TD 2 UE Vision

Dans ce TD, nous aborderons les histogrammes des images et différents traitements qui en découlent.
Dans un premier temps, vous utiliserez la fonction d'Octave pour les histogrammes, puis vous la coderez.

---

1) Histogramme des images

Chargez une image en niveaux de gris « im1 », la fonction d'Octave pour afficher l'histogramme est:
>>figure, imhist(im1) ;

2) Histogramme d'une image couleur

L'image « bureau256_col.png » est une image grise mais codée en couleurs, chargez la dans la variable I.
Vérifier dans l'espace de travail qu'il s'agit de 3 plans couleurs.

Extraire les 3 plans (rouge, vert, bleu) de la manière suivante :
>>R = I(:,:,1); G = I(:,:,2); B = I(:,:,3);

Visualiser et comparer le tout en tapant :
```
>>figure,
>>subplot(1,4,1); imshow(I);title('Couleur');
>>subplot(1,4,2); imshow(R);title('R');
>>subplot(1,4,3); imshow(G);title('G');
>>subplot(1,4,4); imshow(B);title('B');
```

« Igrey » représente l'image I en niveaux de gris (cf. « rgb2gray » dans le TD 1).
Maintenant, quelle conclusion faites-vous à propos des histogrammes suivants :
```
>>figure,
>>subplot(1,4,1); imhist(Igrey);title('Image NG');
>>subplot(1,4,2); imhist(R);title('R');
>>subplot(1,4,3); imhist(G);title('G');
>>subplot(1,4,4); imhist(B);title('B');
```

Refaire la même démarche avec l'image « baboon.bmp », que constatez-vous à propos des différents plans et histogrammes ?

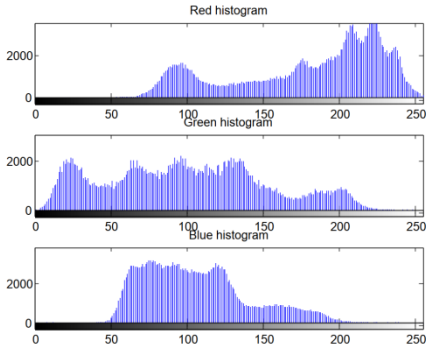
3) White balance

L'Assomption du monde gris (*Gray World*) est une méthode de balance des blancs qui suppose que la scène est en moyenne un gris neutre. L'hypothèse du monde gris est valable si nous avons une bonne répartition des couleurs dans la scène. Ainsi, la couleur moyenne réfléchie est supposée être la couleur de la lumière. Par conséquent, nous pouvons estimer la couleur de la couleur d'éclairage en regardant la couleur moyenne et en la comparant au gris. Voici l'algorithme Gray *World* qui permet une balance des blancs automatique.
On considère une image en couleurs *Ic*. Tout d'abord, calculer sa moyenne : Avg = mean(mean2(Ic)).
Ensuite, pour chaque canal rouge vert et bleu (R, G, B), calculer les 3 entités :

$$K_r = \frac{Avg}{mean(R)},\ K_G = \frac{Avg}{mean(G)},\ K_B = \frac{Avg}{mean(B)}.$$

Les 3 canaux de l'image de sortie sont calculés par :$R_{out} = K_r \cdot R,\quad G_{out} = K_G \cdot G,\quad B_{out} = K_B \cdot B$.
Enfin, reconstruire une image en couleur I_{out} à partir des trois canaux $(R_{out},\ G_{out},\ B_{out})$.

Tester sur l'image « clown.bmp » et comparer les histogrammes des 3 canaux avant et après l'application de l'algorithme de Gray *World*. Utiliser « *subplot* » pour une meilleure visualisation (exemple ci-dessous).



