Quelques filtres non-linéaires

1 Filtrage médian / Median filtering

Ce filtrage consiste à calculer en chaque point de l'image I la valeur médiane au sein d'un certain voisinage défini par une fenêtre F. Ceci conduit à l'image filtrée J:

$$J(x, y) = median\{I(u, v)|(u, v) \in F(x, y)\}$$

où $F(x, y)$ désigne l'ensemble des points voisins du point (x, y).

English:

Median filtering computes, at each pixel of an image I, the median value within a neighborhood (mask F) of the considered pixel. This leads to the filtered image J :

$$J(x, y) = median\{I(u, v)|(u, v) \in F(x, y)\}$$

where $F(x, y)$ defines the set of neighbors of image point (x, y).

50	50	50	50	51	50	50
50	120	120	121	120	50	49
49	121	120	0	120	129	43
47	121	129	120	129	48	48
50	50	50	50	50	49	44
51	255	51	50	50	48	47
49	51	50	50	50	49	46

Table 1: Image

Table 2: Image

1	1	1
1	1	1
1	1	1

Table 3: Masque (ou fenêtre) considérée pour le filtrage médian / Mask related to median filtering

1.1 Cas image synthétique / Case of a synthetic image

En considérant l'image initiale I (tableaux 1 et 2) et le filtre du tableau 3, quelle est la valeur de $J(4, 1)$?

Appliquer ce filtrage à l'ensemble de l'image I (tableaux 1 et 2), en utilisant la fonction `scipy.ndimage.filters.median_filter`. Le masque peut être spécifié par le paramètre `footprint`, sous forme d'un tableau 2D de 0 et de 1: `mask=numpy.ones((3,3))` dans le cas du tableau 3.

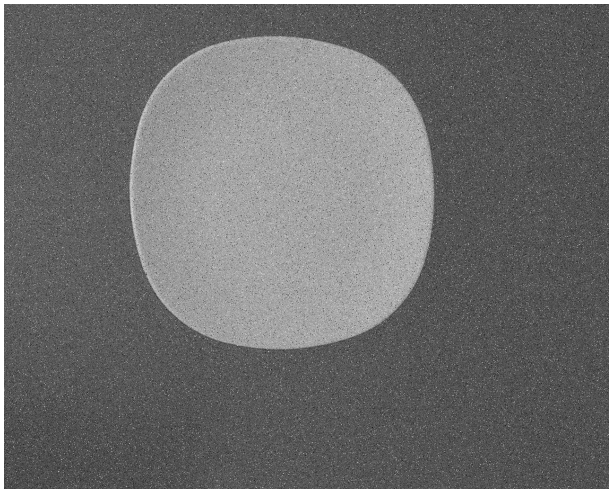

Figure 1: Image

English:
By considering the image I (Tables 1 and 2) and the filter of Table 3, what is the value of $J(4, 1)$?

Apply this median filtering to the entire I image (Tables 1 et 2), by using `scipy.ndimage.filters.median_filter` function. The mask F can be

specified by the parameter `footprint`, as a 2D array of 0 and 1 (`mask=numpy.ones((3,3))` for the case of Table 3).

1.2 Cas image réelle 1 - Real image, case 1

Considérons l'image 1 qui altérée par un bruit impulsionnel: certains pixels sont aléatoirement blancs ou noirs, ce qui se traduit par des pics en 0 et 255 sur l'histogramme. On souhaite segmenter cette image, afin, par exemple d'estimer la surface de l'assiette en pixels (353396 pixels).

1. Appliquer tout d'abord un filtrage "moyenneur" (donc linéaire), afficher l'histogramme, puis seuiller l'image (seuil calculé par la méthode d'otsu). On observe que la segmentation en imparfaite (e.g. pixels du fond à "blanc")

2. Appliquer ensuite un filtrage médian: on constate que le résultat est meilleur.

3. Appliquer ensuite un filtrage médian en augmentant la taille du voisinage considéré pour calculer les valeurs médianes (e.g. 5x5): on constate que le résultat est encore meilleur, avec une surface estimée correcte.

English:
Let us consider image 1, corrupted by an impulse noise: some pixels are randomly completely white (0 value) or black (255 value), leading to peaks in the histogram. One wants to segment this image in order to estimate the area of the plate (353396 pixels).

1. Apply a linear mean filter, display the histogram, and finally threshold the image (threshold determined by Otsu) : the segmentation appears incorrect.

