Assignment : histograms and filtering

1. Histogram-based processing

1.1. In mathematical terms, what is the cumulative distribution function (CDF) with respect to the probability density function (PDF) from the continuous point of view ? What can you infer about the variations of the CDF in general ?

1.2. What are the PDF and CDF of a constant image ? What is the CDF corresponding to a constant PDF ?

1.3. In the discrete setting, implement a MATLAB function that, to a given histogram of a grayscale image, computes its cumulative histogram (you may use MATLAB's *cumsum* function and normalize the expression). Display it for the image *pout.tif*. What do the regions of fast increase of the function correspond to ? And what about its flat regions ?

1.4. Given an image I and its CDF C, write a matlab routine that computes the floating-point image $C(I)$ such that the intensity at each pixel (i, j) is $C(I(i, j))$.

1.5. Is the CDF invertible in general and why ? To overcome the problem of non-invertibility for histogram matching (cf 3.4.5), we can consider instead a pseudo-inverse. For any CDF function f defined on the integer set $\{0, .., 255\}$, we define its pseudo-inverse $f^{(-1)}$ as follows :

$$f^{(-1)}(l) \doteq \min\{s \mid f(s) \geq l\}$$

where $l \in [0, 1]$. Write and add in your report a MATLAB routine that computes the pseudo-inverse of any given CDF (you may use MATLAB *find* and *min* functions).

1.6. Based on equation (3.25) and on the two previous questions, implement your own MATLAB procedure to perform histogram matching between two images and show results. Plot and compare the cumulative histograms of the original image, the target and the one obtained by histogram matching.

1.7. Apply it in the particular case where the target cumulative histogram is the one corresponding to a constant histogram. Compare the result to MATLAB *histeq* built-in function.

Remark. *Instead of matching the histogram of one image on the other, it is often more robust and adequate to match both of them toward a 'midway' histogram of the two images. This is the idea of* ***midway equalization*** *method introduced below, which is notably used for the correction of* ***flicker*** *effects in old movies.*

1.8. Let I_1 and I_2 be two images with cumulative histograms C_1 and C_2. Let's introduce :

$$\phi \doteq \frac{1}{2}\left(C_1^{(-1)} + C_2^{(-1)}\right)$$

and the images $\tilde{I}_1 \doteq \phi(C_1(I_1))$ and $\tilde{I}_2 \doteq \phi(C_2(I_2))$ called the *midway specifications* of I_1 and I_2. If I_1 and I_2 are two constant images with respective values a and b, prove that the midway specifications are both equal to the constant image $\frac{a+b}{2}$. Is the cumulative histogram of the midway specifications equal to the average of the cumulative histograms ?

1.9. Write a MATLAB function that computes the midway specifications of two images. Show the result on the images *movie_.flicker1.tif* and *movie_.flicker2.tif* located in the Test Images folder (convert them to grayscale) which corresponds to two successive frames of the 1948 movie 'Les aventures des Pieds-Nickelés' (copyright Marc Sandberg). How would you generalize midway equalization to an arbitrary number of images ?

2. Image filtering and enhancement

2.1. Consider the following finite difference approximations of the image derivatives given by :

$$\frac{\partial I}{\partial x}(x,y) \approx \frac{I(x+1,y) - I(x-1,y)}{2}$$
$$\frac{\partial I}{\partial y}(x,y) \approx \frac{I(x,y+1) - I(x,y-1)}{2}$$

Adopting the convention of eq.(4.1) page 88, what should be the kernels corresponding to these filters ? And what if you consider instead that the image is convolved by the kernel as in the expression of eq.(2.20) page 35 ? How does MATLAB deal with those two possible conventions (investigate *imfilter* function's help) ? Does it matter for filters like Gaussian or mean ?

2.2. Based upon the separable expressions given in 4.5.2.1, reinterpret the effect of Sobel and Prewitt filters on images : explain in particular why they are more likely to be robust to noise than the simple finite difference approximation of the previous question.

2.3. Compare the effect of mean and median filtering on a noisy version of the image *eight.tif* for salt and pepper as well as gaussian noise. What is the effect of increasing the window size ? Store and plot the different computational times obtained for N=1 to N=25 and each time for 100 executions (use MATLAB functions *tic* and *toc*). Comment.

2.4. Consider a Gaussian filter of fixed standard deviation $\sigma = 5$ and filter the image with increasing value of N. What do you observe when N is big enough and how can it be explained ?

2.5. Experiment filters of increasing σ's, choosing at each iteration $N > 3\sigma$ for the window size and comment on the effect of denoising vs edge accuracy.

3. Bonus questions

In order to better preserve edges when filtering noise, an alternative popular technique is called **bilateral filtering**. Its principle is to do local averaging both in the spatial and photometric domains. Given a Gaussian function $f_\sigma(x,y) = \exp(-\frac{x^2+y^2}{2\sigma^2})$ on the 2D spatial domain of the image and a Gaussian function $g_\tau(u) = \exp(-\frac{u^2}{2\tau^2})$

on the 1D photometric domain, the filtered image $\tilde{I}$ writes :

$$\tilde{I}(x,y) = \frac{\sum_{i=i_{\min}}^{i_{\max}} \sum_{j=j_{\min}}^{j_{\max}} \omega(x,y,i,j) I(x+i,y+j)}{\sum_{i=i_{\min}}^{i_{\max}} \sum_{j=j_{\min}}^{j_{\max}} \omega(x,y,i,j)}$$
$$\text{with } \omega(x,y,i,j) = f_\sigma(i,j).g_\tau(I(x+i,y+j) - I(x,y))$$

3.1. Is it a linear filter ? What term differs from a usual Gaussian filter and what interpretation do you give to parameter τ ? In what limit case does bilateral filtering becomes usual Gaussian filtering ?

3.2. Implement a MATLAB function that takes as input a grayscale image, a window size N, standard deviation parameters σ and τ and returns the filtered image (beware to restrict the neighborhoods for boundary pixels) : add the code to your report.

3.3. Apply it on the image *eight.tif* corrupted by Gaussian noise, for various σ and τ. Compare in addition the output of edge detection's methods after the image is filtered by a Gaussian vs a bilateral filter.