İTÜ

Istanbul Technical University
Computer Engineering Department
BLG 453E - Computer Vision - Fall 23/24
Due: 21.10.2023, 23.59

Introduction

In this assignment, you will be tasked with converting RGB images into various grayscale representations and performing local operations such as minimum and maximum filtering with the given filter shapes.

Q1 - Gray-level Scaling

In Question 1, you will be converting an RGB image into a grayscale image by performing a conversion formula, without a given condition.

The formula for converting an RGB image to grayscale typically involves combining the information from the red (R), green (G), and blue (B) color channels to produce a single grayscale value for each pixel in the image. This process aims to represent the image in tones of gray while considering the varying sensitivities of our eyes to different colors. One commonly used method for this conversion is known as the "NTSC[1] formula" (Eq. 1). It assigns different weights to the R, G, and B channels based on how our eyes perceive these colors. No condition for the RGB pixels is involved in this approach.

$$\text{Grayscale Value} = 0.299 * R + 0.587 * G + 0.114 * B \tag{1}$$

For this task,

- Choose an RGB image (prefer *.png*) to process, and read it. (Use the *OpenCV* library to **read** the image. **Make sure to check the color format of the read image**. Is it RGB?)
- Convert the RGB image into grayscale with the given formula. (**No libraries are allowed, including NumPy!** You can use the **round()** function in Python for non-integer values.)
- Save the original RGB and grayscale image array into separate NumPy arrays (*.npy*).
- Add the input and output images to your report. Explain the different tones of gray and their original RGB colors. What are the colors for the lightest and the darkest gray tones?
- Upload the zipped folder that has the input image, output image, the NumPy arrays of the input and output images, and the code file (*.py*).

Q2 - Conditional Scaling

In Question 2, you will be scaling the intensity range of a grayscale image J with conditional scaling to get a new grayscale image J'. In this approach, you modify the current intensity values of an image to have the same mean (μ) and variance (σ) as another reference image I. The formula for this modification can be seen in Eq. 2, and a conditional scaling example can be seen in Figure 1.

$$J' = (\mu_I \times \frac{\sigma_J}{\sigma_I} - \mu_J + J) \times \frac{\sigma_I}{\sigma_J} \tag{2}$$

[1] https://en.wikipedia.org/wiki/NTSC

Figure 1: The sample RGB image, the reference image, the original grayscale output, and the conditionally scaled grayscale output, respectively.

For this task,

- Choose an RGB image (prefer *.png*) to process, and read it. (Use the *OpenCV* library to **read** the image.)
- Convert the RGB image into grayscale with the NTSC formula. (**No libraries are allowed, including NumPy!** You can use the **round()** function in Python for non-integer values.)
- Perform conditional scaling by using the formula in Eq. 2. (**Only functions embedded into Python are permitted.** You can use the **round()** function in Python for non-integer values.)
- Save the original RGB and conditionally scaled grayscale image array into separate NumPy arrays (*.npy*).
- Add the input and output images to your report. Explain the color differences between the NTSC and the conditionally scaled grayscale outputs.
- Upload the zipped folder that has the input image, output image, the NumPy arrays of the input and output images, and the code file (*.py*).

Q3 - Local Operations

Local operations are image processing operations applied to small regions or neighborhoods within an image rather than the entire image at once. These operations could be useful for extracting features, enhancing image details, and performing various tasks such as edge detection, texture analysis, and object recognition.

For this task, you will perform minimum and maximum local operations with the given shaped filter on the input images. An example can be seen in Figure 2.

- Choose an RGB image (prefer .png) to process, and read it. (Use the *OpenCV* library to **read** the image.)
- Convert the RGB image into grayscale using the NTSC formula as in Question 1. (Use the **round()** function in Python for non-integer values.)
- Perform minimum and maximum filtering on the grayscale image with filters shaped as 7x7 and 21x21. (Note: You should use **zero padding** at the edges, and the stride of the filters should be 7 and 21, respectively). (**No libraries are allowed such as OpenCV or NumPy! Perform array processing in Python.**)
- Save the original RGB, the grayscale image, the padded grayscale images and the results of min-max local operation for both of the filter results into separate NumPy arrays (.npy).
- Add the input and outputs to your report and explain the differences in the output images considering the filter sizes and the min-max operations.
- Upload the zipped folder that has the input image, output images, the NumPy arrays of the input and output images, and the code file (.py).

