Chapitre 3

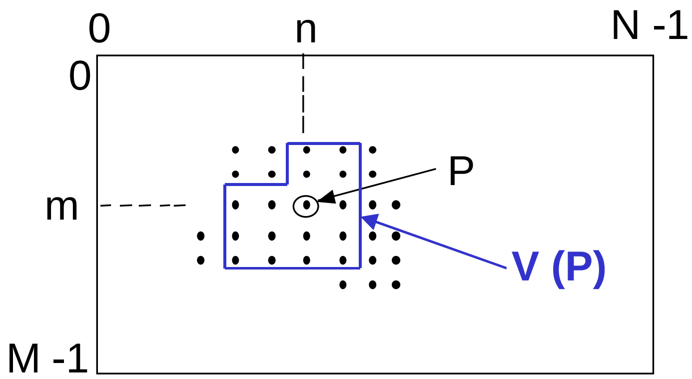
Fondamentaux du traitement d'image

Filtrage Linéaire

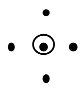
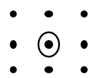
Voisinage d'un point

Notion de voisinage d’un pixel

- Point P d’affixe p = (m, n)
- Son voisinage : V(P) = {P’ connectés à P}



Domaine discret de l’image

Exemples: 4-N  8-N 

4-connexité 8-connexité

Le traitement d’image s’appuie fondamentalement sur des traitements à l’aide de voisinages. Cela signifie que les traitements effectués en un endroit donné correspondant à un pixel dépendent non seulement de ce pixel mais aussi de pixels appartenant à son voisinage.
Considérons un pixel P dont la position dans l’image est donnée par les coordonnées (m, n). Son affixe est donc p = (m, n). Un voisinage de P, noté V(P), se définit comme un ensemble de pixels P’ connectés à P. Le pixel P, cerclé dans la figure, appartient à son propre voisinage V(P).
Il faut donc, à ce niveau, définir la notion de « composantes connexes » au sein d’une image discrète, afin de savoir comment des pixels sont connectés entre eux.
Plutôt que de s’étendre sur la notion de connexité, nous présentons ici des figures illustrant les deux cas les plus couramment choisis :

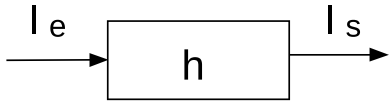
- un voisinage de « 4-connexité » : le pixel (centré et entouré dans la figure) n’a que quatre pixels voisins, chacun d’eux étant à distance unité de P avec la distance d_4 ;
- un voisinage de « 8-connexité » : le pixel (au centre et entouré) a huit pixels voisins, chacun d’eux étant également à distance unité de P, mais avec la distance d_8.

On définit les deux distances suivantes en voisinage numérique à structure d’échantillonnage carrée :

- $d_4(P, P') = | m – m' | + | n – n' |$;
- $d_8(P, P') = \mathrm{Sup}(| m – m' |, | n – n' |)$.

Opérations sur un voisinage

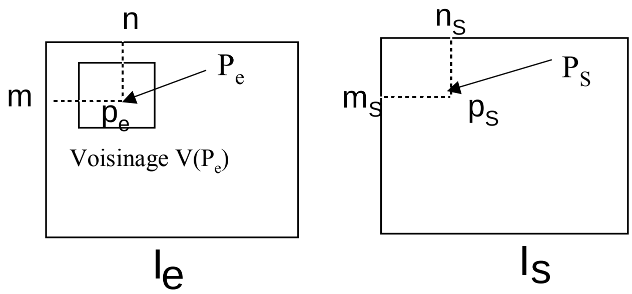
Filtrage linéaire (convolution)



Filtrage « transversal »: h est la réponse impulsionnelle 2-D appelée aussi « fonction d' étalement du point »

➢ Filtre à réponse impulsionnelle finie – RIF

$$I_s(m,n) = \sum_{(i,j)\in V(P)}\sum h(i,j)\, I_e(m-i\ ,n-j)$$



➢ Filtre récursif – IIR

$$Is(m,n) = \frac{1}{b_{0,0}} \times \left(\sum_{(i',j')\in V(P)}\sum a_{i',j'}.Ie(m-i',n-j') - \sum_{(i,j)\in V'(P')}\sum b_{i,j}.Is(m-i-1,n-j-1) \right)$$

Le principe est de construire à partir d'une première image I_e, une seconde image I_S généralement de même taille. Chaque pixel P_S d'affixe $p_S = (m_S, n_S)$ de l'image de sortie I_S est calculé à partir des pixels du voisinage $V(P_e)$ du point P_e d'affixe $p_e = (m, n)$ de l'image I_e. Usuellement $p_S = p_e$, i.e. $m = m_S$, et $n = n_S$.