2. Apply a non-linear median instead of the linear one : the segmentation appears better.

3. Increase the size of the mask considered for the median filter (e.g. 5x5 pixels) : the segmentation is further improved, and the estimated plate's area appears more correct.

1.3 Cas image réelle 2 - Real image, case 2

Considérons l'image 2 qui altérée par un bruit impulsionnel. On souhaite segmenter les 2 rondelles, calculer la surface de chacune d'elle (résultat attendu: 3893 pixels et 4503 pixels).

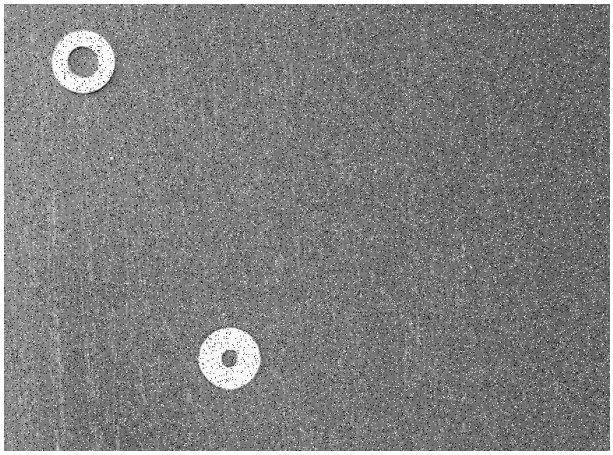

Figure 2: Image

1. Appliquer une filtrage médian (filtre de taille 5x5), suivi d'un seuillage pour segmenter les rondelles (seuil calculé par la méthode d'otsu).
2. Extraire les deux plus grandes composantes connexes (composantes de taille supérieure à 50 pixels), en 8-connexité. Ces deux composantes correspondent aux deux rondelles.
3. Calculer la surface de chacune d'elle.

English:
Let us consider image 2, corrupted by an impulse noise. On wants to extract both washers and estimate the area of each (3893 pixels and 4503 pixels).

1. Apply a median filtering (mask of size 5x5 pixels), and then a thresholding (Otsu threshold determination method).
2. Extract the two largest connected component (sizes larger than 50 pixels - 8-connexity), corresponding to both washers.
3. Estimate the area of each.

2 Filtrage morphologique binaire - Binary Morphological filtering

Il s'agit d'un filtrage s'appliquant sur des images binaires (en "noir et blanc"). Ce type de filtrage, issu de la morphologie mathématique, repose sur des opérations d'érosion et de dilatation, permettant, intuitivement, de rogner ou étendre les objets "blancs" apparaissant dans l'image.

On raisonne est considérant qu'une image binaire est un ensemble X de points (les points "allumés" ou à "1" de l'image).

On définit un ensemble B de pixels que l'on appelle élément structurant (e.g. tableaux 4 et 5), ainsi qu'un point de référence (en gras dans les tableaux 4 et 5). On notera B_p l'élément structurant dont le point de référence (typiquement le centre) est placé au point p de l'image binaire.

Dilatation La dilatation de X par un élément structurant B conduit à l'ensemble $Y = D_B(X)$, obtenu en remplaçant chaque pixel p de X par sa fenêtre B_p :

$$Y = D_B(X) = \{B_p | p \in X\}$$

En pratique, dilater une image binaire revient à placer l'élément structurant en chaque point (x, y) et affecter, au point (x, y) de l'image dilatée, la valeur 1 si l'intersection entre l'élément structurant et "l'objet blanc de l'image" est non nulle. L'effet de la dilatation est, pour les masques considérés, d'élargir la forme binaire initiale (l'élargissement étant fonction de la taille et de la forme de l'élément structurant).

English:
This kind of binary filtering concerns binary images (white and black), and is based erosion and dilation operations. In practice, the dilation of a binary consists in placing the structuring element (mask or filter) at each point (x, y) and to assign, to the (x, y) pixel of the dilated image, 1 if the intersection between the structuring element and the "white" object is not empty. Dilation enables to enlarge binary white regions (enlargement size increases with the size of the structuring element.

Erosion L'érosion de X par B est l'ensemble des pixels p tels que la fenêtre B_p est incluse dans X :

1	1	1
1	**1**	1
1	1	1

Table 4: élément structurant 1 (centre en gras) / Structuring element 1 (center in bold).

