Université de Genève

Départment d'informatique

Advanced Image Processing — Basic Image Processing

TP Class N° 1 — February 28, 2019

1 Noise and Metrics

Additive White Gaussian Noise

AWGN is a type of *channel* in which the discrete channel output Y_i at some time event with index i is the sum on the input X_i and noise Z_i, where Z_i is independently and identically-distributed (i.i.d.) from a zero-mean Gaussian distribution with variance σ^2. Formally:

$$Z_i \sim \mathcal{N}(0, \sigma^2),$$
$$Y_i = X_i + Z_i \sim \mathcal{N}(X_i, \sigma^2). \tag{1}$$

Salt & pepper Noise

Salt & pepper noise can be defined as follows. Let there be a $N \times M$ gray scale image whose datatype supports a value range of $\{s_{min}...s_{max}\}$. Let $\mathbf{y}$ denote the noisy image and $\mathbf{x}$ the original image. Then the observed gray level in image $\mathbf{y}$ at pixel location (i, j) is given by:

$$y_{i,j} = \begin{cases} s_{min} & \text{with probability } p, \\ s_{max} & \text{with probability } q, \\ x_{i,j} & \text{with probability } 1 - p - q. \end{cases} \tag{2}$$

Mean Squared Error

The Mean Squared Error (MSE) between two images $\mathbf{x}$ and $\mathbf{y}$ is defined as:

$$\text{MSE} = \frac{1}{N \cdot M} \sum_{i=1}^{N} \sum_{j=1}^{M} (y[i,j] - x[i,j])^2, \tag{3}$$

where N and M are the width and the height of image $\mathbf{x}$ and $\mathbf{y}$.

Peak Sigin annal to Noise Ratio

The Peak Signal to Noise Ratio (PSNR) is defined as:

$$\text{PSNR} = 10 \log_{10} \left(\frac{\alpha^2}{\text{MSE}(\mathbf{x}, \mathbf{y})} \right), \tag{4}$$

where α is the maximum value possible with the `type` of $\mathbf{x}$ and $\mathbf{y}$. The unit of the PSNR is dB.

Exercise 1.

(a) Write a function that determines the Mean Squared Error (MSE) between two images $\mathbf{x}$ and $\mathbf{y}$.

(b) Read in a new copy of the image `cameraman.tif`, keep it in its original datatype and range, i.e. `uint8` and $\{0..255\}$.

(c) Now read in a second copy of the image `cameraman.tif` but map it to `double` and $\{0..1\}$. See Matlab `im2double`. Compare the two images using the MSE. Can you explain the result?

Exercise 2.

(a) Refractor the PNSR definition such that the PSNR is expressed as a function of the noise variance σ_z^2. You may assume that $\sigma_z^2 = MSE(\mathbf{x}, \mathbf{y})$.

(b) Add Gaussian noise to an image such that the PSNR ratio with the original image is 10dB, 20dB, 30dB and 40dB. Use `randn`, **not** `imnoise`.

(c) Show the noisy images on the screen. How do they look?

(d) Show the histograms for these noisy images, can you explain what you see?

(e) Add salt & pepper Noise to an image until the PSNR ratio between the original and the noisy image is 40 dB. Visually compare it to the 40dB noisy image to which Gaussian noise was added. What can you conclude?

2 Singular Value Decomposition (SVD)

The singular value decomposition (SVD) is a factorization of a real or complex matrix. It is the generalization of the eigenvalue decomposition. Suppose A is a $n \times d$ matrix of real numbers. Then there exists a factorization of A in the form:

$$A = USV^T, \tag{5}$$

where

- U is a $n \times n$ orthogonal matrix;
- S is a diagonal $n \times d$ matrix with non-negative real numbers on the diagonal;
- V is a $d \times d$ orthogonal matrix.

