MEE / MEEE

Visão Computacional / Computer Vision

Assignment 1 – Image Filtering

In this assignment you will be implementing some basic image processing algorithms to study and compare its application for several images that are included for you to test. Like most vision algorithms, the image filters use a number of parameters whose optimal values are (unfortunately) data dependent, that is, a set of parameter values that works really well on one image might not be best for another image. By running your code on the test images, you will learn about what these parameters do and how changing their values effects performance. Many of the algorithms you will be implementing as part of this assignment are functions in the OpenCV library, so it is advisable to use them.

Instructions

a) **Integrity and collaboration:** Students are encouraged to work in groups, so include the names of your collaborators in your write up. Code should **NOT** be shared or copied among groups. Plagiarism is strongly prohibited and may lead to failure of this course.

b) **Start early!** Especially those not familiar with Python and Opencv.

c) **Write-up:** Your write-up should mainly consist of two parts, the resulting images of each step upon changing of the filter's/function's parameters, that is, the output of each tasks below, and the discussions for experiments. Please note that handwritten scans for your write-up in this assignment are **NOT** accepted. Please type your answers discussions for experiments electronically.

d) **Submission:** Your submission for this assignment should be a zip file, **<CVis_2021_Assign1_JohnDoe_JaneDoe.zip>**, composed of your write-up, your Python implementations (including any helper functions), results for extra credit (optional). Please make sure to remove any other temporary files you generated. Substitute both in the folder name and in the filename "**JohnDoe**" and "**JaneDoe**" by each of the group elements name and surname.

Your final upload should have the files arranged in this layout:

<CVis_2021_Assign1_JohnDoe_JaneDoe.zip>

- < JohnDoe_JaneDoe_ Assignment1>.pdf => this is your report.
- **data** => location of the images that you are using.
- **code_python**: <1.1; 1.2; 1.3; my_Harris> subfolders. => location of the scripts that you will develop. Each subfolder should only contain the implementation for the specific part of the assignment, presented below in the document.
- **papers**: <1.1; 1.2; 1.3> subfolders. => location of the papers, documents, etc., that you used (if it is the case) for the specific part of the assignment, presented below in the document.

e) **File paths:** Please make sure that any file paths that you use are relative and not absolute. Don't use cv2.imread('C:/name/Documents/assign1/data/xyz.jpg'), use cv2.imread('../data/xyz.jpg') instead.

f) **Image:** The image that you should use in your assignment corresponds to the number of your group in the google docs, check it here.

1. Implementation

1.1 Noise removal

This task is to eliminate some unknown noise from a given image. You should investigate which filter is the most appropriate (varying their parameters, if existent) in order to achieve the proposed objective.

The application of classic filters is suggested as a first approach (mean filter, median filter, gaussian filter, etc.), but implementations with other approaches that can improve the results when compared to the classical methods will be valued. In this case you should mention in the write-up which articles you consulted (**add the article pdf file in papers/1.1/ folder**), explain briefly the method and explain also what the relevance to the image under study is.

Every script and function that you write or use in this section **should be included in the code_python/1.1/ folder** and **explicitly mention how to run it**. Please include resulting images in your write-up.

1.2 Gradient Operators to edge extraction

This task is to extract and enhance the edges from a given image. You should investigate which gradient-based filters are the most appropriate (varying their parameters if existent) to achieve the proposed objective.

The application of classic gradient detectors is suggested as a first approach (Sobel, Scharr, Prewitt, Roberts, Canny, etc.), but implementations with other approaches that can improve the results when compared to the classical methods will be valued. In this case you should mention in the write-up which articles you consulted (**add the article**

pdf file in papers/1.2/ folder), explain briefly the method and explain also what the relevance to the image under study is.

Every script and function that you write or use in this section **should be included in the code_python/1.2/ folder** and **explicitly mention how to run it.** Please include resulting images in your write-up.

1.3 Gradient Operators to corner detection

This task is to extract and enhance the corners from a given image. You should investigate which gradient-based filters are the most appropriate (varying their parameters if existent) in order to achieve the proposed objective.

The application of classic gradient detectors is suggested as a first approach (Harris for example), but implementations with other approaches that can improve the results when compared to the classical methods will be valued. In this case you should mention in the write-up which articles you consulted (**add the article pdf file in papers/1.3/ folder**), explain briefly the method and explain also what the relevance to the image under study is.

Every script and function that you write or use in this section **should be included in the code_python/1.3/ folder** and **explicitly mention how to run it**. Please include resulting images in your write-up.

1.4 Make you own Harris detector function

Make your own implementation of Harris detector using the “Harris.jpg” image **from data/ folder** and compare your output to the output generated by the OpenCV functions to make sure you are on the right track. Include images showing the output and some of the intermediate steps. Submit your scripts in /. Please include resulting images in your write-up.

2. Experiments

Create a script to run each of the proposed tasks above (make sure it runs correctly) and generate intermediate output images. Include the set of intermediate outputs for one image in your write-up. Did the code work well on all the image with a single set of parameters? How did the optimal set of parameters vary with images? Which step of the algorithm causes the most problems? Did you find any changes you could make to the algorithm to improve performance? In your write-up, you should describe how well the code worked on different images, what effect do the parameters have and any improvements that can make it work better.

Table 1: Classification weights

Activity 1.1	Activity 1.2	Activity 1.3	Own implementations
30%	30%	30%	10%