Curso : Visión Computational 1
Año : 2024

Assignment 4

Exercice 1 (1 pt). ***(Property of the convolution operator)***

Let F be the operator defined, for $f \in L^1(\mathbb{R})$, by

$$F(f)\colon u \longmapsto \int_{\mathbb{R}} f(x)e^{-2i\pi xu}\,dx,$$

where i the imaginary complex number.

Given $f, g \in L^1(\mathbb{R})$, show that $F(f * g) = F(f)F(g)$.

Exercice 2 (3 pts). ***(Noise removal with median filter)***

*The **median filter** is an image processing technique which consists of, for each pixel of a grey-level image, calculating the median value in a neighborhood of the pixel for some window size determined previously by the user (See Fig. 0.1). This is a well-known simple technique to remove the noise in images.*

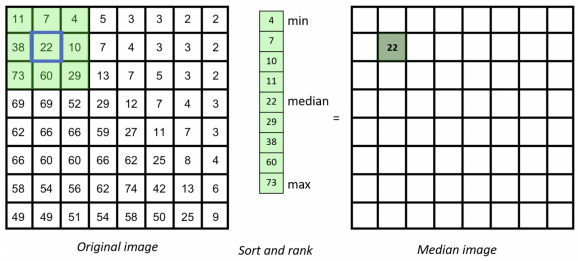

FIGURE 0.1: Median filter at some pixel for a 3x3 window size.

The aim of this exercise is to determine the best strategy to denoise a color image using the median filter.

1. Implement the median filter technique for one-channel images :
- inputs : a one-channel image and a window size.
- output : denoised image.
For simplicity, do not apply the filter when it exceeds the image domain.

2. Implement the median filter technique for color images :
- inputs : a color image, three window sizes (one per component) and a color space (RGB or Opp).
- output : the denoised image expressed in RGB after applying the method implemented in 1. to each of the components of the image in the selected color space and with the selected

window sizes.

3. Apply the method implemented in 2. to the image ***noisy_img1.png****. You'll test different color spaces and different window sizes. Evaluate your result by computing the PSNR between the denoised image and the original clean image* ***img1.png****. Which strategy provides the best results ?*

4. Analyze your results.

Exercice 3 (3 pts)**.** ***(Edge-preserving regularization)***
Convolving an image with a Gaussian kernel not only reduces the noise but smoothes the edges as well. The aim of this exercise is to construct a simple edge-preserving image regularization method which generalizes the convolution with a Gaussian.
Let $I_0 = (I_0^1, I_0^2, I_0^3) \colon \Omega \subset \mathbb{R}^2 \longrightarrow \mathbb{R}^3$ *be a color image expressed in the RGB color space. We consider the following regularization operator*

$$I^c(i,j) = \sum_{(k,l)\in\Omega_{(i,j)}} K_\sigma((i,j),(k,l)) I_0^c(k,l) \qquad c = 1,2,3 \tag{0.1}$$

where $\Omega_{(i,j)}$ *is a neighborhood of* (i,j)*, and the kernel* $K_\sigma((i,j),(k,l))$ *generalizes the 2D Gaussian kernel in the sense that* K_σ *is of the form*

$$K_\sigma((i,j),(k,l)) = Ae^{-\frac{d((i,j),(k,l))^2}{2\sigma^2}} \tag{0.2}$$

for some distance function $d \colon \Omega \times \Omega \longrightarrow \mathbb{R}^+$*. The term* A *is the normalization factor, i.e* A *is such that*

$$\sum_{(k,l)\in\Omega_{(i,j)}} K_\sigma((i,j),(k,l)) = 1.$$

1. Determine a distance function d *such that :*
- $d((i,j),(k,l))$ *is small if the pixels* (i,j) *and* (k,l) *are close to each other and the color vectors* $I_0(i,j)$ *and* $I_0(k,l)$ *are close to each other.*
- $d((i,j),(k,l))$ *is high if the pixels* (i,j) *and* (k,l) *are far from each other and the color vectors* $I_0(i,j)$ *and* $I_0(k,l)$ *are far from each other.*

Reminder : a distance function satisfies three properties : identity, symmetry and triangular inequality.

2. Apply the operator (0.1) to the image ***img2.png*** *where :*
- the distance d *in (0.2) is the one constructed in question 1.*
- the kernel K_σ *in (0.2) has a support of size* 5×5*, and* $\sigma = 10$*.*
- The RGB values of I_0 *are normalized (i.e. the values are in the range [0,1]).*

Exercice 4 (3 pts). ***(Demosaicking)***
The camera sensor produces an image in which for each pixel we only get one of the image channel intensity values (either red, green or blue) as we can see in Fig. 0.2.