Low-Rank Approximations from the SVD

Given an $n \times d$ matrix A and a target rank $k \geq 1$, we produce a rank-k approximation of A as follows (see also Figure 1):

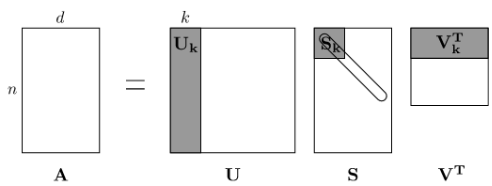


Figure 1: Low rank approximation via SVD.

1. Compute the SVD of A : $A = USV^T$;
2. Keep only the top k right singular vectors: set V_k^T equal to the first k rows of V^T (a $k \times d$ matrix);
3. Keep only the top k left singular vectors: set U_k equal to the first k columns of U (a $n \times k$ matrix);

4. Keep only the top k values: set S_k equal to the first k rows and columns of S (a $k \times k$ matrix);

The computed low-rank approximation is then:

$$A_k = U_k S_k V_k^T. \tag{6}$$

Exercise 3. Read the image `peppers.png` and convert it to grayscale. Perform its low-rank approximation for $k = 1, ..., n$. Plot the dependence between the k and MSE of k-rank approximation version of original image. Make a conclusion.

Exercise 4. Read the image `peppers.png` and convert it to grayscale and add Gaussian noise $\mathcal{N}(0, 625)$. Perform its low-rank approximation for $k = 1, ..., n$. Plot the dependence between the k and MSE of k-rank approximation version of original image. Make a conclusion.

3 Noise Visibility Function (NVF)

The Noise Visibility Function (NVF) describes noise visibility in an image. The most known form of NVF is given as:

$$NVF(i,j) = \frac{1}{1 + \theta \sigma_x^2(i,j)}, \tag{7}$$

$$\theta = \frac{D}{\sigma_{x_{max}}^2},$$

where $\sigma_x^2(i,j)$ denotes the local variance of the image in a window centred on the pixel with coordinates (i,j), θ plays the role of contrast adjustment for every particular image, $\sigma_{x_{max}}^2$ is the maximum local variance for a given image and D is an experimentally determined parameter.

The final embedding equation is:

$$y_{i,j} = x_{i,j} + (1 - NVF) z_{i,j}, \tag{8}$$

where $z \sim N(0, \sigma_z^2)$ denotes a watermark.

Exercise 5.

(a) Read the image `lena.png` and convert it to grayscale.

(b) Add a watermark to the image with and without applying NVF function the different values of σ_z^2 (10, 25, 50, 75) and D. Choose the window size appropriate to used image. What can you say about the impact of NVF function?

(c) Report the dependency between the parameters σ_z^2, D and original image.

4 Image enhancement

High Dynamic Range (HDR)

The HDR processing allows to improve the quality, by merging together several images or some parts of them. That allows to reduce the noise level, boost the local contrasts and make the image look dramatic. Like for example:

[1] `https://en.wikipedia.org/wiki/High-dynamic-range_imaging`

(a) (b) (c) (d) (e)

Figure 2: Image (e) is the results after combining and processing the images (a) - (d)[1].

Exercise 6.

(a) You are given a set of images `hdr_images`. Combine the images (not necessary all) to one image in such a way that the result image has higher quality then all given images in the set. You can sum, subtract the images, divide by some constant, multiply by some mask, etc.

(b) Visualise the results and explain how did you obtain them.

RENOIR

You are given a *"renoir"* set of two images (reference and noisy) from the RENOIR dataset[2].

Exercise 7.

(a) Visualise all color channels of both images. Are the all channels equally affected by the noise?

(b) Try to decrease the noise via image down/up sampling.

  (a) Do it for the RGB image. Measure the *PSNR* between the reference and de-noised images [3].

  (b) Do it for the grayscale image. Measure the *PSNR* between the grayscale reference and de-noised images.

  (c) Do the denoising for the RGB image and after convert it to the grayscale. Measure the PSNR. Does the obtained result is different from the (b)? Explain the result.

