`: Rotation angle in degrees.
  b. `img`: The input image (default is `cam`).