

### Phase préparatoire : statistiques des classes

L'objectif de cet exercice est de manipuler une image couleur. Pour cela vous allez travailler avec l'image 'coloredChips.png'.

- 1) Charger cette image sous matlab en tapant `I=imread('coloredChips.png');`
- 2) Quelles sont les dimensions de cette image ? quel est le nombre de composantes
- 3) Afficher dans la même fenêtre l'image couleur et ses trois composantes indépendantes R(red) , G(green) et B (blue). Vous utiliserez pour cela la fonction subplot (help subplot) donner un titre à chacune de vos images :

Exemple 'RGB' pour l'image couleur, 'Composante R' pour l'image R,...

- 4) Pour réaliser la classification, on a besoin de déterminer à priori les paramètres des classes qui sont présentes dans l'image. Pour cela on va créer une variable `names` contenant le nom de chaque classe, comme suit :

```
names={'pieces jaunes','pieces rouges','pieces vertes','pieces  
bleues','pieces oranges','stylo','fond'};
```

définissant 7 classes. Les paramètres de chaque classe, c'est-à-dire la moyenne  $\mu$  et l'écart type  $\sigma$  seront calculés par la fonction :

```
[M,S]=BuildClass(I,tb,names);
```

Ou **I** désigne l'image, **tb** la taille du bloc de l'échantillon et **names** le tableau des noms des classes. La fonction retourne **M** et **S**, les tableaux des moyennes et des écart-types. La fonction permet de prendre plusieurs échantillons de la même classe et fournir une valeur moyenne de la moyenne  $\mu$  et l'écart type  $\sigma$ .

AN : vous fixerez `t_b=10`.

- a) Vous prendrez un échantillon par classes. Vous sauvegarderez les résultats en tapant : `save stat1 M S`
- b) Vous prendrez trois échantillons par classes. Vous sauvegarderez les résultats en tapant : `save stat3 M S`

- c) Vous prendrez cinq échantillons par classes. Vous sauvegarderez les résultats en tapant : `save stat5 M S`

## Classification de Bayes

Dans cette partie, nous allons étudier la classification de Bayes.

Vous allez écrire un script nommé **Ex\_Cb** dans lequel vous exécuterez les commandes suivantes :

- 1) Charger l'image 'coloredChips.png'
- 2) Charger le fichier 'stat1.mat'
- 3) La classification Bayésienne est réalisée par la fonction Cb.m fournie en TP et définie par :

```
[A, ID, centers]=Cb(I, M, S) ;
```

La fonction reçoit l'image I, le tableau des moyennes M et le tableau des écarts types S. Elle retourne l'image couleur A qui représente le résultat de la classification, le tableau unidimensionnel ID représentant le n° de classe de chaque pixel de l'image I et centers un tableau de taille 3\*nombre de classes qui représente la moyenne RGB (ligne) de chaque classe (colonne)

- (a) Dans une figure Afficher à la fois I et l'image A.
- (b) La classification est t'elle parfaite ? Commentez vos résultats...
- (c) Refaite les questions a) et b) avec les fichiers stat3 et stat5

- d) Comment évolue la classification de stat1 à stat5 ?
- e) Comment évolue le pourcentage d'erreur de classification entre stat1 et stat3 et entre stat3 et stat5.

(Aide : Pour chaque expérience vous garderez en mémoire le tableau ID en définissant ID1 pour stat1, ID3 pour stat3 et ID5 pour stat5. Pour calculer le pourcentage d'erreur de classification, vous comptabiliserez le nombre de pixels mal classés entre 2 expériences en utilisant le tableau ID)

## Affichage de la classification dans l'espace RGB

- 1) Construire une matrice de données  $X$  où la 1<sup>ère</sup> colonne regroupe les pixels de la composante R, la 2<sup>ème</sup> colonne la composante G et la 3<sup>ème</sup> colonne la composante B.
- 2) Le résultat de la fonction  $Cb$  est un vecteur ID qui donne une étiquette pour chaque pixel de l'image. A partir du vecteur ID des n° de classes et  $X$  la matrice des pixels dans l'espace RGB, afficher la classification dans l'espace RGB.  
(aide : définissez un tableau de couleurs et à l'aide de la fonction `plot3` de matlab afficher les pixels avec une couleur correspondant à leur classe.

