

Gabor par Didier STRAUS & Julien GONY

1) Informations générales :

La formule utilisée pour le filtre de Gabor est celle-ci :

$$G(fx0,fy0,X,Y)=1/(2*\pi*\sigma^2)*\exp(-(X^2+Y^2)/(2*\sigma^2))*\exp(j*2*\pi*(fx0*X+fy0*Y)).$$

(Les Gaussiennes sont circulaires).

Cette formule n'est pas appliquée directement, on sépare la partie réelle, et la partie imaginaire.

Le banc de filtres que nous avons utilisé est constitué de 4 directions (Vertical, Horizontal, deux diagonales), et de 2 échelles de fréquences.

Les variables sont demandées à l'utilisateur :

-SIGMA : le diamètre du filtre (compris entre 2 et 11) influe sur le temps de calcul car la matrice représentant le filtre est de $3*\sigma$.

-Un coefficient k : qui multipliera les fréquences par défaut $fx0$, $fy0$. Ce coefficient est compris entre 1 et 10.

Les fréquences par défaut sont :

0.2;0.0 et 0.4;0.0 pour la direction horizontale

0.15;0.15 et 0.3;0.3 pour la première diagonale

0.0;0.2 et 0.0;0.4 pour la verticale

-0.15;0.15 et -0.3;0.3 pour la deuxième diagonale

2) Mode d'emploi :

Compilation :

Dans le terminal, placez vous dans le répertoire où se trouvent les sources du programme, puis tapez "make" (sans les guillemets), le terminal compile les sources.

Il est possible qu'un warning s'affiche :

acces_lena.c: In function 'lit_lena':

acces_lena.c:219: warning: comparison is always true due to limited range of data type

cc -o progabor gabor.o acces_lena.o -lm

Si c'est le cas, n'en tenez pas compte, le programme fonctionnera.

Utilisation :

Pour exécuter le programme vous devez vous trouver dans son répertoire (dans le terminal), ensuite tapez "./progabor" (sans les guillemets).

Le programme vous invite à saisir le nom de votre image (cette image doit être au format LENA pour pouvoir être lue) sans l'extension .lena donc pour l'image exemple.lena, vous devrez juste indiquer : exemple

Ensuite le programme vous invite à saisir la valeur de Sigma (un entier compris entre 2 et 11), par défaut utilisez 2.

Ensuite le programme vous invite à choisir un coefficient (compris entre 1 et 10) pour les fréquences fx_0 et fy_0 , par défaut utilisez 1. Les fréquences prédéfinies dans le programme (dans la variable : `float freq[8][2]`) seront multipliées par ce coefficient.

Le programme applique donc les filtres sur l'image fournie, pour chaque filtre, il affiche le nom de l'image correspondante quand l'application du filtre est terminée. (Le programme applique au total 8 filtres).

Les noms des images filtrées sont du même type que le nom original avec l'ajout d'un chiffre :

exemple.lena deviendra : exemple1.lena pour le filtre 1, exemple2.lena pour le filtre2,...

3) Le programme :

Pour la lecture/écriture des images LENA, le programme utilise les fichiers sources : `acces_lena.c` et `structlena.h`

Le programme `gabor.c` : dans le main, on demande à l'utilisateur de donner le nom de l'image, la valeur de sigma, et le coefficient multiplicateur des fréquences (fx_0 et fy_0).

Ensuite, le programme exécute une boucle for (8 fois) pour appliquer les 8 filtres:
_ on calcul le nom de l'image filtrée;

_ on initialise l'image filtrée en noire (0);

_ on exécute la procédure `calculMat` qui va calculer le filtre (la matrice). Pour cela la procédure utilise la fonction `gabor` qui pour un pixel donné et des fréquences fx_0 et fy_0 données retourne un complexe (complex), la fonction `gabor` applique la formule de la feuille. Nous obtenons ainsi la partie gauche bas du filtre, pour déterminer les autres parties, on applique les symétries;

_ on applique le filtre sur l'image(on convolue la matrice avec l'image), pour tous les pixels (sauf ceux qui se trouvent aux bords de l'image) de l'image on applique le filtre sur le pixel et ces voisins (le nombre de voisins est dépendant de sigma) : pour chaque pixel on le multiplie avec son équivalent dans la matrice du filtre, et l'on additionne les résultats pour obtenir la valeur du pixel filtré;

De retour dans le main, nous avons filtré l'image, maintenant nous allons effectuer un recadrage dynamique sur l'image obtenue (de manière à l'éclaircir), nous avons besoin pour cela des valeurs minimum et maximum des pixels, pour la valeur minimum, c'est 0 puisque les bords sont toujours en noire (0). on recherche le maximum en parcourant toute l'image.

Ensuite pour recadrer l'image il suffit de multiplier chaque pixel par $255/(max-min)$ c'est-à-dire : $255/max$.

Enfin, on sauvegarde cette image.

Listing des fichiers sources :

```
structlena.h  
acces_lena.c  
gabor.c  
makefile
```

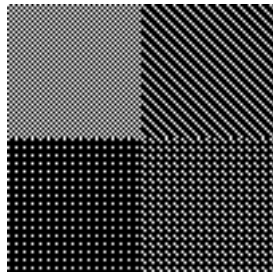


Image originale :

Après exécution du programme quelques résultats que l'on peut obtenir :