0	1	0
1	**1**	1
0	1	0

Table 5: élément structurant 2 (centre en gras) / Structuring element 2 (center in bold).

$$Y = E_B(X) = \{p | B_p \subset X\}$$

En pratique, éroder une image binaire revient à placer l'élément structurant en chaque point (x, y) et affecter, au point (x, y) de l'image érodée, la valeur 1 si l'élément est "inclu" dans l'objet.

English:
In practice, the erosion of a binary consists in placing the structuring element (mask or filter) at each point (x, y) and to assign, to the (x, y) pixel of the eroded image, 1 if the structuring element is included in the "white" object. Erosion enables to reduce the size of white regions.

Ouverture et fermeture Sur la base des opérateurs élémentaires que sont l'érosion et la dilatation, on peut définir l'opérateur d'ouverture ("opening"):

$$O_B(X) = D_B(E_B(X))$$

et l'opérateur de fermeture ("closing"):

$$C_B(X) = E_B(D_B(X))$$

L'ouverture a notamment pour effet d'éliminer les structures de petite taille et/ou les excroissances. La fermeture permet quand à elle de "boucher les trous" et/ou de fermer les concavités.

English:
Opening consists in performing an erosion and then a dilation: this enables to remove small objects. Closing consists in dilating and then eroding an image : this allows to close concavities and fill holes of binary regions.

0	0	0	0	0	0	0
0	0	1	1	1	1	0
0	1	1	1	0	1	0
0	0	1	1	1	1	0
0	1	0	0	0	0	0
0	1	1	0	0	0	0
0	0	0	0	0	0	0

Table 6: Image binaire / Binary image.

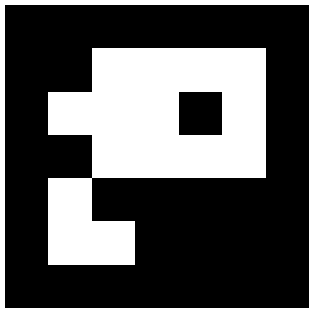

Table 7: Image binaire / Binary image.

2.1 Cas image synthétique - Synthetic image

- Calculer manuellement l'érosion ainsi que la dilatation de l'image binaire représentée dans les tableaux 6 et 7, pour chaque élément structurant (tableaux 4 et 5).
- Appliquez ces opérations à l'image binaire représentée dans le tableau 6 en utilisant les fonctions `scipy.ndimage.morphology.binary_erosion` et `scipy.ndimage.morphology.binary_dilation`. L'élément structurant (tableau `numpy` 2D composé de 0 et de 1) correspond au paramètre `structure`.
- Quelle opération (et quel élément structurant) permet d'obtenir l'image représentée dans les tableaux 8 et 9 ? Pour répondre, n'hésitez pas à tester les différentes opérations (érosion, dilatation, ouverture et fermeture).

English:

- Manually compute the erosion and the dilation of the binary image reported by Tables 6 and 7, for both structuring elements (Tables 4 and 5).
- Compute these operations using `scipy.ndimage.morphology.binary_erosion` and `scipy.ndimage.morphology.binary_dilation` functions. A structuring element (2D `numpy` array composed of 0s and 1s) is specified by parameter `structure`.

- Which operation (with which structuring element) enables to obtain the image related to Tables 8 and 9 ? To answer, test differents operators (erosion, dilation, opening and closing).

0	0	0	0	0	0	0
0	0	1	1	1	1	0
0	1	1	1	1	1	0
0	0	1	1	1	1	0
0	1	1	1	0	0	0
0	1	1	0	0	0	0
0	0	0	0	0	0	0

Table 8: Image traitée / Processed image.

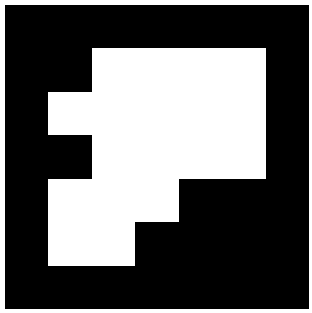

Table 9: Image traitée / Processed image.

2.2 Cas image réelle: réduction du bruit - Real image : noise removal

En reprenant l'image de l'assiette et en appliquant un seuillage (otsu) sans filtrage préalable, on obtient une segmentation imparfaite (voir figure 3). L'objectif est corriger cette segmentation (à partir de l'image binaire) en "bouchant" les trous noirs présents au niveau de l'assiette, et en éliminant les points blancs du fond.

Trouver la combinaison d'opérations, à base de morphologie mathématique, permettant d'obtenir l'assiette.