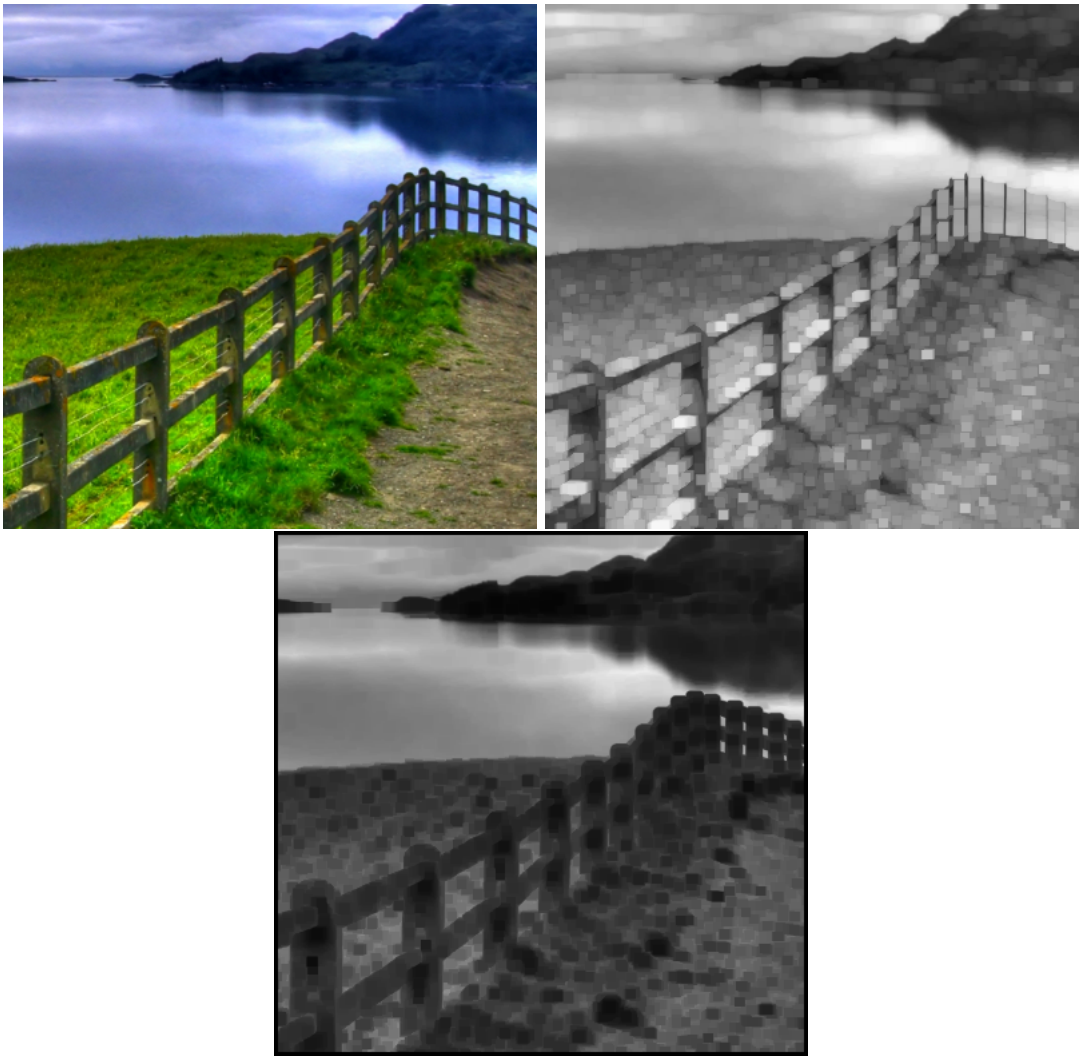

Figure 2: A sample RGB image, maximum and minimum filtered output images by a 7x7 filter, in the clockwise order.