4) Fabriquer un histogramme

Chargez l'image « bureau256.png », puis créez la fonction *f_histo* qui construit l'histogramme d'une image codée en 8 bits. Affichez l'histogramme ainsi :

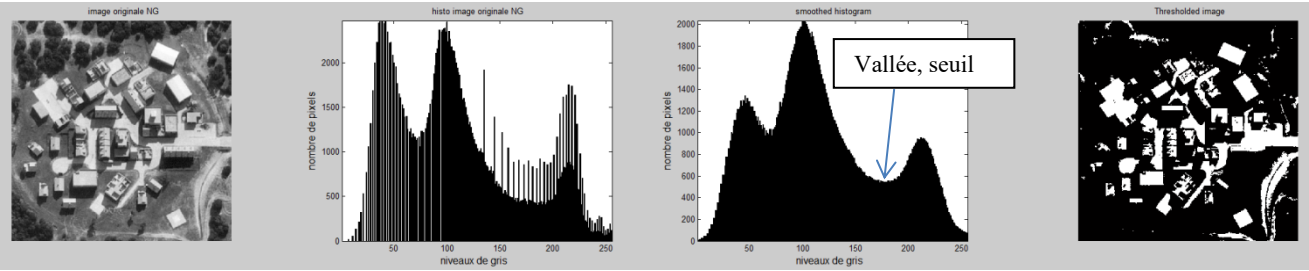
```
>> histo = f_histo(I);
>> figure, bar(histo), xlim([1 256]),
>> ylim([0 max(histo)])
>> title(' histo image originale NG')
>> xlabel('niveaux de gris  ','FontSize',12)
>> ylabel('nombre de pixels','FontSize',12)
```

5) Seuillage à la vallée

Créer une fonction *th_vallee* qui calcule le seuil minimal dans une vallée d'un histogramme lissé, comme illustré ci-dessous. Voici les différentes étapes.

a) Chargez l'image « oizo.tif ».
b) Calculer l'histogramme, appelé *H* (cf. question précédente).
c) Lissez *H* tel que :
```
>> s = 19 ;   m = ones(1, s)/s ;
>> H2 = conv2(H, m, 'same');
```
d) Calculez la valeur du seuil à la vallée puis seuillez l'image originale à cette valeur.
e) Que se passe-t-il lorsque `>> s = 15 ;` ?

Faire de même sur l'image « roue.tif ». Enfin, essayez sur l'image « bureau.png », en déduire les limites, même en changeant les valeurs de *m* pour le lissage de l'histogramme.



6) *Histogram equalisation*

Pour calculer l'égalisation d'histogramme, il faut commencer par calculer l'histogramme cumulatif de l'image. Tout d'abord, chargez l'image « paulina.png » appelée *I*, composée de *N* pixels.

a) Calculer l'histogramme, appelé *H* (cf., question précédente).
b) L'histogramme cumulatif se calcule par : $C(k) = \sum_{i=1}^{k} H(i)$, avec $k \in [0, 255]$.
c) Les nouvelles intensités sont calculées par : $J(i,j) = \frac{C(I(i,j))}{N} \cdot 255$, où *J* représente l'image égalisée.

Pour les images en couleurs, il faut rajouter une étape. Calculer l'image moyenne des 3 canaux : $I = \frac{R+B+G}{3}$.
Refaire les étapes a) et b) précédentes sur *I*. Appliquer l'égalisation d'histogramme sur chaque canal puis reconstruire l'image en couleurs à partir des canaux ainsi obtenus.
Enfin, appliquez l'égalisation d'histogramme sur les images « fox.jpg » et « montagne.jpg », comparer les résultats avec l'algorithme de *Gray World* (cf. question **3)**). Essayez avec vos images personnelles.

7) Comparaison d'images et d'objets avec les histogrammes et la distance de Bhattacharyya

Chargez les images «bureau1.png », «bureau2.bmp » et «cameraman.tif » appelée *I, J* et *K* respectivement.

a) Ecrire une fonction nommée « *[S] = f_bhattacharyya(H1, H2)* » calculant la distance entre deux histogrammes *H1* et *H2* normalisés (vérifier que la taille des histogrammes est la même dans la fonction pour calculer les scores). Voici sa formule : $B(H_1, H_2) = -\ln\left(\sum_{i=1}^{n} \sqrt{H_1(i) \cdot H_2(i)}\right)$, où n représente la longueur de l'histogramme.
b) Comparaison d'images : Comparez les histogrammes de *I, J* et *K* avec cette fonction.
c) Détection d'objets : Extraire deux sous images de *I* et de *J* nommées *Ic* et *Jc* telles que :
- Ic : coordonnées du premier point (69, 88), coordonnée du dernier point : (240, 199).
- Jc : coordonnées du premier point (76, 152), coordonnée du dernier point : (220, 246).

Comparer *Ic* et *Jc* avec la fonction «*f_bhattacharryya*» puis comparez les scores avec d'autres sous-images.