Traitement d'image, bases.

1• Objectif de cette première partie.

Découvrir quelques méthodes de base du traitement d'images. Pour en savoir plus sur telle ou telle fonction de traitement d'images, vous pouvez toujours vous aider du *help* de matlab. Nous vous conseillons d'ailleurs de regarder la documentation offerte par le *help* pour toutes les fonctions que vous allez utiliser.
Matlab est un langage qui utilise les données sous forme matricielle. Ce format est très compatible avec le format de représentation des images.

2• Chargement, affichage, ...

Liste des instructions disponibles sous matlab :
- imread : chargement d'une image, syntaxe im1=imread('nom_de_fichier.ext');
- image(im1) : affiche l'image im1 dans une fenêtre du type *figure*
- colormap(gray(255)) : spécifie pour l'affichage d'une image, la palette de couleur
- axis('image') : échelle carrée des pixels
- ginput(1) permet de saisir les coordonnées d'un point dans l'image avec la souris, syntaxe : [x,y]=ginput(1)
- double : transforme des données uint8 en double
- max(data) recherche la valeur maximum dans data
- min(data) recherche la valeur minimum dans data
- figure(n) fait apparaître la figure n

2.1• Chargement d'image.

Nous allons travailler sur une image présentant une vue de divers chocolats sur fond blanc. Pour charger cette image, utilisez la fonction `imread` :

```
ImageChocolat = imread('Chocolat.jpg') ;
```

L'objet `ImageChocolat` est une matrice Nlin x Ncol. Vous pouvez obtenir ses dimensions en utilisant la fonction `size` :

```
[Nlin, Ncol] = size(ImageChocolat) ;
```

Le < ; > permet d'éviter l'affichage, alors que la < , > permet l'affichage des valeurs.
Pour obtenir la valeur de niveau de gris d'un pixel, il suffit de l'adresser. Par exemple, si on veut connaître la valeur de niveau de gris du pixel central de l'image il suffit d'écrire :

```
ImageChocolat(floor(Nlin/2), floor(Ncol/2)) , % virgule pour l'affichage
```

La fonction `floor` assure que la coordonnée est bien une valeur entière.

2.2• Affichage de l'image.

Pour afficher l'image dans une figure, il faut, tout d'abord, créer la figure, puis demander au système d'afficher l'image dans la figure :

```
figure(12) ; image(ImageChocolat) ;
```

Votre image ne s'affiche pas avec les bonnes couleurs. Pour obtenir un affichage correct, il faut changer la table des couleurs. Vous pouvez le faire avec la commande colormap :

```
colormap(gray(256)) ;
```

Enfin, si vous tentez de déformer l'image en manipulant la fenêtre la figure, vous ne maîtrisez pas la dimension de l'image. Pour que l'affichage respecte les dimension de l'image il faut lui préciser le rapport hauteur largeur par :

```
axis('image') ;
```

2.3• Ecriture de script.

Vous pouvez résumer la totalité de ces instruction dans un ficher script. Pour ce faire, placez-vous dans un répertoire de travail vous appartenant (menu déroulant dans le bandeau du haut de la fenêtre interface), puis tapez la commande :

```
edit MonProgramme.m
```

ce qui provoquera l'appel de l'editeur de matlab. Vous pouvez écrire dedans toutes les commandes que nous avons utilisé :

```
clear all ; close all ; % ces commandes permettent de fermer
% toutes les fenêtres et de liberer toutes les variables
ImageChocolat = imread('Chocolat.jpg') ;
[Nlin, Ncol] = size(ImageChocolat) ;
figure(12) ; image(ImageChocolat) ;
colormap(gray(256)) ;
axis('image') ;
```

Vous sauvez votre script et vous pouvez le lancer en mettant son nom dans la fenêtre de commande :

```
MonProgramme
```

2.4• Transformation du format.

Matlab calcule sur 32 bits tandis que les données images sont codées sur 8 bits. Pour passer de 8 bits a 32 bits il faut taper la commande :

```
ImageChocolat = double(ImageChocolat) ;
```

2.5• Inversion d'image.

Les chocolats apparaissent en noir sur fond blanc, si on veut inverser l'image il suffit de taper :

```
ImageChocolat = 255 - ImageChocolat ;
```

vérifiez cela en affichant l'image.

2.6• Selection d'une sous-image.

Une sous-image pour matlab n'est autre qu'une sous-matrice. Pour créer une sous image-dont les dimensions sont delta_lin et delta_col et dont le coin en haut à droite a pour coordonnées (lin, col) il suffit de saisir :

```
Detail = ImageChocolat( lin : lin+delta_lin, col : col+delta_col ) ;
```

vérifiez cela en affichant l'image. Vous pouvez sélectionner le point en haut à droite par :

```
[lin, col] = round( ginput(1) ) ;
```

Vous pouvez aussi utiliser la fonction `imcrop` (regardez comment vous en servir).