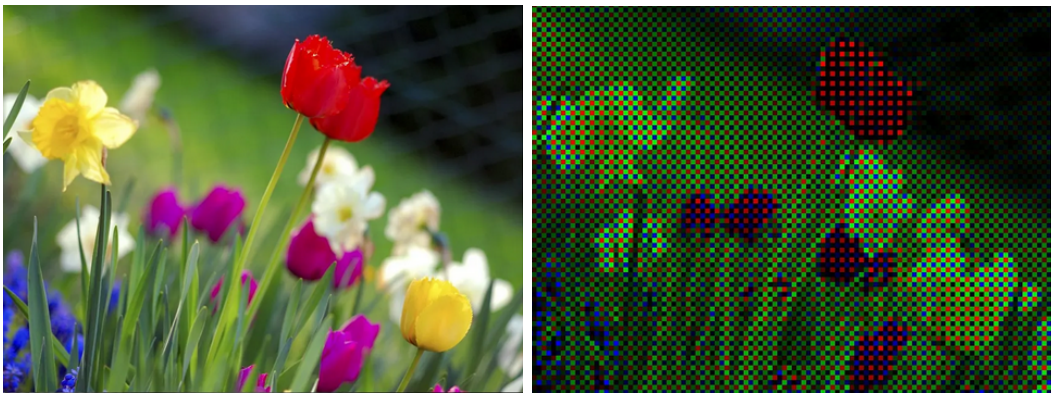

FIGURE 0.2: Left : Original scene - Right : Output of camera sensor

It is then required to find an estimate of the other two missing values to obtain RGB value at each pixel. This is done by an interpolation process called ***demosaicking****, which produces an image with 3 channels. (see Fig. 0.3 and Fig. 0.4).*

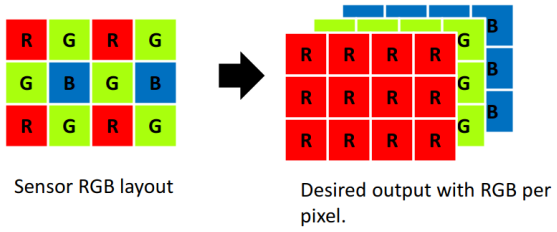

FIGURE 0.3: From camera sensor output to RGB image

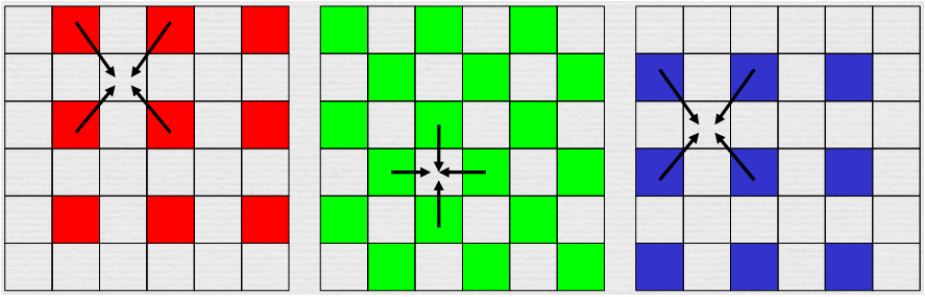

FIGURE 0.4: General principal of interpolation

The following figure describes a simple interpolation method. Here, the given pixel has only the red channel intensity, and green and blue values at this pixel are interpolated according to the 2 formulas.

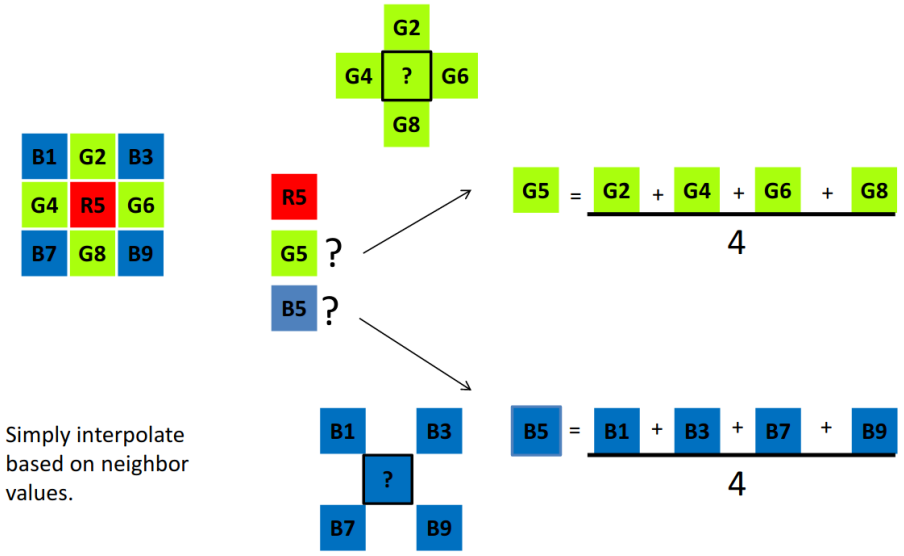

FIGURE 0.5: Simple demosaicking method

The aim of this exercise is to simulate the demosaicking stage performed in cameras.

1. Implement the demosaicking method described in Fig. 0.5. For simplification, do not perform interpolation at the boundary of the image domain (set the boundary values of the demosaicked image to (0,0,0)).
2. Apply the interpolation to the images ***img3.png*** *and* ***img4.png****.*