جـــامعة هواري بومـــدين للعلـــوم و التكنـــولوجيـــا
Université des Sciences et de Technologie Houari Boumediene
Faculté d'Informatique

Année 2022/2023
Master Informatique Visuelle
Vision Artificielle

Série de travaux pratiques n°1
Vision Artificielle

Exercice 1.

Le code du prog1.py permet d'appliquer la convolution d'une image avec un filtre Gaussien : $g(x,y) = f(x,y) \otimes G_\sigma$

1- Transformer le code fourni en prog2.py qui réalise les itérations suivantes :

$g(x,y) = f(x,y) \otimes G_\sigma$
For i=1 to n Do
    $g(x,y) = g(x,y) \otimes G_\sigma$
EndDo

2- Sachant que $g(x,y) = (f(x,y) \otimes G_{\sigma 1}) \otimes G_{\sigma 2} = (f(x,y) \otimes G_\sigma)$ où $\sigma = \sqrt{\sigma 1^2 + \sigma 2^2}$
Vérifier cette relation moyennant les deux programmes.
La valeur de sigma est communiquée comme paramètre à la fonction : cv.GaussianBlur. Voir en fin de la série la syntaxe de l'appel à la fonction.
A Gaussian kernel followed by a convolution with again a Gaussian kernel is equivalent to convolution with the broader kernel. Of course we can concatenate as many blurring steps as we want to create a larger blurring step.
[Réf: https://pages.stat.wisc.edu/~mchung/teaching/MIA/reading/diffusion.gaussian.kernel.pdf.pdf]

Exercice 2.

Mise en œuvre de « Laplacian scale space »
Il s'agit d'appliquer le Laplacien de Gaussienne à une image avec différentes valeurs de sigma, allant de la petite valeur à la plus grande $\sigma \times k^n$. N étant le nombre de niveaux sans l'espace d'échelles, k=1,24

Pour cela, nous utiliserons la fonction de la librairie python-opencv :
cv.Laplacian(src, ddepth[, ksize[, scale[, delta[, borderType]]]])
# src - input image
# ddepth - Desired depth of the destination image.
# ksize - kernel size

Qui réalise les opérations suivantes :

- Appliquer le lissage Gaussien puis le Laplacien

- Trouver les passages par zéro dans l'image
- Appliquer un seuil pour sélectionner les passages par zéro forts.
  Ceci est équivalent à
- Appliquer la convolution à l'image avec le noyau LoG directement
- Trouver les passages par zéro dans l'image
- Appliquer un seuil pour sélectionner les passages par zéro forts.

$$LoG(x,y) = -\frac{1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\sigma^2}\right] e^{-\frac{x^2+y^2}{2\sigma^2}}$$

La troisième approche consiste à appliquer la différence entre les images convoluées avec des filtres gaussiens à différentes valeurs de sigma.

DoG est équivalent à LOG.

En utilisant la librairie opencv-python, réalisez les tâches suivantes :

1. Lire une image
2. Appliquez un lissage avec un filtre Gaussien de valeur sigma 1
3. Appliquez un lissage avec un filtre Gaussien de valeur sigma 2.
4. Appliquez un lissage avec un filtre Gaussien de valeur sigma 3
5. Calculer la différence pour chaque paire d'images résultat.

Annexe:

void cv::GaussianBlur(InputArray src, OutputArray dst, Size ksize, double sigmaX, double sigmaY = 0, int borderType =BORDER_DEFAULT)

Python:
cv.GaussianBlur(src, ksize, sigmaX[, dst[, sigmaY[, borderType]]]) -> dst
#include <opencv2/imgproc.hpp>

Blurs an image using a Gaussian filter.

The function convolves the source image with the specified Gaussian kernel. In-place filtering is supported.

Parameters

src	input image; the image can have any number of channels, which are processed independently, but the depth should be CV_8U, CV_16U, CV_16S, CV_32F or CV_64F.
dst	output image of the same size and type as src.
ksize	Gaussian kernel size. ksize.width and ksize.height can differ but they both must be positive and odd. Or, they can be zero's and then they are computed from sigma.
sigmaX	Gaussian kernel standard deviation in X direction.
sigmaY	Gaussian kernel standard deviation in Y direction; if sigmaY is zero, it is

	set to be equal to sigmaX, if both sigmas are zeros, they are computed from ksize.width and ksize.height, respectively (see getGaussianKernel for details); to fully control the result regardless of possible future modifications of all this semantics, it is recommended to specify all of ksize, sigmaX, and sigmaY.
borderType	pixel extrapolation method, see BorderTypes. BORDER_WRAP is not supported.