WorkShop7

Contents

WorkShop7

Instructions:

- The workshop can be completed **in groups (maximum of four members per group (recommended))**.
- All group members should work together, and they will receive the same mark.
- This workshop is worth 2.5% of the total course grade, and it will be graded out of 25 marks and evaluated through your written submission, as well as the lab demo as follows:
    - 25 marks (2.5% of the total course grade)
        - 15 out of 25 (60%): Blackboard submission
        - 10 out of 25 (40%): Lab demo during the lab session
- Please submit the submission file(s) through Blackboard. **Only one person must submit for the group; only the last submission will be graded**.
- During the lab demo, group members are ***randomly*** selected to explain the submitted solution.
- **Group members who do not present during the lab demo will lose the demo mark.**

WorkShop7

Part One: Description

This workshop is just to ensure that you can perform noise reduction for digital images using different filters: an average, a Gaussian, or a median filter.

Filtering has a wide range of uses in image processing. The biggest reason to use filters is **noise reduction** (which we have called ***Image Restoration*** or ***Image Denoising***).

Read more about image noise and filtering from lecture slides and this online public reference.

Numpy Library (Necessary, because OpenCV uses it in the background) - Numpy help:

a) Python Numpy Tutorial (with Jupyter and Colab): https://cs231n.github.io/python-numpy-tutorial/
b) Learn NUMPY in 5 minutes - BEST Python Library! : https://youtu.be/xECXZ3tyONo
c) UCSB Numpy Tutorial: https://sites.engineering.ucsb.edu/~shell/che210d/numpy.pdf
d) Numpy Tutorial: A Simple Example-based Guide: https://stackabuse.com/numpy-tutorial-a-simple-example-based-guide/

WorkShop7

Part Two: Workshop Structure

Download the starter code "ws7.zip" for workshop3 from Blackboard

The workshop folder/directory structure is as follows:

- **code**/: directory contains files named as **question<x>.py**. For example, **question1.py**, …. you need to write your code in these python files.
- **data**/: directory contains the input images, videos or other data supplied with this workshop.
- **output**/: directory must be used to store your output from this workshop. Your output can be images, videos, or other file types.
- **zip_submission.py**: This Python file will be used by you to create a zip file containing your code and output. To generate the submission once you are finished, use the following command: "**python zip_submission.py**".
    - Note: **run this command inside the ws7 directory**
    - You must submit this Zip file with your pdf report through Blackboard.
- ***.py:** add any other supporting files you may need to complete your workshop to the **code** directory.

WorkShop7

Part Three: Add Noise to an Image and Then Try to Remove it by Running Image Filtering

Question 1. Write a code in **question1.py** file to do the following tasks:
Note: You can divide your code into several functions.

Step1. Read image "**FlowerA.jpg**" from the data folder with option **cv2.IMREAD_COLOR**.
Step2. View the image on screen using **cv2.imshow**.
Step3. Add **salt-and-pepper** noise to the color image using a **signal-to-noise ratio equal to 0.9** (**SNR=0.9**).
Step4. Display the noisy image produced in the previous step, and then save the noisy image inside the output folder with the name "**<groupNO>_<questionNo>_noisy_9_FlowerA.jpg**".
Step5. Apply the **Mean Filter** (also called **Average** Filter or **Box** Filter) with Mask (also called kernel) of size $\mathbf{3 \times 3}$ on the noisy image produced in **Step3** and set the border type argument to **cv2.BORDER_REPLICATE**. It should be noted that this process is called **image smoothing** or **image blurring**.

Note:

a. The mean, average or Box filter with a mask of size 3×3 is shown below.

$1/9$	$1/9$	$1/9$
$1/9$	$1/9$	$1/9$
$1/9$	$1/9$	$1/9$

b. You can use OpenCV functions: **cv2.blur()**, **cv2.boxFilter()** or **cv2.filter2D()** to apply the mean filter on any given image.

Step6. Display the image produced in **Step5** and then save this image inside the output folder with the name "**<groupNO>_<questionNo>_Mean_noisy_9_FlowerA.jpg**".

WorkShop7

Step7. Apply **Gaussian** filter with Mask (also called kernel) of size $\mathbf{3 \times 3}$ on the noisy image produced in **Step3** and set the border type argument to **cv2.BORDER_REPLICATE**. It should be noted that this process is called **image Gaussian smoothing** or **image Gaussian blurring**.

Note:

a. The Gaussian filter with a mask of size 3×3 is shown below.

$$h_3 = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

b. You can use the OpenCV function **cv2.getGaussianKernel()** to create a Gaussian kernel/mask.
c. You can use OpenCV functions: **cv2.GaussianBlur ()** or **cv2.filter2D()** is used to apply the mean filter to any given image.

Step8. Display the image produced in **Step5** and then save this image inside the output folder with the name "**<groupNO>_<questionNo>_Gauss_noisy_9_FlowerA.jpg**".

Step9. Apply **Median** filter with Mask (also called kernel) of size $\mathbf{3 \times 3}$ on the noisy image produced in **Step3**.
You can use OpenCV functions: **cv2.medianBlur ()**.

Step10. Display the image produced in **Step5** and then save this image inside the output folder with the name "**<groupNO>_<questionNo>_median_noisy_9_FlowerA.jpg**".

Step11. Briefly discuss your approach to adding salt-and-pepper noise to any given image.

