M2 AIC: Introduction au traitement d'images

Yohann Tendero[*]

Nom1:
Nom2:
Nom3:

Étape 0): Aller sur la page `http://perso.telecom-paristech.fr/~ytendero/https://perso.telecom-paristech.fr/ytendero/master_aic.html` et télécharger le fichier TP_1.zip et éventuellement les slides du cours.
Un compte rendu sur papier devra être rendu en fin de séance, il comportera notamment les commandes que vous avez complétées et vos commentaires/analyses.

Preambule

Q1. Rappeler les commandes utilisées pour lire une image et l'afficher

Q2. Donner les lignes de commandes utilisées créer une image en niveaux de gris de taille 512 x 512 dont les pixels $\{200, \ldots, 250\}^2$ sont blancs et les autres noirs.

Q3. Requantification : donner la formule pour J en fonction de N:

$J = \ldots$

Q4. Empiriquement à partir de quelle valeur de N (nb de niveaux pour chaque canal couleur) commencez vous à percevoir une différence entre l'image originale et l'image re-quantifiée ?

Q5. Image de "bruit". Creer une image niveaux de gris de taille 512 x 512 dont chaque pixel est la réalisation d'une v.a. Gaussienne i.d.d. de moyenne $\frac{1}{2}$ et de variance σ^2. Visualiser cette collection d'images pour quelques valeurs σ^2.

[*]yohann.tendero@telecom-paristech.fr

From an article to a code

We'll do the implementation of the bilateral filter in its simplest form. The bilateral filter produces its output from an weighted combination of neighboring pixels. The spatial and intensities kernels will be Gaussian functions with standard-deviation σ_s and σ_i respectively. The window defining the neighborhood will be a square window.

- The image is defined on the finite grid
$$D = \{1, \ldots, N\} \times \{1, \ldots, M\} \subset \mathbb{N}^2. \tag{1}$$
- $p = (p_1, p_2) \in D$ is the pixel to be processed
- $u : D \to [0, 1]$ is the image, so $u(p)$ is the value of the pixel p
- $f_s(z) := \exp(\frac{-z^2}{2\sigma_s^2})$, $z \in \mathbb{R}$, is the Gaussian spatial kernel (it measures how far two pixels are), σ_s is a parameter
- $f_i(z) := \exp(\frac{-z^2}{2\sigma_i^2})$, $z \in \mathbb{R}$, is the Gaussian intensity kernel (it measures how two gray-values differ), σ_i is a parameter

We want to denoise the pixel $p \in D$, far enough from the boundaries of D. To this aim we define:

- the square of size $(2w + 1) \times (2w + 1)$ (neighborhood) centered around p defined by
$$\Omega(p) := p + \{-w, \ldots, w\}^2 = \Big\{y = (y_1, y_2) : y_1 \in \{p_1 - w, \ldots, p_1 + w\},\ y_2 \in \{p_2 - w, \ldots, p_2 + w\}\Big\}, \tag{2}$$
for p far enough from the boundaries of D. The number $w \in \mathbb{N}^+$ is a parameter.

The output of the bilateral filter is (we shall define the effective domain of p later on):

$$\begin{aligned} u_{denoised}(p) &:= \frac{1}{C} \sum_{y \in \Omega(p)} u(y) f_s\left(\|y - p\|_{\ell^2}\right) f_i\left(|u(y) - u(p)|\right) \qquad (3) \\ &= \frac{1}{C} \sum_{y_1 = p_1 - w}^{p_1 + w} \sum_{y_2 = p_2 - w}^{p_2 + w} u(y_1, y_2) \exp\left(-\frac{(y_1 - p_1)^2 + (y_2 - p_2)^2}{2\sigma_s^2}\right) \exp\left(-\frac{[u(y_1, y_2) - u(p_1, p_2)]^2}{2\sigma_i^2}\right), \end{aligned}$$

where the normalization constant is given by

$$\begin{aligned} C : &= \sum_{y \in \Omega(p)} f_s\left(\|y - p\|_{\ell^2}\right) f_i\left(|u(y) - u(p)|\right) \qquad (4) \\ &= \sum_{y_1 = p_1 - w}^{p_1 + w} \sum_{y_2 = p_2 - w}^{p_2 + w} \exp\left(-\frac{(y_1 - p_1)^2 + (y_2 - p_2)^2}{2\sigma_s^2}\right) \exp\left(-\frac{[u(y_1, y_2) - u(p_1, p_2)]^2}{2\sigma_i^2}\right). \end{aligned}$$

To denoise the entire image, loop over $p \in D$ (and therefore $\Omega(p)$) so that $\Omega(p) \subset D$ (as usual, be careful with the boundaries) and apply the above formulas.

Part 1. Implementation
Q1. Compute the range of $y - p$ when $y \in \Omega(p)$:

Q2. We recall that s and w are fixed. We consider the function $\Omega(p) \ni y \mapsto f_s\left(\|y - p\|_{\ell^2}\right)$. Compute the range of f_s. Hint: Use **Q1.**.

Q3. Which part of u do we need to compute $u_{denoised}(p)$?

Q4. From (3) compute the range of p_1 and p_2, i.e., the p_1 and p_2 for which (3) is valid

Q5. We recall that in Matlab, the first index of a vector is 1. Rewrite (3) so that the summation indexes starts at 1.

Q6. Rewrite the formula you obtained for **Q5** using the function S et $\tilde{u}$ given by

$$S : \{1, \dots, 2w+1\} \times \{1, \dots, 2w+1\} \ni (x_1, x_2) \mapsto \exp\left(-\frac{(x_1 - w - 1)^2 + (x_2 - w - 1)^2}{2\sigma_s^2}\right), \tag{5}$$

$$\tilde{u} : \{1, \dots, 2w+1\} \times \{1, \dots, 2w+1\} \ni (x_1, x_2) \mapsto u(x_1 + p_1 - w - 1, x_2 + p_2 - w - 1). \tag{6}$$

Q7. Deduce the similar formula for C defined in (4).

Q8. List the objects needed to implement the program (Matrices of fixed size/varying size, matrices constant w.r.t the loop over p, real numbers, etc.)

Q9. Deduce the pseudo code of the program.

Part 2. Short analysis of the behavior of the bilateral filter
Q10. What is the behavior of the bilateral filter when $\sigma_i \to +\infty$?

Q11. What happens when $\sigma_i \to 0$?

Q12. What happens when $\sigma_s \to 0$?

Q13. What happens when $\sigma_s \to +\infty$?

Part3. Programming Complete the file bilateral.m and test your program. Write your program below.