

Computer Vision

Exercises of Lab 1

Exercise 1.1: Noise generation and restoration

Download the 'lena.jpg', which is a 512x512 gray-level image.

Add the following noises to the image respectively:

- a) Gaussian noise: Use `imnoise` function to generate the noisy image.
- b) Salt & pepper noise: Use `imnoise` function to generate the noisy image.

Restoration of Noisy Images:

Create 3 × 3

- a) Arithmetic mean filter:

$$\hat{f}(x, y) = \frac{1}{mn} \sum_{(s,t) \in S_{xy}} g(s, t)$$

MATLAB: `f = imfilter(g, fspecial('average', [m n]));`

- b) Geometric mean filter:

$$\hat{f}(x, y) = \left(\prod_{(s,t) \in S_{xy}} u(s, t) \right)^{\frac{1}{mn}},$$

MATLAB: `f = exp(imfilter(log(g), ones(m, n), 'replicate')) .^ (1/(m*n));`

! $\log(a*b*c...) = \log(a) + \log(b) + \log(c) + \dots$

- c) Harmonic mean filter:

$$\hat{f}(x, y) = \frac{mn}{\sum_{(s,t) \in S_{xy}} \frac{1}{u(s, t)}},$$

MATLAB: `(m*n) ./ imfilter(1 ./ (g + eps), ones(m, n), 'replicate');`

Note. Present your resulting images in four 2-by-2 subplots. The first subplot contains the original image and three types of noisy images. In the second subplot to the forth, you should have the noisy image of one type on the top left and filtered images in the following.

Exercise 1.2: Noise cleaning with Weighted Median Filter

Median filter is a non-linear filter that is able to provide better results in white noise:

- a. Load the `Miranda1.tif`, and add some random white noise in the image center of 100×100 in size.

- b. Load the threshold median filter file `medfilt_th.m`. Read the comments in the file and filter the scratched image with your adjusted parameter.
- c. Construct your own median filter in 5×5 neighborhoods. (The matlab built-in function `medfilt2` is not allowed to be used.)
- d. Present the original image, your scratched image, smoothed image by threshold median filter and your own median filter in a 2×2 subplot.

Exercise 1.3: Histogram Equalization

In this exercise, you will apply histogram equalization on images (`moon.png`, `spine.jpg`, `house.png`, `church.png`). Follow the processing steps in the provided code to perform histogram equalization on the above images by using your own method. Compare the histograms and images before and after processing. (for loops are allowed)! Do not use MATLAB `histeq` function! Compare outputs to the `ContrastStretch` results.

Exercise 1.4: Color image histogram equalization

Here you will apply histogram equalization on a color image by representing its colors in the HSV color space.

- a. Load the image `fruits.jpg` and apply histogram equalization on each R, G, and B components. (using the Matlab function `histeq`)
- b. Write a function `intensityeq` which takes as an input a color image in RGB color space, converts it into HSV color space, applies histogram equalization on the brightness component and converts the image back to RGB color space. (`rgb2hsv`)
- c. Load images `fruits.jpg` and `festia.jpg` and apply your function on them