Dans le cas le plus simple, la valeur du pixel P_S est obtenue par une combinaison linéaire des valeurs des pixels de $V(P_e)$, on parle d'opération de convolution (filtrage linéaire).

On donne ici l'exemple d'un filtrage transversal où la valeur Is(m, n) du pixel P_S est une combinaison linéaire des valeurs $I_e(m, n)$ des pixels du voisinage de P_e, pondérées par les coefficients h(i, j) de la réponse impulsionnelle du filtre :

$$I_S(m,n) = \sum_{(i,j)\in V(P_e)}\sum h(i,j).I_e(m-i,n-j)$$

La taille du voisinage est donc définie par la taille du support de la Réponse Impulsionnelle Finie (RIF) « h » du filtre.

Notons qu'il existe également des filtres récursifs. Le calcul des valeurs des pixels de I_S est alors plus complexe car la combinaison linéaire tient compte :

- du voisinage de I_e (m, n), le jeu de coefficients du filtre associé est l'ensemble $\{a_{i',j'}\}$;
- des valeurs antérieures de Is(m, n) précédemment calculées (récursivité), le jeu de coefficients du filtre associé est alors l'ensemble $\{b_{i,j}\}$.

$$I_S(m,n) = \frac{1}{b_{0,0}}.\sum_{(i',j')\in V(P_e)}\sum a_{i',j'}.I_e(m-i',n-j') - \sum_{(i,j)\in V(P_S)}\sum b_{i,j}.I_S(m-i-1,n-j-1)$$

Le résultat dépend donc de la méthode choisie pour parcourir l'image Ie (notion de causalité) car cela influe directement sur les valeurs antérieures I_S (m, n).

Exemple – Convolution – Corrélation

Exemple : filtre de support de taille (3×5) (3 en direction verticale et 5 en horizontale)

Noyau de convolution: $h = \begin{bmatrix} h_{-1,-2} & h_{-1,-1} & h_{-1,0} & h_{-1,1} & h_{-1,2} \\ h_{0,-2} & h_{0,-1} & h_{0,0} & h_{0,1} & h_{0,2} \\ h_{1,-2} & h_{1,-1} & h_{1,0} & h_{1,1} & h_{1,2} \end{bmatrix}$

Convolution

$$I_s(m,n) = \sum_{j=-2}^{+2} \sum_{i=-1}^{+1} h(i,j).I_e(m-i, n-j)$$

Corrélation Soit $h^*(i,j) = h(-i,-j)$

(=> symétrique de h par rapport à (0,0))

$$I_s(m,n) = \sum_{j=-2}^{+2} \sum_{i=-1}^{+1} h^*(i,j).I_e(m+i, n+j)$$

$$h^* = \begin{bmatrix} h_{1,2} & h_{1,1} & h_{1,0} & h_{1,-1} & h_{1,-2} \\ h_{0,2} & h_{0,1} & h_{0,0} & h_{0,-1} & h_{0,-2} \\ h_{-1,2} & h_{-1,1} & h_{-1,0} & h_{-1,-1} & h_{-1,-2} \end{bmatrix}$$

Dans le cas d'une convolution, le filtre est entièrement caractérisé par l'ensemble des coefficients { h(i,j) }, noté aussi { $h_{i,j}$ } pour rappeler que i et j sont des variables discrètes. On appelle cet ensemble le « noyau » du filtre.

De fait, ce noyau définit à la fois :

- le voisinage $V(P_e)$ à utiliser (dans l'exemple il s'agit d'un voisinage de taille (3×5) où la position (0,0) est à centrer sur P_e) ;
- les poids respectifs (les valeurs des h(i, j)) de chaque pixel de ce voisinage, afin de calculer la nouvelle valeur P_S.

La taille du support rectangulaire (I, J) et les poids étant connus, il est alors possible de calculer tous les pixels de l'image Is grâce à la relation :

$$I_S(m,n) = \sum_{i \in I} \sum_{j \in J} h(i,j).I_e(m-i, n-j)$$

<u>Remarque</u> : il faut prendre soin de définir des « conditions aux bords », c.a.d proposer une solution particulière afin de traiter les pixels aux bords de l'image. En effet, ces derniers n'ont pas de voisins à l'extérieur de I_e (une solution simple est de ne retenir que les pixels existants). Diverses méthodes de traitement des bords sont présentées dans l'exercice : « Filtrage Linéaire» de ce même chapitre.

