

Dernière mise à jour	Informatique	Denis DEFAUCHY
01/06/2023	7 - Matrices de pixels et images	TD 7-15 – Détection de zones et motifs

# Informatique

## 7

# Matrices de pixels et images

***TD 7-15***



***Détection de zones et motifs***

Dernière mise à jour	Informatique	Denis DEFAUCHY
01/06/2023	7 - Matrices de pixels et images	TD 7-15 – Détection de zones et motifs

## Contexte

De manière générale, le travail que nous allons réaliser permet d'extraire n'importe quelle forme/zone ou motif d'une image à partir du moment où il/elle possède des caractéristiques analysables (couleurs, contours etc.). Vous trouverez plusieurs sujets sur mon site qui la mettent en œuvre pour différentes applications (extraction de galaxies dans une image de l'ESA, extraction de chiffres manuscrits pour de l'IA et la reconnaissance d'écriture, recadrage et correction automatique de QCM, différenciation de lignes blanches pour la détection de lignes dans des applications d'aide à la conduite ou de conduite autonome).

Dans le travail qui suit, nous allons mettre en place la recherche de formes et motifs :

Trouver le centre des disques noirs	Trouver le centre des motifs
	

On peut alors envisager par exemple le recadrage automatique (cf. TD16- Recadrage bilinéaire) de l'image à partir de l'identification automatique de ses 4 coins, ou l'utilisation d'une cible au sol pour guider un drone, etc.

## Affichage de l'image

Afin d'assurer un fonctionnement rapide sur tous les ordinateurs, je vous mets à disposition un dossier à télécharger **COMPLETEMENT**, soit le dossier contenant tous les fichiers, et non les fichiers pris séparément

Sans ouvrir le dossier, faite juste « Télécharger – Téléchargement direct » puis mettez ce dossier dans votre répertoire personnel.

**[LIEN](#)**

Dossier\_Partagé

Copier dans Dropbox

↓ Télécharger

Nom

7-16 - Recadrage bilinéaire

Si le téléchargement est sous forme de Rar, Zip... Pensez à dézipper l'archive afin d'avoir le dossier voulu !

Vous y trouverez un code élève et l'image « Image.bmp » ainsi que sa version numpy « Image.npy ».

**Question 1: Télécharger et exécuter le code fourni qui créera et affichera l'image « Image » sous Python**

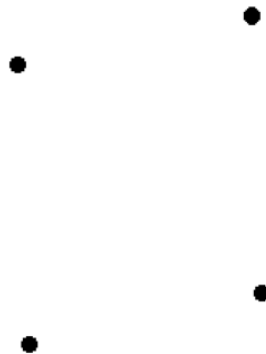
Dernière mise à jour	Informatique	Denis DEFAUCHY
01/06/2023	7 - Matrices de pixels et images	TD 7-15 – Détection de zones et motifs

## *Sélection des zones noires*

On souhaite extraire de l'image fournie les 4 disques noirs dans une image en noir et blanc. Dans l'image en couleurs, on suppose qu'un pixel est noir si  $R < 10$  &  $G < 10$  &  $B < 10$ .

**Question 2: Créer une fonction NB(im) qui renvoie une nouvelle image imnb (array) qui sera la transformation de l'image im en noir et blanc en sélectionnant le noir**

**Question 3: Afficher l'image en noir et blanc et vérifier le résultat obtenu**



Dans la suite, l'argument imnb lors de la définition des fonctions sera cette image.

## *Etude des zones de l'image*

L'image obtenue ci-dessus présente une grande « zone » blanche et un certain nombre de « zones » noires. Nous allons programmer une méthode permettant d'associer un numéro à chacune de ces zones afin de pouvoir les dénombrer et de les étudier.

Pour chaque pixel de l'image étudiée désigné par une liste [l,c] de sa ligne l et sa colonne c, on souhaite déterminer chacun de ses 4 voisins (gauche, droite, bas, haut) dans cet ordre sous la forme d'une liste de 4 listes [li,ci]. Un voisin qui sortirait de l'image sera défini comme égal au pixel [l,c].

**Question 4: Proposer une fonction Liste\_voisins(l,c,Nl,Nc) prenant en argument la ligne l et colonne c du pixel étudié, les nombres de lignes Nl et colonnes Nc de l'image, et renvoyant la liste de ses 4 voisins comme précisé ci-dessus**

```
>>> Liste_voisins(0,0,10,10)
[[0, 0], [0, 1], [1, 0], [0, 0]]

>>> Liste_voisins(1,1,10,10)
[[1, 0], [1, 2], [2, 1], [0, 1]]

>>> Liste_voisins(9,9,10,10)
[[9, 8], [9, 9], [9, 9], [8, 9]]
```

Dernière mise à jour	Informatique	Denis DEFAUCHY
01/06/2023	7 - Matrices de pixels et images	TD 7-15 – Détection de zones et motifs

On suppose avoir à disposition une table T sous forme d'array à deux dimensions, de mêmes dimensions que l'image étudiée, contenant des entiers codés sur 64 bits. Elle est initialisée à des valeurs de -1 partout. On souhaite que cette table contienne par la suite des entiers positifs croissants tels que toutes les cases contenant l'entier k représentent une « zone » numéro k de l'image noir et blanc.