3• HISTOGRAMME D'IMAGES.

3.1• Transformation d'un tableau en un vecteur.

Pour pouvoir effectuer un histogramme, il faut transformer votre matrice image en un vecteur contenant toutes le valeurs de niveau de gris. Pour faire cette opération, créez un nouveau script Histogramme.m et saisissez :

```
clear all ; close all ;
ImageChocolat = imread('Chocolat.jpg') ;
ImageChocolat = double(ImageChocolat) ;
[Nlin, Ncol] = size(ImageChocolat) ;
% creation d'un vecteur de 0 de Nlin*Ncol elements
Vecteur = ImageChocolat(:) ;
```

3.2• Histogramme.

La fonction hist(Vecteur,N) dessine l'histogramme de Vecteur sur N niveaux de gris. Déterminez un seuil global possible pour isoler les chocolats du fond blanc. Vous pouvez récuperer l'histogramme calculé en écrivant :

```
NiveauxDeGris = 0:255 ;
MonHistogramme = hist(Vecteur,NiveauxDeGris) ;
```

Réalisez un étirement puis une égalisation d'histogramme pour l'image des chocolats (méthode vue en cours). Vous essayerez aussi ces méthodes sur les images faiblement contrastées disponibles pour ce TP.

Essayez de calculer automatiquement le seuil en utilisant la méthode de Fisher (méthode vue en cours).

3.3• Seuillage

Le seuil ayant été déterminé dans la phase précédente, on peut obtenir une image binaire de la façon suivante :

```
seuil = ??? % vous donnez une valeur au seuil
ImageBinaire = zeros( Nlin, Ncol ) ;
for lin=1:Nlin
  for col=1:Ncol
    if( ImageChocolat(lin,col) < seuil )
      ImageBinaire(lin,col) = 1 ;
    end
  end
end
```

Pour afficher l'image binaire, il faut réduire la palette de couleurs :

```
figure(22) ; image(uint8(ImageBinaire)) ;
colormap(gray(2)) ; axis('image') ;
```

Effectuer le travail à l'aide éventuellement d'un script, analysez le résultat en fonction du seuil choisi (faites varier les seuils).

3.4• Obtention d'une forme pleine.

Pour "boucher" les trous qui peuvent apparaître après le seuillage, on utilise un opérateur logique appelé fermeture morphologique. Une fermeture morphologique est une succession de deux opérations morphologiques de base qui sont

- une dilatation qui grossit les formes blanches
- une érosion qui grossit les formes noires (c'est une opération duale).

Faites des essais d'érosion et de dilatation.

```
ImageDilatee = bwmorph(ImageBinaire,'dilate') ;
ImageErodee = bwmorph(ImageBinaire,'erode') ;
ImageFermee = bwmorph(ImageBinaire,'close') ;
ImageOuverte = bwmorph(ImageBinaire,'open') ;
```

Vérifiez que la fermeture est bien obtenue par la succession d'une dilatation et une érosion. Que se passe-t-il si on fait 2 dilatations et deux érosions ? 3 ? expliquez le résultat.
Vous pouvez aussi calculer les surfaces attribuées à chaque classe (fond / forme) grâce à la fonction bwarea(ImageDilatee) qui renvoie le nombre de pixels différents de 0.
Remarque : les images que vous obtenez sont binaires (1 ou 0). Pour les afficher il faut les multiplier par 255 ou changer l'échelle de niveaux d'affichage (colormap).

3.5• Segmentation

La fonction bwlabel(imb) permet de réaliser la segmentation de l'image c'est à dire de marquer tous les pixels liés à une première forme par un 1 puis tous les pixels liés à une 2° forme d'un 2 etc... La syntaxe est :

```
[ImageDesLabels,NombreDeLabels] = bwlabel(ImageBinaire);
```

`NombreDeLabels` est le nombre de formes trouvées. Regardez l'image `ImageDesLabels` et commentez.
Pour associer une couleur à chaque nombre de forme, vous pouvez procéder ainsi :

```
ImageColoree = ind2rgb(ImageDesLabels+1,[0 0 0;jet(NombreDeLabels)]);
figure(2) ; imshow(ImageColoree);
```

On peut récupérer les coordonnées de tous les pixels de la forme numéro n par la commande :

```
[Coo_lin,Coo_col] = find(ImageDesLabels == n);
```

Faites un script qui va créer une image (à niveau de gris) par chocolat isolé.