English:
By considering the image with the plate, and by thresholding it (Otsu threshold), on obtains an imperfect segmentation (see to Figure 3). The purpose is to correct this, by filling holes of the plate and by removing white pixels from the background. Find the set of operations (based on binary mathematical morphology) allowing to correctly extract the plate.

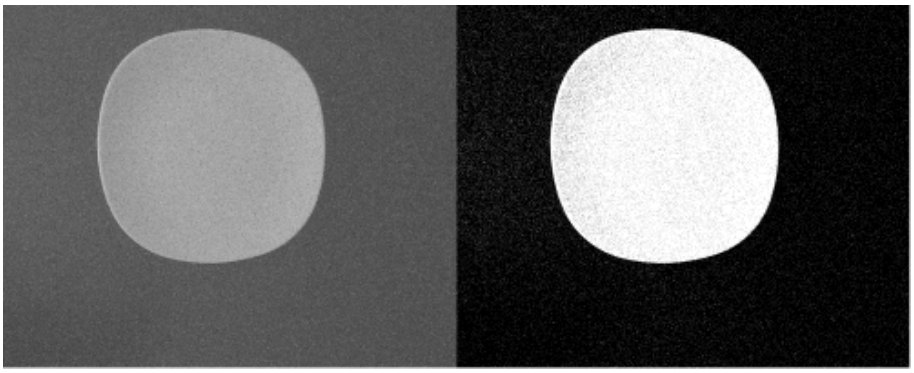

Figure 3: Image initiale et image seuillée / Initial image and thresholded one.

2.3 Cas image réelle: estimation taille cavités / Real image : estimation of hole size of washers.

Dans le cas de l'image avec les rondelles, on souhaite estimer la taille des cavités (1230 pixels et 360 pixels). Pour cela, nous allons utiliser la morphologie mathématique pour détecter les trous à partir de l'image segmentée, la procédure étant résumée par la figure 4:

1. Boucher les cavités à partir de l'image binaire, en utilisant l'opérateur adapté (érosion, dilation, ouverture ou fermeture). L'élément structurant devra être de grande taille. Une alternative (moins couteuse en temps de calcul) consiste à répéter une opération élémentaire plusieurs fois avec un petit élément structurant (paramètre `iterations` des opérateurs fournis par le package `scipy.ndimage.morphology`). La difficulté concerne le choix du nombre d'itérations. A noter que l'on peut également boucher des cavités avec la fonction `ndimage.morphology.binary_fill_holes`.

2. Soustraire de l'image "bouchée" l'image binaire non bouchée: vous devez obtenir les trous, dont vous pouvez finalement calculer la taille. Pour calculer la différence, faites attention à soustraire des images ayant les mêmes valeurs maximales et minimales !

English:
One wants to estimate the size (1230 pixels et 360 pixels) of holes of two

washers. For this, one proposes to use binary binary mathematical morphology to detect and recover holes from the segmented image (see Figure 4 for details) :

1. Fill holes from the binary thresholded image, using the appropriate operator (with a large structuring element). Note that an alternative (less time consuming) consists in repeating a given operation (with a small structuring element) several times (parameter `iterations`. Note that, holes can also be filled using the `ndimage.morphology.binary_fill_holes` function (also based an mathematical morphology).

2. Substract, from the filled image, the initial image : you should obtain holes, the size of which can finally be easily measured. Note : for the substraction, check that both images (filled - not filled) depict the same minimal/maximal values (e.g. both maximum at 1 or at 255 but not a mix !).

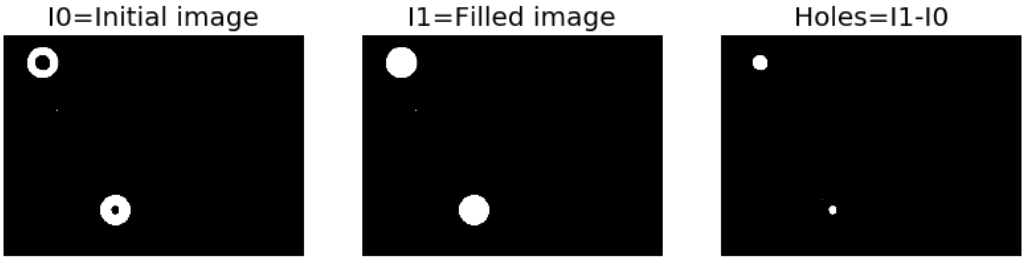

Figure 4: Image initiale et image seuillée / Initial image and thresholded image