CSC 481 – Intro. To Image Processing
2024-25 – Winter Term
Instructor: Kenny Davila Castellanos

Weekly Assignment 4: Edges, Filtering, and more Histograms

Objective

The main objective of this assignment is to help student get familiar with multiple filtering operations that were studied in class. In addition, the homework provides an opportunity to learn more about image segmentation using histograms.

Data

To complete this assignment, please pick 3 images of your liking. You should use:

1) A picture of a street sign (e.g. a road sign or traffic sign). This must be a picture taken by you.
2) A picture of a neon sign. This can be a picture taken by you or from the internet. Please clearly indicate the source on your report.
3) A random picture of your liking different from the first two. Ideally, look for an image that has at least 3 interesting objects of different intensities.

   You will have to run your code with each of these images, and then you must display and briefly discuss the results in your report.

Part 1 – Edge Detection

For each image, do the following main steps:

a) Read and display the image.
b) If not grayscale, make it grayscale.
c) Run the following edge detection algorithms:
   I. Roberts
   II. Canny
   III. Sobel
   IV. Prewitt
d) Some of these algorithms (e.g. Canny) use multiple parameters. You need to explore different values, and try to get the best results for each one of them. Keep in mind that every input image might require a different set of parameter values. In the **report**, briefly describe the parameters that you considered per image.
e) In the **report**, explain which algorithm do you think that worked best, and why do you think so.

You may need to explore a little bit if your image processing library does not directly allow you to perform all four with a single function. Worse case, you do as you did for Assignment 3 and use old fashioned filtering.

For **every input image** (you have 3 inputs, see **Data**), you must show: the input image (grayscale), best results for each of the edge detection methods (4 methods in total). Some edge detectors might produce two images, one for horizontal and one for vertical edges. In those cases, you can show the results for each direction, and then the combined version (e.g. a binary OR).

Part 2 – Edge Filter

Design a 7x7 Sobel filter. The main idea behind the design of a proper Sobel-ish operator is to model the Gaussian derivate in one direction and the Gaussian in the perpendicular direction. In this case, we can consider the gradient in x as follows:

$$\frac{\partial \mathrm{G}(\mathrm{x}, \mathrm{y}, \sigma)}{\partial x} = -\frac{x}{2\pi\sigma^4} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

We can also create another filter for the gradient in y using:

$$\frac{\partial \mathrm{G}(\mathrm{x}, \mathrm{y}, \sigma)}{\partial y} = -\frac{y}{2\pi\sigma^4} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

In this formulation, ***x*** and ***y*** are the distance from the center pixel of your filter. The standard deviation σ becomes a parameter of the filter. You can choose different values, including decimal, and you will obtain very different results with the filter. You can start with σ=1.0, and then play with bigger and smaller values.

For each image, do the following main steps:

a) Read and display the image.
b) If not grayscale, make it grayscale.
c) Show the results of filtering the image using the custom Sobel for the gradient in x.
d) Show the results of filtering the image using the custom Sobel for the gradient in y.
e) Experiment with different values of sigma. Keep in mind that every input image might require a different value for this parameter. In the **report**, briefly describe the parameters that you considered per image.
f) In the **report**, explain the difference between this filter and the Sobel filter used in the previous task.

For **every input image** (you have 3 inputs, see **Data**), you must show: the input image (grayscale), results of the Sobel filter for the gradient of x using best Sigma, and results of the Sobel filter for the gradient of y using the same best Sigma.

Part 3 – Histogram-based Segmentation

Implement histogram-based segmentation on your image as follows:

a) Read and display the image.
b) If not grayscale, make it grayscale.
c) Compute and display the image histogram.
d) Analyze the histogram, and identify how the peaks of your histogram correspond to particular objects. You must specify **at least three** ranges. if your image does not have at least three histogram peaks, then you can make up ranges. In the **report**, please describe the ranges that you picked and provide a brief explanation on why did you pick these ranges (e.g. what objects were you trying to capture).
e) Show the identified objects as binary images for each range.
f) Finally construct the histogram-based segmented image, by combining the binary images.