(c) What other methods can you suggest to improve the noisy image quality?

5 Image segmentation

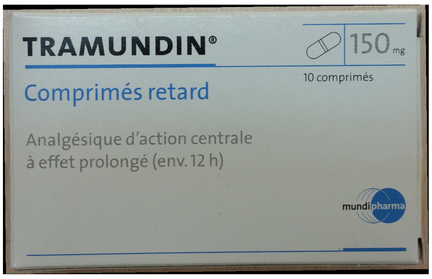
Test segmentation

Exercise 8. You are given a set of 4 images: `tp1_101.png` - `tp1_104.png`. For one of these images perform the segmentation of the text information. See the example in Figure 3. Some graphical elements can be segmented as well.

Hint: use edge detection and image filtering techniques. The next Matlab function can be useful *imdilate*, *imfill*, *bwconncomp*, *regionprops*

[2] `http://ani.stat.fsu.edu/~abarbu/Renoir.html`

[3] The PSNR for color image you can obtain as a mean value of PSNRs between all color channels.



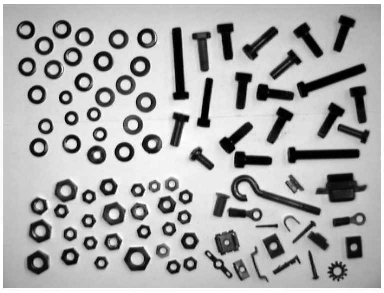
(a) Original image

(b) Segmented image

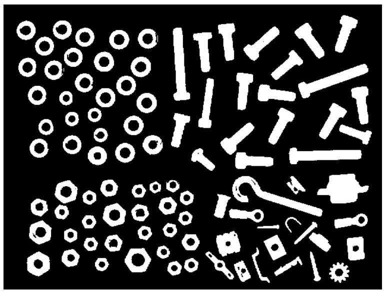
Figure 3: Text segmentation example.

Nuts and bolts

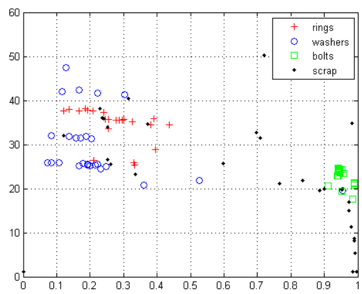
In this little project you will design and test a program that can recognize various nuts and bolts in an image using Matlab's *morphological* functions and a bit of statistics. The image can be seen in Figure 4. Matlab contains an excellent tutorial segmenting and counting rice in an image, which you can work through as preparation.[4]



(a) Source picture



(b) Fore- and background



(c) Separation based on two `regionprops` statistics

Figure 4: Recognizing nuts and bolts.

The principle steps that need to be done are the following:

- Segment the foreground which contains all parts, from the background. You can use

[4] `http://www.mathworks.ch/ch/help/images/image-enhancement-and-analysis.html`

morphological opening, e.g. `imopen` to ascertain background statistics or use the so called *Otsi's* method implemented by Matlab `graythresh`.

- Use morphology to remove any noise from the image
- Select all individual items using Matlab's `bwlabel` and `bwconncomp`.
- To gather statistics deploy Matlab's `regionprops` function. It is capable of collecting a vast amount of information on binary objects which in term can be used to distinguish the various parts from each other.
- Find a combination of metrics to separate the different parts as best as possible.

Exercise 9.

1. Implement the image segmentation and statistics gathering functions
2. Report on what statistics work and why (not).

Submission

Please archive your report and codes in "Name_Surname.zip" (replace "Name" and "Surname" with your real name), and upload to "`Assignments/TP1: Basic Image Processing`" on `https://chamilo.unige.ch` before Wednesday, March 13 2018, 23:59 PM. Note, **the assessment is mainly based on your report, which should include your answers to all questions and the experimental results**.