>>>>>Write here <<<<<

WorkShop7

Step12. Write a small paragraph to compare the results of the above filters: Mean, Gaussian and Median—which filter works better for denoising (removing the noise) the noisy image.

>>>>>Write here <<<<<

Step13. Repeat the steps of Question 1, but this time add salt-and-pepper noise to the color image using a **signal-to-noise ratio equal to 0.8** (**SNR=0.8**). **Use the following name to save the produced images inside output folder**
"**<groupNO>_<questionNo>_noisy_8_FlowerA.jpg**",
"**<groupNO>_<questionNo>_Mean_noisy_8_FlowerA.jpg**",
"**<groupNO>_<questionNo>_Gauss_noisy_8_FlowerA.jpg**", and
"**<groupNO>_<questionNo>_median_noisy_8_FlowerA.jpg**"

Step14. Write a small paragraph to discuss the difference between adding noise to an image with **SNR=0.9** and **SNR=0.8**. Did you notice any relationship between the filter performance to remove the noise and the amount of noise added to the image? Explain your answer.

>>>>>Write your answer here <<<<<

WorkShop7

Answer the following sub-questions:

Sub-Question 1. What is salt-and-pepper noise, and how does it affect an image's appearance?

>>>>>Write your answer here <<<<<

Sub-Question 2. Describe the method you used to add salt-and-pepper noise to the color image. What considerations did you have to make to achieve a signal-to-noise ratio (SNR) of 0.9?

>>>>>Write your answer here <<<<<

Sub-Question 3. Explain the concept of image smoothing or blurring. Why is it important in image processing?

>>>>>Write your answer here <<<<<

Sub-Question 4. What is the mean (average or box) filter, and how does it work? Describe how you applied this filter to the noisy image.

>>>>>Write your answer here <<<<<

Sub-Question 5. How does the Gaussian filter differ from the mean filter? Describe the process of applying a Gaussian filter to the noisy image.

>>>>>Write your answer here <<<<<

WorkShop7

Part Four: Perform noise reduction

Question 2. Write a code in **question2.py** file to do the following tasks:
Note: You can divide your code into several functions.

Step1. Read image "**peppers.tif**" from the data folder with option **cv2.IMREAD_GRAYSCALE**.
Step2. View the image on screen using **cv2.imshow**.
Step3. Read image "**peppers_noisy.tif**" from the data folder with option **cv2.IMREAD_GRAYSCALE**.
Step4. View the image on screen using **cv2.imshow**.
Step5. Compute the mean-squared error (MSE) between these two images: "**peppers.tif**" and "**peppers_noisy.tif**"

Note: The mean-squared error is given by this formula

$$MSE = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\left(f(x,y) - f'(x,y)\right)^2$$

where $f(x,y)$ is "**peppers.tif**" image and $f'(x,y)$ is "**peppers_noisy.tif**" image

Step6. (**Minimizing the MSE**), We aim to find a **mean filter** that minimizes the mean-squared error between the **filtered image** and the **original** (**noise-free image)**. Use **cv2.blur()** and set the border type argument to **cv2.BORDER_REPLICATE**.
Step7. Apply the **mean filter** with a mask of sizes: $3 \times 3, 5 \times 5, 7 \times 7, 9 \times 9,$ **and** 11×11
Step8. Find a **filter width** that minimizes the mean-squared error. What is this filter width and the corresponding mean-squared error? (**Hint: you might want to plot the mean-squared error as a function of filter width.** Use Matplotlib)

>>>>>Write here, and write the generated MSE here <<<<<

Mask Size	MSE Value
3×3	
5×5	
7×7	
9×9	
11×11	

WorkShop7

Step9. Display the filtered image with the **smallest mean-squared error**, and then save this image inside the output folder with the name "**<groupNO>_<questionNo>_noisy_free_peppers.jpg**".

Step10. Write a paragraph explaining what you see in the above-generated images using different mask sizes. Briefly discuss the procedure that you used to remove the noise from the above noisy image "**peppers_noisy.tif**"

>>>>>Write your answer here <<<<<

Answer the following sub-questions:

Sub-Question 6. Describe the process you used to compute the MSE between the original and noisy images. What functions and methods did you use?

>>>>>Write your answer here <<<<<

Sub-Question 7. Did you encounter any challenges while computing the MSE, applying filters, or saving the images? How did you resolve them?

>>>>>Write your answer here <<<<<

WorkShop7

Deliverables and Group Work

Create workshop report with the following name format

group_<number>_ws_<workshop number>_report.pdf

For example, if **group16** created a report for **workshop20,** then the report name should be

group_16_ws_20_report.pdf

The workshop report should include:

(a) Complete this declaration by adding your names:

We, ---------- (mention your names), declare that the attached assignment is our own work in accordance with the Seneca Academic Policy. We have not copied any part of this assignment, manually or electronically, from any other source including web sites, unless specified as references. We have not distributed our work to other students.

(b) Specify what each member has done towards the completion of this work:

	Name	Task(s)
1		
2		
3		
4		

(c) Read the workshop questions carefully, if the workshop questions (or part of question) asked for output or response then you should include the output images under the question (or part of question) and write response to answer some of the workshop questions (or part of question).

(d) Submit two files
   a. **submission.zip**
   b. **group_<number>_ws_<workshop number>_report.pdf**