Thème Image - TP4 - Filtres différentiels, détection de contours (2ème partie)

5 - Mise en évidence du problème

L'opération de seuillage décrite dans la première partie du TP impose de choisir un seuil sur la norme du gradient, au-delà duquel un pixel donné est interprété comme un contour. Mais les pics de la norme du gradient n'ont pas tous la même hauteur, et il faut choisir un seuil suffisamment faible pour capter tous les contours significatifs. De ce fait, l'épaisseur des contours n'est pas constante et la méthode peut aboutir à des résultats qui ne sont pas satisfaisants. L'exercice suivant illustre ce problème.

Exercice 5 :

On considère l'image `im` obtenue de la manière suivante :

```
// image intialement noire
N = 40;
M = 2*N
im = zeros(N,M);

// l'image est composée de trois bandes de taille n:
n = floor(M/3)

// le premier tiers reste noir

// le deuxième tiers est un dégradé avec un fort gradient
// de 30 à 255
for j = n:2*n
    im(:, j) = 30 + ((j - n)/n)^4*225;
end

// le dernier tiers est à 255
im(:, 2*n+1:M) = 255;

// lissage gaussien
G = W_gauss(0.5);
im = conv2(im, G);
```

Vous pouvez visualiser cette image avec la fonction `afficher_image`. Vous pouvez également tracer une coupe horizontale avec la commande suivante :

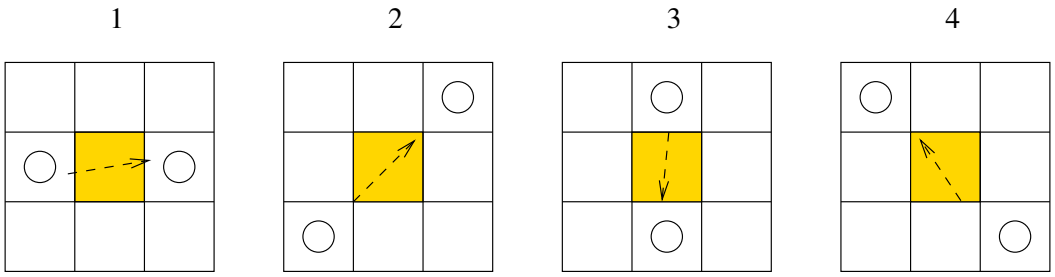
```
scf() ; plot(im(N/2, :))
```

Comme vous pouvez le constater, cette image contient deux variations importantes en les colonnes `j = M/3` et `j = 2M/3` qu'on aimerait pouvoir identifier comme des contours.

1. Essayez d'extraire les contours avec la fonction `contours_p`, en faisant varier le paramètre `p`. Que pensez-vous du résultat (sans se préoccuper des bords) ? Pouvez-vous trouver une valeur de `p` qui permette d'obtenir simultanément les contours en `M/3` et `2M/3` avec une épaisseur comparable ?
2. Proposez une explication. *Indication :* visualiser la norme du gradient.

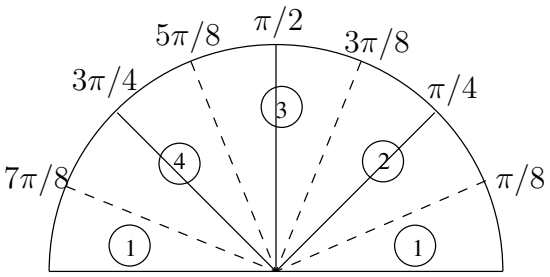
6 - Elimination des non-maxima locaux

Avec la méthode précédente, on identifie trop de points en tant que contours. Afin de pallier à ce problème on réexamine chaque pixel du contour et on essaie d'identifier s'il correspond à un maximum de la norme du gradient dans la direction donnée par le vecteur gradient (qui est la direction de plus grande variation locale). Ainsi, pour un pixel donné on compare sa valeur à celle de deux de ses voisins, choisis de manière à être les plus proches possibles de la direction du gradient. Il y a quatre cas possibles à choisir en fonction de la direction du gradient, comme illustré ci-dessous avec des exemples où le vecteur gradient est représenté par la flèche en pointillés.



