Nom: Ivanina IVANOVA, Preshtibye RAGGOO MIN INT

Compte-rendu TP3 MAP201
Introduction au Itrage

Exercice 1

1. Avec scilab on peut construire le sous-tableau de s contenant les 2m+1 pour u qui commence à m+1 et finit à M-m.

```
function afficher_fenetre(s, m)
   // taille du signal
   M = size(s,2);
   for i =m+1:M-m
      disp(s(i-m:i+m))
   end

endfunction
```

2. On peut constater que la fonction marche bien avec les tests proposés.

Exercice 2

1.

```
function s2=moyenne(s1, m)
   // taille du signal
   M = size(s1,2);
   // on initialise un vecteur de zeros
   // de meme taille que s1
   s2 = zeros(s1);
   for i =m+1:M-m
      s2(i)=mean
   end
   //disp(s2)
endfunction
```

3. Les résultats sont satisfaisants

Exercice 3

```
function s2=moyenne_W(s1, W)
   // taille du signal
   M = size(s1,2);
   // m est tel que la taille du filtre est
   // 2m+1
   m = (size(W,2)-1)/2;
   // on initialise un vecteur de zeros
   // de meme taille que s1
```

```
    s2 = zeros(s1);
    for i =m+1:M-m
        s2(i)=sum(W.*(s1(i-m:i+m)))
    end
    //disp(s2)
endfunction
```

Exercice 4

```
function G=W_gauss(sigma)
    // seuil pour definir L
    eps = 10^(-3);
    ////// A COMPLETER //////
    // Le tableau G en sorti doit etre
    // horizontal
    deff("y=f(x)","y=exp(-(x.^2)./(2.*(sigma.^2)))");
    L = floor(sigma*sqrt(-2*log(eps)))

    G=-L:L
    M=size(G,1)
    G=f(G);
    G=G/sum(G);
    //G(0)=f(0)

endfunction
```

Sigma	0.5	1	1.5	2
taille	3	7	11	15
Valeurs	0.106507 0.786986 0.106507	0.004433 0.0540056 0.2420362 0.3990503 0.2420362 0.0540056 0.004433	0.0010284 0.0075988 0.0360008 0.1093607 0.2130055 0.2660117 0.2130055 0.1093607 0.0360008 0.0075988 0.0010284	0.0004364 0.0022163 0.0087655 0.0269996 0.0647686 0.1210037 0.1760593 0.1995013 0.1760593 0.1210037 0.0647686 0.0269996 0.0087655

				0.0022163 0.0004364

Exercice 5

1. En augmentant la taille de la fenêtre avec la fonction moyenne on voit que la qualité de sortie de l'image est bien plus lisse.
2. On observe la même chose pour la qualité de l'image avec le filtre Gaussien.
3. Cependant on fait la remarque qu'on ait un affichage bien plus lisse avec le filtre Gaussien en comparaison avec le filtre moyen.
4. Cependant, l'efficacité des deux filtres est remise en question par l'observation qu'on vient de faire, ceci en observant les valeurs du tableau ci-dessus et l'affichage des images avec les 2 filtres. Plus les valeurs de m et sigma augmentent, on remarque que les deux filtres nous donne une sortie d'image trop lisse, on perd ainsi beaucoup d'information dans ce cas.

Exercice 6

1. On construit avec scilab le sous-tableau avec : *im(u-m:u+m, v-m:v+m)*
2. u doit être entre m+1 et M-m, mais v doit être entre m+1 et N-m
3. La fonction affiche les mêmes résultats.

```
function afficher_fenetre2D(im, m)
   // taille de l'image
   [M,N] = size(im);
   for u = m+1:M-m
      for v = m+1:N-m
         disp(im(u-m:u+m,v-m:v+m));
      end
   end
endfunction
```

Exercice 7

```
function im2=moyenne2D(im1, m)
   // taille de l'image
   [M,N] = size(im1);
```

```
    // tableau de zeros
    // de la meme taille que im1
    im2 = zeros(im1);
    for u = m+1 : N-m
        for v = m+1 :N-m
            im2(u,v) = floor((mean(im1(u-m: u+m, v-m:v+m))));
        end
    end
endfunction
```

On exécute la fonction pour l'image d'un carré noir sur fond blanc ainsi qu'avec l'image papillon.bmp, comme indiqué dans l'énoncé. On peut faire la même observation pour les deux cas 2) et 3) : l'affichage de l'image se détériore, elle devient de plus en plus flou. On conclut que plus on augmente la taille de la fenêtre, plus la qualité de l'image diminue. L'image devient de plus en plus floue. Quand m = 5 , on peut pratiquement voir les pixels de l'image, ce qui nous indique une dégradation conséquente de l'image.

Exercice 8

```
function im2=moyenne2D_W(im1, W)
   // taille de l'image
    [M,N] = size(im1);
    // m est tel que la taille du filtre
    // dans une dimension est 2m+1
    m = (size(W,1)-1)/2;
    im2 = zeros(im1);
    for u = m+1:M-m
        for v = m+1:N-m
            im2(u,v)=floor((mean(im1(u-m:u+m,v-m:v+m))));
        end
    end
endfunction
```

On a ensuite exécuté la fonction qu'on a définie ci-dessus. On observe évidemment bien ce qu'on attendait, c'est-à-dire, on a les mêmes résultats qu'avec l'exercice précédent.