## Classification par les k moyennes

Dans cette partie, nous allons étudier la classification par les k moyennes.

Vous allez écrire un script nommé **Ex\_Km** dans lequel vous exécuterez les commandes suivantes :

- 4) Charger l'image '`coloredChips.png`'
- 5) La classification par k moyennes est réalisée par la fonction « `Km.m` » fournie en TP et définie par :

```
[A,ID,centers]=Km(I,K) ;
```

La fonction reçoit l'image  $I$  et  $K$  détermine le nombre de clusters à priori dans l'image. Elle retourne l'image couleur  $A$  qui représente le résultat de la classification, le tableau unidimensionnel  $ID$  représentant le n° de classe de chaque pixel de l'image  $I$  et  $centers$  un tableau de taille  $3 \times \text{nombre de classes}$  qui représente la moyenne RGB (ligne) de chaque classe (colonne)

Pour les valeurs de  $K=(3,6,7,10)$ , réalisez les différents points ci-dessous

- (a) Dans une figure Afficher à la fois  $I$  et l'image  $A$ .
- (b) La classification est-elle parfaite ? Commentez vos résultats...
- (c) Comment évolue la classification ? (étude visuelle)

- (d) **Prenez le meilleur niveau de classification obtenue par la méthode de Bayes (ID) et calculez le pourcentage d'erreur obtenu pour les valeurs de K proposées.**

### Classification par le Mean shift

Dans cette partie, nous allons étudier la classification par le mean shift.

- 1) La classification par Mean shift est réalisée par la fonction « Ms.m » fournie en TP et définie par :

```
[A, ID, centers]=Ms (I, R) ;
```

La fonction reçoit l'image I et R détermine la valeur du rayon du cercle englobant. Elle retourne l'image couleur A qui représente le résultat de la classification, le tableau unidimensionnel ID représentant le n° de classe de chaque pixel de l'image I et centers un tableau de taille 3\*nombre de classes qui représente la moyenne RGB (ligne) de chaque classe (colonne)

- **Pour les valeurs de R=(0.1,0.2,0.3,0.4,0.5), faite la même étude que précédemment.**

$$\frac{P(X/C_i)}{P(X/C_j)} \begin{cases} > \frac{P(C_j)}{P(C_i)} \\ < \frac{P(C_j)}{P(C_i)} \end{cases}$$

- $P(X/C_i)$  définit comme une loi normale de dimension  $d$  :

Avec  $\mu_i$  représente la moyenne de la classe  $C_i$ ,  $E_i$  la matrice de variance-covariance de la classe  $C_i$  définie ci-dessous :

$$Ei = \begin{bmatrix} \sigma_1^2 & \sigma_1\sigma_2 \dots & \sigma_1\sigma_d \\ \vdots & \ddots & \vdots \\ \sigma_d\sigma_1 & \dots & \sigma_d^2 \end{bmatrix}$$

- $P(C_i)$  la probabilité d'apparition de la classe  $C_i$

L'expression  $p(X/C_i)$  représente une formulation générale de la loi de probabilité lorsque les variables aléatoires formant X ne sont pas indépendantes. Dans le cas où les variables sont toutes indépendantes,  $p(X/C_i)$  se simplifie par :

$$p^{(X/C_i)} = p^{(X_1/C_i)} p^{(X_2/C_i)} \dots p^{(X_d/C_i)}$$

avec  $p(X_k/C_i)$  est la loi normale unidimensionnelle de la classe  $C_i$  pour la composante  $k$ .