Ainsi, comme le premier pixel en haut à gauche est blanc, la première zone de numéro k=0 sera l'intégralité du blanc de l'image ne formant qu'une zone. La zone 1 sera une première zone noire, la zone deux une seconde, etc.

Voici un exemple de table T associée à l'image de 8x8 pixels ci-contre en réalisant un parcours colonne par colonne (pour chaque colonne, puis pour chaque ligne) de la procédure que nous allons mettre en place :




0	0	0	0	0	4	0	6
0	2	2	0	0	4	0	0
0	2	2	0	0	4	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	5
1	0	3	0	1	0	5	5
1	0	0	0	1	0	5	5
1	1	1	1	1	0	0	0

Ce qui donnera, en affichant la table T avec notre fonction Affiche (lorsque ce sont des entiers, et non des triplets, on obtient un dégradé de couleurs de bleu à jaune automatiquement) :



Dernière mise à jour	Informatique	Denis DEFAUCHY
01/06/2023	7 - Matrices de pixels et images	TD 7-15 – Détection de zones et motifs

Dans la suite, on dira qu'une case  $T[l,c]$  a été **marquée** si sa valeur  $T[l,c]$  est différente de -1.

Par ailleurs, on dira par abus de langage « **pixel  $[l,c]$**  » pour désigner le pixel à la ligne  $l$  et la colonne  $c$  et « **case  $[l,c]$**  » la valeur de la case à la ligne  $l$  et la colonne  $c$  de la table  $T$ .

Enfin, comme nous travaillerons sur l'image en noir et blanc, on parlera de « **valeur** » 0 ou 255 pour identifier les valeurs  $R=G=B$  de tous les pixels.

On souhaite créer une procédure qui, à partir d'un pixel initial  $[l,c]$  quelconque de l'image associé à une case  $T[l,c]$  initialisée à un entier  $k$  dans  $T$ , explore tous ses voisins non encore marqués ayant la même valeur, afin de les marquer du même entier  $k$ , et procède ainsi pour tous les voisins des voisins (etc.), tant qu'il y en a. Cette procédure va donc permettre de remplir la table  $T$  avec le même entier  $k$  pour chaque zone de l'image.

Principe de la fonction d'exploration à partir du pixel initial  $[l,c]$  :

- Initialiser une pile (liste) avec le couple  $[l,c]$  du pixel initial
- Tant que la pile n'est pas vide :
  - o Dépiler un pixel  $[l,c]$
  - o Affecter l'entier  $k$  à la case  $[l,c]$
  - o Déterminer tous les voisins du pixel  $[l,c]$
  - o Pour chaque voisin non marqué de même valeur :
    - Ajouter ce voisin à la pile

*Attention : Tout n'est volontairement pas dit*

Exemples : L'appel de la fonction en

- $[0,0]$  va créer toute la zone de 0 de la table  $T$
- $[0,5]$  va créer toute la zone de 1 dans la table  $T$

**Question 5: Proposer une fonction `Explorer(imnb,l,c,T,k)` prenant en argument la ligne  $l$  et la colonne  $c$  du pixel initial et l'entier  $k$ , et réalisant la procédure attendue**

Il reste maintenant à appeler cette procédure sur l'intégralité des pixels non marqués de la table  $T$  sur l'image en noir et blanc à partir du pixel  $[0,0]$ .

Principe de la fonction de détermination des zones :

- Initialisation d'une table  $T$  d'entiers à -1 (`dtype='int64'`)
- Initialisation de l'entier  $k$  à 0
- Parcours de tous les pixels colonne par colonne
- Exploration du pixel s'il n'est pas marqué en lui attribuant l'entier  $k$  (création de la zone  $k$ )
- Incrémentement de l'entier  $k$

*Attention : Tout n'est volontairement pas dit*

**Question 6: Créer la fonction `Zones(imnb)` réalisant cette procédure et renvoyant la table  $T$  associée à `imnb`**

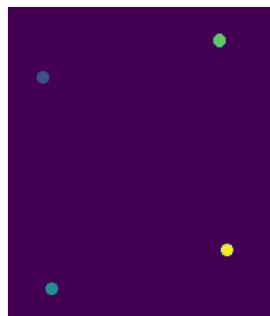
Dernière mise à jour	Informatique	Denis DEFAUCHY
01/06/2023	7 - Matrices de pixels et images	TD 7-15 – Détection de zones et motifs

**Question 7: Ecrire les lignes de code créant la table « Table » associée à l'image fournie, l'affichant, et créant et affichant dans la console le nombre de zones trouvées « Nb\_Zones »**

Remarque : On pourra utiliser `np.amax(A)` pour obtenir le maximum d'un array A.

