

## OS09B-Image - TP 3 Morphologie mathématique

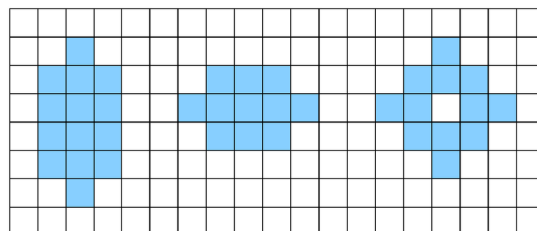
### Partie TD


#### Exercice 1 :


On applique à l'image illustrée ci-dessous des opérateurs de morphologie mathématique binaire avec un élément structurant de taille  $1 \times 3$  (1 ligne, 3 colonnes) :  $[1, 1, 1]$ .

Calculer les résultats des opérations suivantes :

- 1 érosion,
- 2 érosions consécutives,
- 1 ouverture.



 Pixel à 1 (objet)

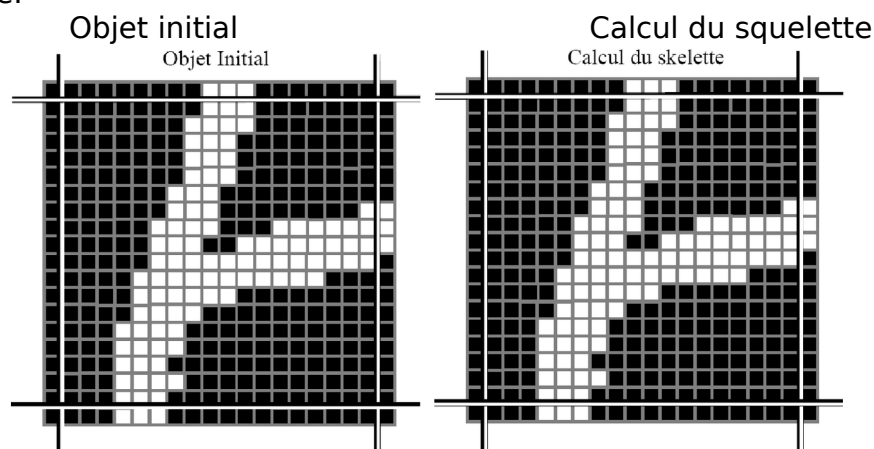
 Pixel à 0 (fond)

#### Exercice 2 :

On considère l'image ci-dessous. Le squelette est obtenu en amincissant successivement avec les éléments structurants suivants (pixel noir = 0, pixel blanc = 1, pixel gris = 1 ou 0) :



Effectuer la squelettisation de l'objet (pixels blancs) de la figure ci-dessous en appliquant successivement chacun des 8 masques et en itérant jusqu'à idempotence.



## Rappel définition Kernel:

```
# Rectangular Kernel
>>> cv2.getStructuringElement(cv2.MORPH_RECT,(5,5))
array([[1, 1, 1, 1, 1],
       [1, 1, 1, 1, 1],
       [1, 1, 1, 1, 1],
       [1, 1, 1, 1, 1],
       [1, 1, 1, 1, 1]], dtype=uint8)

# Elliptical Kernel
>>> cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(5,5))
array([[0, 0, 1, 0, 0],
       [1, 1, 1, 1, 1],
       [1, 1, 1, 1, 1],
       [1, 1, 1, 1, 1],
       [0, 0, 1, 0, 0]], dtype=uint8)

# Cross-shaped Kernel
>>> cv2.getStructuringElement(cv2.MORPH_CROSS,(5,5))
array([[0, 0, 1, 0, 0],
       [0, 0, 1, 0, 0],
       [1, 1, 1, 1, 1],
       [0, 0, 1, 0, 0],
       [0, 0, 1, 0, 0]], dtype=uint8)
```

### Exercice 1 :

Réaliser les transformations de l'exercice 1 de la partie TD en utilisant les fonctions python/opencv `cv2.erode()` et `cv2.dilate()`, en les appliquant sur une image de 64 \* 64 pixels avec en son centre un carré blanc (des pixels 16 à 48). Vous testerez avec un élément structurant de taille et forme différentes.

Erode avec des élément structurant de type: croix, carré. (dans une matrice 3x3)

Puis Dilate avec des élément structurant de type: croix, carré, cercle. (dans une matrice 9x9)

(3 types d'érosions et 3 types de dilatations demandées)

### Exercice 3 :

Créer une image binaire B de 48 lignes et 64 colonnes contenant en son milieu un rectangle blanc de 24 lignes et 32 colonnes sur fond noir. Calculer le squelette de ce rectangle en utilisant la fonction suivante:

```
def skeletonize(img):
    """ OpenCV function to return a skeletonized version of img, a Mat
    object"""

    # hat tip to http://felix.abecassis.me/2011/09/opencv-morphological-skeleton/

    img = img.copy() # don't clobber original
    skel = img.copy()

    skel[:, :] = 0
    kernel = cv2.getStructuringElement(cv2.MORPH_CROSS, (3,3))

    while True:
        eroded = cv2.morphologyEx(img, cv2.MORPH_ERODE, kernel)
        temp = cv2.morphologyEx(eroded, cv2.MORPH_DILATE, kernel)
        temp = cv2.subtract(img, temp)
        skel = cv2.bitwise_or(skel, temp)
        img[:, :] = eroded[:, :]
        if cv2.countNonZero(img) == 0:
```

```
break  
  
return skel
```

1) Commentez le code, définissez pour chaque fonction les arguments utilisés et renvoyés (nom, type, taille, description).

2) Modifiez ensuite l'image binaire en ajoutant des points blancs sur les bords du rectangle et en ajoutant des points noirs dans celui-ci. Autre solution, ajoutez un bruit 'Sel et poivre' sur l'image. Conclure sur la robustesse de la squelettisation en regard de la qualité de l'image à traiter. (Faire varier l'intensité du bruit poivre et sel et conclure)

#### Exercice 4 :

Ecrire les fonctions suivantes :

FctDilat : réalise la dilatation « manuelle » d'une image binaire avec un élément structurant 3\*3 (OU logique)

FctEros : réalise l'érosion « manuelle » d'une image binaire avec un élément structurant 3\*3 (ET logique)

FctDilatNG : réalise la dilatation « manuelle » d'une image NG avec un élément structurant 3\*3 (MAX)

FctErosNG : réalise l'érosion « manuelle » d'une image NG avec un élément structurant 3\*3 (MIN)

Ecrire le programme principal permettant de :

1. Binariser d'une image en vous basant sur les TP précédents.
2. Réaliser l'érosion (3 fois), la dilatation (3 fois), l'ouverture et la fermeture de l'image **binarisée**.
3. Réaliser l'érosion (1 fois), la dilatation (1 fois), l'ouverture et la fermeture de l'image **en niveaux de gris**.
4. Binariser l'image résultante du point 3 puis comparez avec les résultats du point 2.