Assignment 2

The purpose of this assignment is to give you some insight into image filtering and edge detection methods, as well as to give you some practical experience in processing images using Python, OpenCV, and NumPy.

Task 1. Implement from scratch the convolution operations for

a. An averaging smoothing filter (filter size: 3*3)
b. A Gaussian smoothing filter (filter size: 7*7, sigma=1, mean=0)
c. The Sobel sharpening filter (filter size: 3*3, both horizontal and vertical)

Task 2. Use the OpenCV built-in functions for

a. An averaging smoothing filter (Use the same size as in Task 1)
b. A Gaussian smoothing filter (Use the same size as in Task 1)
c. The Sobel sharpening filter (Use the same size as in Task 1)

Apply the filters in Tasks 1 and 2 on the image "Cameraman.tif" which will be provided with the assignment resources. Compare the results of your implementations with those of OpenCV. Are there any differences in the results? Why do you think your implementation(s) is/are giving different results, if so? The comparison and discussion should be done in "Assignment 2_Report.pdf".

Task 3. Use the OpenCV built-in functions for

a. The Marr-Hildreth edge detector
b. The Canny edge detector

Apply the edge detectors in Task 3 on "Cameraman.tif". Remember, different parameter values can significantly impact the performance of these edge detectors. Experiment with different range of parameters and choose the ones giving best results. Compare the results of the two edge detectors. Are there any differences in the results? Why do you think the edge detectors are giving different results, if so? Which factors contributed more to the differences – the parameters or the algorithms themselves? The comparison and discussion should be done in "Assignment 2_Report.pdf".

Task 4. Implement from scratch the algorithm to group adjacent pixels in an edge map, as given in the "Edge Detection" lecture slides. Apply your algorithms on the edge maps generated using Tasks 3.a and 3.b. and mark different connected regions in different colours.

Compare the results. Which edge map resulted in more connected components (regions)? Why? Which output represents better features? The comparison and discussion should be done in "Assignment 2_Report.pdf".

Bonus Tasks

1. Make a table to compare the efficiency of from-scratch methods and OpenCV. For each task, enter the computation time for your implementation, OpenCV implementation, and the difference between the two. Why do you think OpenCV implementation is faster? How could you modify your code to get closer to OpenCV efficiency? The comparison and discussion should be done in “Assignment 2_Report.pdf”.
2. Experiment with different kernel sizes for the filters and discuss how the results vary with the kernel size.

Marking Scheme

Task	Marks
Task 1	6
Task 2	3
Task 1 and Task 2 discussion	2
Task 3	4
Task 3 discussion	1
Task 4	4
Task 4 discussion	1
Total	**21**

Deliverables

Your deliverable should include the following material:
D1 – Complete source code. This should be complete, so that it could be easily executed on another system without additional modifications. The code may contain Python file(s) (.py) or Jupyter Notebook(s) (.ipynb).
D2 – Resulting output images. Make sure that you include all output images. Also, make sure to follow the given names for the output images. Non-compliance will result in marks deduction.
D3- Documentation in pdf format. Name the file “Assignment 2_Report.pdf”

Naming Conventions for Output Images

Save all output files as “tif”. Name your output images “tx.tif” where x is the task (and subtask) number. For example, for Task 1, name the three output images as “t1a.tif”, “t2a.tif” and “t3a.tif”.

Instructions

- For the Tasks involving “implementation from scratch”, you may use any NumPy functions but are not allowed to use any high-level OpenCV functions (you may use OpenCV functions for reading, writing, and changing formats of images, if needed).
- The report should not be more than 2 pages of text (excluding the images and tables).
- Submit the material to MLS in a single zip file (described below).
- Inside your root directory, make a separate directory for each deliverable.
- Use Python version > 3.6 and OpenCV version > 3.4
- All code that you submit should be original. Do not scour the Internet for solutions.