A titre d'exemple, on considère un filtre « h », dont le noyau de convolution est :

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Ce filtre est en fait la somme d'un filtre Laplacien (détection de contours) et d'un filtre Identité, l'ensemble forme donc un filtre rehausseur de contours (*enhancement*).

On considère une image I_e de taille 3×3 :

$$\begin{bmatrix} 4 & 125 & 255 \\ 7 & 0 & 45 \\ 9 & 56 & 13 \end{bmatrix}$$

L'image I_e est filtrée par h. Voici le détail du calcul pour déterminer, après filtrage, la valeur du pixel central $I_S(1, 1)$:

$$I_S(1,1) = \sum_{i=-1}^{+1} \sum_{j=-1}^{+1} h(i,j) \,.\, I_e(1-i,1-j)$$

$$= \sum_{i=-1}^{+1} h(i,-1) \,.\, I_e(1-i,1-(-1)) + h(i,0) \,.\, I_e(1-i,1-0) + h(i,1) \,.\, I_e(1-i,1-1)$$

$$= \Big[h(-1,-1).I_e(1+1,2) + h(-1,0).I_e(1+1,1) + h(-1,1).I_e(1+1,0)$$

$$+ h(0,-1).I_e(1,2) + h(0,0).I_e(1,1) + h(0,1).I_e(1,0)$$

$$+ h(1,-1).I_e(1-1,2) + h(1,0).I_e(1-1,1) + h(1,1).I_e(1-1,0) \Big]$$

$$= (0\times13) + (-1\times56) + (0\times9) + (-1\times45) + (5\times0) + (-1\times7) + (0\times255) + (-1\times125) + (0\times4)$$

$$I_S(1,1) = -233$$

<u>Mise en garde</u> : la numérotation des lignes et des colonnes n'est pas la même pour l'image d'entrée et le filtre. Pour l'image, les indices vont de 0 à M-1 (lignes) et de 0 à N-1 (colonnes). Pour le filtre, l'élément $h_{0,0}$ est placé au centre du noyau de convolution, les indices vont donc de –I à +I (lignes) et de –J à J (colonnes).

Notons que, si l'on considère la fonction bidimensionnelle « h^* » symétrique de « h » par rapport à (0, 0), on peut également écrire la valeur du pixel de l'image de sortie I_S comme étant la corrélation entre I_e et h^* :

$$I_S(m,n) = \sum_{i \in I} \sum_{j \in J} h^*(i,j).I_e(m+i,n+j)$$

Exemples de masques de convolution typiques

$$\begin{bmatrix} \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix} \qquad \begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix} \qquad \begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix}$$

Masque 3×3 Masque 3×5 Masque 5×5

Masques de convolution plus larges :

(~ circulaire: 45 pixels ; rectangulaire: 11x11)

- Gain en continu « DC » du filtre : $\sum_i \sum_j h(i, j) = H(\nu_X = 0, \nu_Y = 0)$
- Filtre symétrique : h (-i,-j) = h (i,j) alors : convolution ≡ corrélation

=> H (ν_X, ν_Y) est une fonction de transfert réelle (pas de terme de phase)

On peut imaginer toutes sortes de masques de convolution. La taille de ce dernier définit la taille du voisinage. Différents supports pour le noyau sont présentés ici : 3×3, 3×5, et 5×5.
Comme dans le cas des signaux 1-D, on définit la réponse fréquentielle $H(\nu_X, \nu_Y)$ du filtre qui évolue en fonction des fréquences spatiales horizontales (ν_X) et verticales (ν_Y).
$H(\nu_X, \nu_Y)$ est la transformée de Fourier 2-D de la réponse impulsionnelle « h » :

$$\mathbf{H(\nu_X, \nu_Y) = \sum_n \sum_m h(m, n) \exp\left[-2j\pi(n\nu_X + m\nu_Y)\right]}$$

On peut montrer que le gain en continu (gain de la réponse fréquentielle H aux fréquences spatiales nulles : $\nu_X = 0$, $\nu_Y = 0$) du filtre se calcule comme la somme de tous les coefficients h(i, j) du noyau : $\text{Gain\_DC} = \sum_i \sum_j h(i,j)$.

L'exercice « FFT » présente plus en détail les particularités et les méthodes de représentation pour les spectres d'images et les réponses fréquentielles des filtres. La notion de fréquences spatiales y est approfondie.
Dans le cas particulier où le filtre est symétrique, $H(\nu_X, \nu_Y)$ est une réponse fréquentielle réelle.

<u>Remarque</u> : dans la suite du chapitre, par abus de langage, on parlera de fonction de transfert d'un filtre plutôt que de réponse fréquentielle.