

TP 4 : détection de contours

1. Le but du TP

Ce TP consiste à utiliser des méthodes de segmentation des images basées sur la détection des contours.

2. Détection des contours (*edge detection*)

L'analyse des contours consiste à segmenter l'image par détection de frontières. Cette détection s'apparente à une dérivation, aussi parle t-on de filtrage dérivatif. Le but est d'obtenir l'image gradient de l'image initiale. Le contour sur l'image gradient possède deux composantes :

$$M(x_0, y_0) = \sqrt{\frac{\partial I(x_0, y_0)^2}{\partial x} + \frac{\partial I(x_0, y_0)^2}{\partial y}} \quad \text{Module du contour}$$

$$D(x_0, y_0) = \arctg \left(\frac{\frac{\partial I(x_0, y_0)}{\partial x}}{\frac{\partial I(x_0, y_0)}{\partial y}} \right) \quad \text{Direction du contour}$$

Un gradient nul correspond donc à une zone homogène alors qu'un gradient non nul correspond à :

- Une ligne de contour si la bande est assez étroite
- Une région texturée ou dégradée si la région est étendue

Dans le cas d'un gradient horizontal gauche droite :

$$\frac{\partial I(x_0, y_0)}{\partial x} \rightarrow \frac{I(x_0 + 1, y_0) - I(x_0 - 1, y_0)}{2} \quad \text{Soit le masque discret de convolution suivant :}$$

-1	0	1
----	---	---

Dans le cas d'un gradient vertical haut bas :

$$\frac{\partial I(x_0, y_0)}{\partial y} \rightarrow \frac{I(x_0, y_0 + 1) - I(x_0, y_0 - 1)}{2} \quad \text{Soit le masque discret de convolution suivant :}$$

-1
0
1

3. Travail à réaliser

- Créer sous MATLAB un script vide nommé TP4.m
- Créer une image X de taille 256 × 256 représentant un carré gris clair sur un fond gris foncé. Le carré se situera vers le milieu de l'image.

```
X = ones(256,256)*80; % Cree une matrice de taille M x N ne
    contenant que des valeurs de 80
X(110:140,110:140)=200; % Cree un carré dont les valeurs: 200
imshow(X/255);
imwrite(X/255,'si.png'); %donne un nom pour l'image
```

Filtres gradients

$$h1 = [1 \quad -1]$$

$$h2 = [1 \quad -1]$$

- Appliquer le filtre de masque h1 en utilisant la fonction `imfilter` (consulter le `help`) et stocker l'image filtrée dans la matrice Y1. Afficher sur la même figure X, `mat2gray(abs(Y1))` et `mat2gray(Y1)`. Dans quelle direction ce filtre rehausse-t-il les contours ?
- Appliquer le filtre de masque h2 et stocker l'image filtrée dans la matrice Y2. Afficher sur la même figure X, `mat2gray(abs(Y2))` et `mat2gray(Y2)`. Dans quelle direction ce filtre réhausse-t-il les contours ?
- on considère les masques suivants :

$$h3 = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \quad h4 = \begin{bmatrix} 2 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -2 \end{bmatrix} \quad h5 = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \quad h6 = \begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{bmatrix}$$

- Appliquer les filtre de masque h3, h4, h5 et h6 et stocker les image filtrées dans des matrices. Afficher les résultats. Dans quelle direction ces filtres rehaussent-t-ils les contours ? ? Comparer les résultats ?

Filtres Laplacien

$$h7 = \frac{1}{8} \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

- Appliquer le filtre de masque h7 et stocker l'image filtrée dans la matrice Y7. Afficher le résultat sur la même figure que X. Dans quelle direction ce filtre réhausse-t-il les contours ?

Filtre LOG.

Le filtre de Marr et Hildreth, aussi appelé LOG pour Laplacien du Gaussien est très utilisé pour détecter les contours car il permet de résoudre certains problèmes rencontrés avec le filtre Laplacien. En effet, le filtre Laplacien réalise une double dérivation, qui est extrêmement sensible au bruit. L'idée est donc d'appliquer le Laplacien non pas directement sur l'image, mais sur une image déjà filtrée par un filtre Gaussien (donc potentiellement débruitée). Le filtre LOG dépend donc d'un paramètre qui est l'écart-type du filtre Gaussien appliqué en prétraitement. On choisit de façon à réduire l'effet nuisible dû au bruit.

```
h8 = fspecial('log',[5 5],1); % Filtre LOG de taille 5 x 5 et d'écart type 1
g8=abs(imfilter(double(X),h8));
```

- Appliquer un filtre LOG de taille 15×15 et d'écart-type $\delta = 1$ (dont on stockera le masque dans h8), afficher le résultat
- Faire varier la valeur de δ et comparer les résultats.
- Relancer tout votre script, en définissant cette fois-ci X comme une image réelle

Détection de contours

La détection des contours est réalisée par une segmentation qui va se baser non pas sur les valeurs des pixels, mais sur une détection de contours. Nous n'allons donc pas travailler sur l'image originelle, mais sur une image filtrée ayant des contours rehaussés. Nous allons définir un seuil T : tous les pixels de l'image rehaussée dont la valeur est supérieure à T seront mis à 1, tandis que tous les autres seront mis à 0. Les contours seront donc en blanc et le reste en noir. La fonction Matlab `edge` réalise cette tâche.

- Tester la fonction `edge` pour la détection de contours des différentes images. Consulter le `help` pour voir les paramètres de cette fonction