TEAM AMRIT

Problem 1 - Satellite Image Brightness Normalizer

Team ID - 1

Team Members:

Ganesh Mali (Team Captain)
Ramchadra Potadar
Sandip Sargar

PROBLEM EXPLANATION

Satellite images often vary in brightness due to different lighting or weather conditions. Normalizing brightness makes images consistent, allowing better analysis and decision-making in various fields.

Objectives and Task:

- Normalize the brightness of 10 grayscale satellite images (PNG format, 256x256 pixels).
- Adjust each image’s average intensity to match a given global average (e.g., 127.5).
- Each image’s final average intensity must be within ±1 of the global average.
- Ensure consistent brightness across all images without degrading image quality.

SOLUTION APPROACH

Extract Files

Unzip satellite_images.zip to access 10 grayscale PNGs (image1.png to image10.png).

Read Images

Load each image using OpenCV and convert to grayscale NumPy arrays.

Compute Global Average

Flatten all 10 images into a single array and calculate the mean pixel intensity.

Normalize Each Image

- Compute current image average.
- Calculate scaling factor = global_avg / current_avg.
- Scale all pixels, clip to range [0, 255].

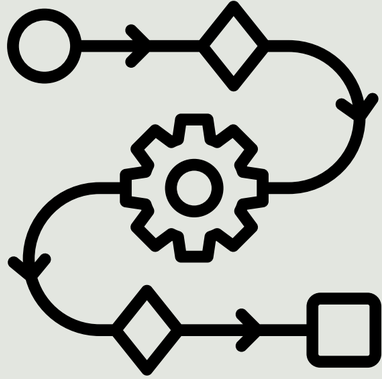
Save Output

Save each image as normalized_imageX.png using PIL.

Validation

Ensure each output image’s average intensity is within ±1 of the global average

WORK FLOW



Step 1

Row code testing on the sample test cases

Step 2

Getting smaple output on row code

Step 3

Matches the results and save aoutput file

Step 4

Verify our outcome image files against there expected image files.

Step 5

Enhance the code readibility, scalability, and modularity.

TOOLS AND PROGRAMMING LANGUAGES

Programming Languages:

We used Python due to its simplicity and strong support for image processing.

Tools:

1. **OpenCv -** To read and Write the images.
2. **NumPy -** To perform the operation on the images.
3. **zipfile -** To extract the images from zip file
4. **git and GitHub -** To clone the dataset and to push the code

IMPLEMENTATION

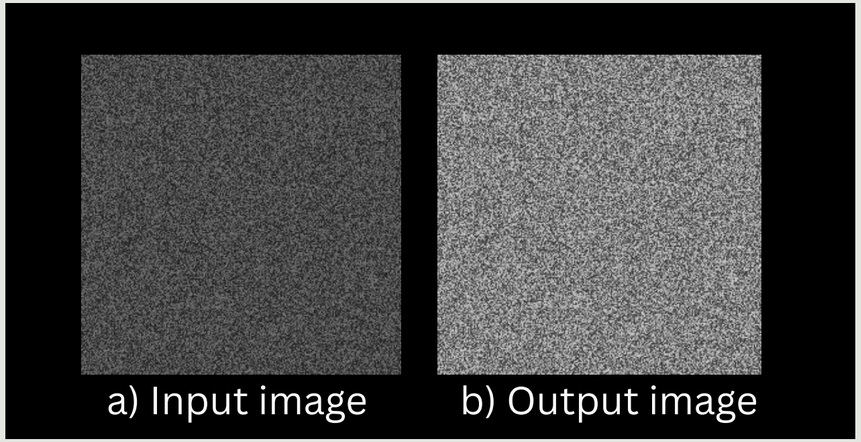
```python
# Normalizing the images and checking whether absolute(diff. bet. each img mean and global_avg ) is <= 1
def NormImages_Output(global_avg, images_load_grayscale, image_indices, missing_images):

    # Step - 1 Converting Images to normalized_Images
    normalized_images = []
    for img in images_load_grayscale:
        current_avg = img.mean()

        factor = global_avg /current_avg

        normalized = img * factor

        normalized = np.clip(normalized,0,255)

        normalized = normalized.astype(np.uint8)

        normalized_images.append(normalized)
```

Normalized each image by calculating the current and global average of intensity and the normalized factor, which is the ratio of global_avg to current_avg.

```python
# Converting Images to GrayScALE using cv2.IMREAD_GRAYSCALE and Calculating Global AVg for each image array
def GlobalAverage():
    images_load_grayscale = []                          # store the image grayscale data
    image_indices = []                                  # Store the Image number which is Available
    missing_input_img = []                              # Store the Misssing Image Number

    for i in range(1,11):
        img = cv.imread(f"test cases/sample_input/image{i}.png", cv.IMREAD_GRAYSCALE)
        """   To read the image file .....
        change the path for hidden test case according to your location
        """

        if img is not None:
            images_load_grayscale.append(img)
            image_indices.append(i)                     # track the present image number AS index
        else:
            missing_input_img.append(i)                 # track the missing image number As index


    all_pixels = np.concatenate([img.flatten() for img in images_load_grayscale])

    global_avg = np.mean(all_pixels)
    return global_avg, images_load_grayscale, image_indices, missing_input_img
```

In the above code we have implimented the error handling by keeping track on the missing and non-missing images.

RESULTS



The input image is normalized to by adjusting the brightness

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SEARCH ERROR

(myvenv) PS S:\Shivanjal\Hackathons\KVGC Hackathons\problem 1> python main.py
Missing Images: []
GLobal Average = 130.20
image1.png: avg=129.71 (within ±1 of 130.20)
image2.png: avg=129.73 (within ±1 of 130.20)
image3.png: avg=129.71 (within ±1 of 130.20)
image4.png: avg=129.70 (within ±1 of 130.20)
image5.png: avg=129.69 (within ±1 of 130.20)
image6.png: avg=129.69 (within ±1 of 130.20)
image7.png: avg=129.69 (within ±1 of 130.20)
image8.png: avg=129.70 (within ±1 of 130.20)
image9.png: avg=129.71 (within ±1 of 130.20)
image10.png: avg=129.67 (within ±1 of 130.20)
Score for Sample Test Case = 10
```

The average intensity of each output image is within a (-1, 1) of the global average.

TEAM CONTRIBUTION

Ganesh Mali - Programming using pil python library, Documentation

Ramchandra Potadar- Programming using open-cv python library, PPT Making

Sandip Sargar - Analyzing Algorithm, Documentation, PPT Making

REFERENCES

- **OpenCV Library:** https://opencv.org
- **Zipping in Python:** https://docs.python.org/3/library/zipfile.html
- **Numpy Library:** https://numpy.org
- **GitHub Dataset:** https://github.com/arshad-muhammad/kvgce-hackwise

Thank You..