Etant donnée une image `im`, on applique l'algorithme suivant :

- Soient `imx` et `imy` les tableaux contenant les dérivées partielles par rapport à x et y, `imn` le tableau contenant la norme du gradient. On rappelle que ces trois tableaux ont la même taille que l'image `im` mais ne sont pas des images en tant que tels ; ainsi il n'est pas nécessaire que leurs valeurs soient des entiers dans l'intervalle $[0, 255]$. Soit `imc` l'image des contours obtenue par seuillage de la norme du gradient (voir la première partie).
- Pour chaque pixel `(u,v)` qui n'est pas sur le bord de l'image et tel que `imc(u,v) = 255` :
  - Calcul de l'angle ϕ du vecteur gradient (`imx`(u, v), `imy`(u, v)) avec l'axe x, à l'aide de la fonction `atan` de `scilab` : `phi = atan(imy(u,v), imx(u,v))`.
  - Si $\phi \leq 0$, on remplace ϕ par $\phi + \pi$, ce qui ne modifie pas la direction et permet de se ramener à une valeur dans l'intervalle $[0, \pi]$.
  - On détermine les pixels voisins qui sont les plus proches de la direction du gradient parmi les quatre cas illustrés ci-dessus en fonction de l'angle ϕ :
    - Si $\phi \leq \pi/8$ ou $\phi \geq 7\pi/8$: cas 1.
    - Si $\pi/8 < \phi \leq 3\pi/8$: cas 2.
    - Si $3\pi/8 < \phi < 5\pi/8$: cas 3.
    - Si $5\pi/8 \leq \phi < 7\pi/8$: cas 4.



  - On compare la valeur de la norme du gradient dans le pixel central `(u,v)` à celles des deux voisins de l'étape précédente ; si une des deux valeurs est strictement supérieure, le pixel central n'est pas un maximum local et on considère qu'il ne fait plus partie du contour : `imc(u,v) = 0`.

Exercice 6 :

Identification des voisins les plus proches d'une direction donnée. Dans le fichier `contours.sci`, ajoutez une fonction `indices_voisins` avec la signature suivante :

```
function [u1,v1,u2,v2] = indices_voisins(u,v,phi)
```

Complétez-la pour qu'elle renvoie les indices `[u1,v1,u2,v2]` des pixels voisins du pixel `(u,v)` les plus proches de la direction donnée par l'angle `phi`. Cette fonction devra avoir le comportement suivant :

```
exec('contours.sci', -1);
u = 5; v = 5;
//// 1er cas ////
phi = %pi/16;
[u1,v1,u2,v2] = indices_voisins(u,v,phi);
// on attend (u1,v1) = (5,4) et (u2,v2) = (5,6)
//// 2eme cas ////
phi = %pi/4;
[u1,v1,u2,v2] = indices_voisins(u,v,phi);
// on attend (u1,v1) = (6,4) et (u2,v2) = (4,6)
//// 3eme cas ////
phi = %pi/2;
[u1,v1,u2,v2] = indices_voisins(u,v,phi);
// on attend (u1,v1) = (6,5) et (u2,v2) = (4,5)
//// 4eme cas ////
phi = 3*%pi/4;
[u1,v1,u2,v2] = indices_voisins(u,v,phi);
// on attend (u1,v1) = (6,6) et (u2,v2) = (4,4)
```

Exercice 7 :

Mise en oeuvre de l'algorithme.

— Dans le fichier `contours.sci`, ajoutez une fonction `contours_max` avec la signature suivante :
`function im_out = contours_max(im, p)`
Complétez-la pour qu'elle prenne en argument une image `im` et un pourcentage `p`, calcule l'image des contours par seuillage (comme à la séance précédente), applique l'algorithme détaillé plus haut pour éliminer les non maxima locaux, puis renvoie l'image des contours.

— Vous pouvez la tester avec l'image de l'exercice 5, vous devez obtenir comme contours deux traits verticaux d'épaisseur égale (en faisant abstraction des bords de l'image).

```
exec('contours.sci', -1);
p = 0.8;
// im est l'image de l'exercice 1
afficher_image(contours_max(im, p));
```

— Pour tester la bonne prise en compte des directions du gradient, vous pouvez réaliser le test suivant :

```
exec('contours.sci', -1);
p = 0.8;
// la fonction disque est fournie dans contours.sci
im = disque();
afficher_image(contours_max(im, p));
```

Vous devez obtenir un cercle.

7 - Bruit, lissage, et détection des contours

Exercice 8 :

Reprenez les images de l'exercice 4 (première partie du TP) et les valeurs de `sigma` et `p` que vous aviez choisies, et testez l'effet de l'élimination des non maxima locaux en comparant avec les résultats précédents.