For **every input image** (you have 3 inputs, see **Data**), you must show: the input image (grayscale), the image histogram, binary image per range (at least 3 images), and final segmented image.

Part 4 – Noise Reduction

The goal of this part is to learn how to add and remove noise from images. In particular, you will need to search how to create two types of noise: “Gaussian Noise” and “Salt and Pepper Noise”. In MATLAB, this can be done using the function “imnoise”. In python, you can find a few useful alternatives. Feel free to check this reference:

https://stackoverflow.com/questions/14435632/impulse-gaussian-and-salt-and-pepper-noise-with-opencv

For each image, do the following main steps:

a) Read and display the image.
b) If not grayscale, make it grayscale.
c) Create a copy of the image and corrupt it with gaussian noise.
d) Create another copy of the image and corrupt it with salt and pepper noise.
e) Use mean filters of different sizes (3x3, 5x5 and 7x7) to clean **each** of the corrupted images.
f) Use median filters of different sizes (3x3, 5x5 and 7x7) to clean **each** of the corrupted images.
g) In the **report**, describe which of the filters worked better for each type of noise.

For **every input image** (you have 3 inputs, see **Data**), you must show: the input image (grayscale), a copy corrupted with gaussian noise, a copy corrupted with salt and pepper noise, and results for each size of each type of filter on each type of noise ([2 filter types] x [3 filter sizes] x [2 noise types] = [12 results]).

The Program

In this assignment, you will write a program that handles multiple operations. You can use any programming language of your preference, but Python is highly recommended. Your code should be properly organized and well-commented to clearly identify the correspondence between portions of the code and each part of this assignment. The code should compile and run on any properly configured programming environment without the need of making significant changes to it. In this sense, it is highly recommended to avoid the usage of absolute paths.

The report

You are asked to submit a written report showing the results for each part of the assignment. Use proper section headings and descriptions to clearly identify the results of each part. If I cannot easily find a match in the report for a required result, I might assume that this portion was not completed, and a score of 0 will be given by default.

For students working with Python, you are allowed to use Jupyter Notebooks. These allow you to combine code with the report in a single deliverable, which is perfectly acceptable for this course. Besides the original Jupyter notebook file, please also submit an export to PDF.

Delivery Instructions

You are given the freedom to use any programming language and IDE of your preference. You are also required to provide your own images, and to produce a written report. When done, **you should submit everything using a single Zip file**.

File names. The zip file that you submit should use "[Last Name(s)], [Given Name(s)].zip" as it appears in D2L. For example, "Kenny Davila Castellanos" (Davila Castellanos is two last names), would have to submit the homework with the name "Davila Castellanos, Kenny.zip". Another student named "Kenny Mauricio Davila" (Mauricio is a middle name), would have to submit the homework as "Davila, Kenny Mauricio.zip". Not following these instructions might lead to a penalty.

Policies

1. All general policies about Plagiarism and Cheating apply to this homework. If you plagiarize or receive code from other people, you will be caught and you will receive a score of 0, and a report of the academic integrity violation will be filed.
2. Please limit the usage of Chat-GPT and other code generators to ethical usage only. Submitting code that was directly generated by these tools is considered a form of plagiarism.
3. Do not post your solutions online and do not share them with anyone. It is your responsibility to safeguard your private data.
4. **Code that does not compile due to syntax and/or semantic errors will automatically receive a score of 0.** It is hard to assign partial credit when I cannot even run your code.
5. You can use any programming language of your choosing.
6. You must follow the delivery instructions.
7. Very late homework's without justification will receive a score of 0.
8. The homework description outlines very specific requirements. You are welcome to try other things and report your results. However, no amount of extra work can be used as a substitution of the actual requirements.
9. Do ask for help if anything is unclear, but do it in a timely manner (e.g., by e-mail, Discord or during the Office Hours).