Dans la suite, les arguments `n` et `T` lors de la définition des fonctions seront le nombre de zones « Nb\_Zones » et la table « Table ».

Vous devriez obtenir ceci :



Nombre de zones: 5

On souhaite mettre en place une fonction qui, en un seul parcours de la table `T`, renvoie une liste `LD` des données `D` de chaque zone `LD[k]=D=[Taille,MI,Mc,Col,l_min,c_min,l_max,c_max]` avec :

- Taille : nombre de pixels de la zone `k` (entier)
- `MI` : ligne moyenne de la zone `k` (flottant)
- `Mc` : colonne moyenne de la zone `k` (flottant)
- `Col` : Valeur `R` du triplet `RGB` de la zone `k` (0 : Noir, 255 : Blanc)
- `l_min,c_min,l_max,c_max` sont les limites de la zone `k` (incluses)

Remarque : le point `[MI,Mc]` est le centre de la zone `k` sur l'image. Pour rappel, on obtient le centre

$(L,C)$  d'un ensemble de `n` pixels  $(L_i,C_i)$  ainsi : 
$$\begin{pmatrix} M_l \\ M_c \end{pmatrix} = \begin{pmatrix} \frac{1}{n} \sum_{i=0}^{n-1} L_i \\ \frac{1}{n} \sum_{i=0}^{n-1} C_i \end{pmatrix}$$

Une fonction à compléter est donnée dans le code élèves.

**Question 8: Compléter la fonction `Donnees(T,n,imnb)` réalisant le travail attendu et créant la liste « `Donnees_Zones` »**

Vérifier :

```
>>> Donnees_Zones
[[136530, 183.99999999999926, 184.500000000000188, 255, 0, 0, 368, 369]]
```

Dans la suite, l'argument `LD` lors de la définition des fonctions sera cette liste « `Donnees_Zones` ».

Dernière mise à jour	Informatique	Denis DEFAUCHY
01/06/2023	7 - Matrices de pixels et images	TD 7-15 – Détection de zones et motifs

## ***Identification des disques***

On souhaite garder les 4 zones noires de l'image étudiée. On supposera que la table a bien mis en évidence une zone blanche et les 4 zones noires. On suppose que les points O1 et O2 ont tous deux une ligne supérieure aux points O3 et O4 et on triera la liste LC en une nouvelle liste LCsort=sorted(LC) afin d'obtenir un tri par ligne des 4 points.

**Question 9: Créer la fonction Selection(LD) renvoyant une liste LDs des données Di des 4 premières zones noires et créer la liste « Zones\_Sel » associée**

**Question 10: Créer la fonction Centres(LDs) renvoyant la liste des centres des zones sélectionnées (arrondis à l'entier) et créer la liste LC associée**

Vérifier :

```
>>> LC
[[58, 29], [28, 168], [225, 35], [194, 175]]
```

On souhaite que les points soient ordonnés ainsi : LC=[P1,P2,P3,P4] avec :

- P1 : point en haut à gauche
- P2 : point en haut à droite
- P3 : Point en bas à gauche
- P4 : point en bas à droite

**Question 11: Créer la fonction Identification(LC) créant et renvoyant la liste ordonnée des centres et le code remplaçant la liste LC par celle-ci**

## ***Affichage***

Nous allons afficher un pixel de couleur prédéfinie au centre de chacune des 4 zones trouvées par la procédure précédente afin de vérifier notre programmation. Voici la démarche :

- Définition d'une fonction Point(im,C,Col) prenant en arguments une image im, un point C (liste [l,c]) et une couleur Col (triplet d'entiers sous forme de liste [R,G,B]) et remplaçant le pixel im[l,c] par la couleur Col
- Définition d'une fonction Points(im,LP) récupérant les 4 points de LP=[P1,P2,P3,P4] et leur affectant respectivement la couleur rouge, vert, bleu et violet [255,0,255]
- Le code créant une copie de l'image « Image », appelant « Points » et affichant l'image avec ses pixels colorés




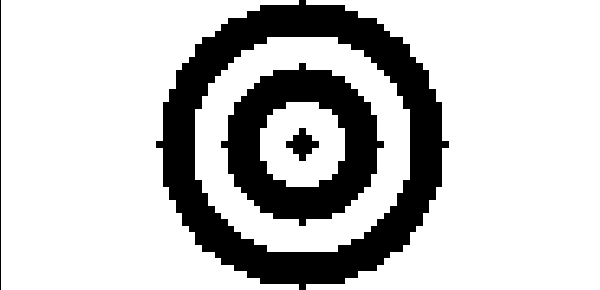
**Question 12: Créer les fonctions et le code attendus et vérifier que le résultat est correct**

Dernière mise à jour	Informatique	Denis DEFAUCHY
01/06/2023	7 - Matrices de pixels et images	TD 7-15 – Détection de zones et motifs

## Détection de motifs

La procédure mise en place précédemment est adaptée à la présence de zones noires uniques. Selon le problème traité, il faudra s'adapter. Ainsi, on peut par exemple envisager de créer des disques d'une couleur bien spécifique [R,G,B], d'extraire ces zones dans la fonction NB, de les trier par tailles décroissantes puis de ne garder que les 4 premières zones noires de l'image noir en blanc afin que de petites zones improbables de même couleur soient ignorées.

Dans la suite, on passe à la reconnaissance d'un motif, ce qui sera bien plus polyvalent et utilisable sur des images en noir et blanc, quel que soit leur contenu. Vous avez à disposition les deux images suivantes :

Image_Motif.bmp	Motif.bmp
	
	<a href="#">Lien vers le code pour la créer</a>

J'ai veillé à créer un motif axisymétrique afin de pouvoir l'utiliser dans des applications de recadrage d'images qui ne sont pas forcément scannées verticalement/horizontalement.

## Sélection des motifs

On rappelle que la distance euclidienne entre deux n-uplets  $u = (u_0, u_1, \dots, u_{n-1})$  et  $v = (v_0, v_1, \dots, v_{n-1})$  est le résultat du calcul suivant :

$$D = \sqrt{\sum_{i=0}^{n-1} (v_i - u_i)^2}$$

**Question 13:** Créer la fonction `Distance_uv(u,v)` calculant et renvoyant la distance euclidienne entre deux n-uplets (listes de n termes) u et v



Dernière mise à jour	Informatique	Denis DEFAUCHY
01/06/2023	7 - Matrices de pixels et images	TD 7-15 – Détection de zones et motifs

A partir d'une image sous forme d'arrays de pixels [R,G,B], on souhaite créer une liste de toutes les valeurs  $L\_RGB=[R,G,B,R,G,B,R,G,B,...]$  au format float de tous ses pixels ligne par ligne.

**Question 14: Créer la fonction `Analyse(Image)` renvoyant la liste `L_RGB` associée**

Vérifier :

```
>>> len(Analyse(Image))
160500
```

Il ne reste plus qu'à créer une fonction « `Selection_Motif` » analogue à la fonction « `Selection` » précédente, mais qui sélectionne parmi toutes les zones, les 4 zones de distance euclidienne la plus faible à l'image du motif en noir et blanc (image fournie). Attention : Le motif inclus dans l'image étudiée étant en noir et blancs, et parce que l'image étudiée peut provenir d'un scan/une photo qui transforme quelques pixels « en noir pas tout à fait noir » ([0,0,0] peut être [0,1,0] etc.), on transformera l'image étudiée en noir et blanc pour identifier les motifs (on utilisera la fonction NB précédente).

Pour pouvoir réaliser le calcul de distance, chaque zone et le motif devront être redimensionnés à la même taille, j'ai choisi de leur imposer une dimension 50x50. Pour cela, on utilisera les instructions suivantes :

```
from skimage.transform import resize
im = resize(im, (50,50), anti_aliasing=False, order=0)
```

Pour récupérer l'intégralité d'une zone (et tout le reste de l'image inclus dedans), on utilisera des slices :

```
im_loc = imnb[l_min:l_max+1, c_min:c_max+1]
```

Enfin, pour trouver les 4 zones les plus proches du motif, on pourra stocker dans une liste `L` des couples `[di,i]` des distances euclidiennes au motif de la zone d'indice `i`, trier la liste avec la méthode `L.sort()` et récupérer les indices des 4 zones sélectionnées (distances les plus faibles).

**Question 15: Créer la fonction `Selection_Motif(imnb,motif,LD)` prenant en argument l'image noir et blanc, le motif en noir et blanc (format array) et `LD`, et renvoyant la liste des données `Di` des 4 zones associées au motif recherché**

Dernière mise à jour	Informatique	Denis DEFAUCHY
01/06/2023	7 - Matrices de pixels et images	TD 7-15 – Détection de zones et motifs

## *Mise en place du code*

Il ne reste plus qu'à réaliser les étapes suivantes :

- Ouverture de l'image avec motif
- Transformation en noir et blanc avec sélection du noir
- Création de la table des zones
- Création des données des zones
- Ouverture du motif
- Sélection des 4 motifs
- Création des centres des motifs
- Identification des points  $P_i$
- Copie de l'image étudiée
- Ajout des points colorés aux centres des motifs
- Affichage du résultat

**Question 16:** Ecrire le code réalisant l'image attendue et vérifier le résultat obtenu