- 1) Un pixel  $X$  de l'image couleur peut être considéré comme un vecteur aléatoire à 3 dimensions tel que :

$$X = (R, G, B)$$

Ou  $(R, G, B) \in [0, \dots, 255]^3$ . Si l'on considère que les 3 composantes sont indépendantes, donnez une expression simplifiée de  $(X/C_i)$ .

- 2) En considérant que toutes les classes sont équiprobables, donnez l'expression de la fonction de décision de Bayes  
3) En déduire un algorithme de classification :

Voici les étapes principales de cet algorithme (faire un script) :

- Charger l'image en mémoire
- Séparer les composantes de l'image à l'aide de 3 variables :  $R, G, B$
- Transformer chaque composante en vecteur colonne.
- Standardiser les composantes  $R, G, B$  par rapport à la classe  $k$ :

$$R_k = \frac{R - \mu_k(R)}{\sqrt{2}\sigma_k(R)}, \quad G_k = \frac{G - \mu_k(G)}{\sqrt{2}\sigma_k(G)}, \quad B_k = \frac{B - \mu_k(B)}{\sqrt{2}\sigma_k(B)}$$

Ou

$\mu_k = (\mu_k(R), \mu_k(G), \mu_k(B))$  est la moyenne de la classe  $k$  et  
 $\sigma_k = (\sigma_k(R), \sigma_k(G), \sigma_k(B))$  est son écart type.

Faite une fonction permettant de calculer ses nouvelles composantes

$$[R_k, G_k, B_k] = \text{Standard\_Ck}(R, G, B, \mu_k, \sigma_k)$$

- e) Faite une fonction calculant la distance  $d_k$ , c'est-à-dire la distance de chaque pixel de l'image par rapport à la classe  $k$  :

$$d_k = (R_k^2 + G_k^2 + B_k^2) - \log(\sigma_k(R) \cdot \sigma_k(G) \cdot \sigma_k(B))$$

$$[d_k] = \text{Calcul\_dk}(R_k, G_k, B_k, \sigma_k)$$

- f) Dans une boucle, construisez une variable  $D$  qui regroupe tous les vecteurs  $d_k$  :

Voici le pseudo code : considérons K classes

```

M= [ $\mu_1, \mu_2, \dots, \mu_K$ ]
S= [ $\sigma_1, \sigma_2, \dots, \sigma_K$ ]
D=[] ;
Pour k=1 : K
    [ $R_k, G_k, B_k$ ] = Standard_Ck(R, G, B,  $\mu_k, \sigma_k$ ) ;
    [ $d_k$ ] = Calcul_dk( $R_k, G_k, B_k, \sigma_k$ ) ;
    D=[D, dk] ;
Fin pour

```

Questions:

- Que représente les matrices M et S. Décrivez les ?
- Que réalise ce bout de programme ?
- afficher sur la même figure la distance dk obtenue pour chaque classe k. Utilisez « hold on »

g) le principe de la règle de bayes est de déterminer pour chaque pixel de l'image sa classe k. Pour cela, le pixel i appartiendra à la classe k si :

$$k = \max_{j=1:K} (D(i,j))$$

Rq : N'oubliez pas que D est une matrice dont le nombre de ligne est le nombre de pixels de l'image et le nombre de colonne indique le nombre de classes. Donc, la colonne k représente la distance de chaque pixel de l'image par rapport à la classe k.

La fonction **max** de matlab permet de calculer la plus grande composante d'un vecteur v mais aussi de fournir son index. Exemple :

```

>> v=[3,5,9,3,12,2,1];
>> [vmax,k]=max(v)

vmax = 12
k = 5

```



**IMPORTANT** : si v plusieurs composantes maximales identiques ne sélectionner que la première

en vous basant sur cette fonctionnalité construisez un vecteur de décision « dec » permettant d'associer une classe à chaque pixel de l'image.

- h) Le vecteur "dec" a été construit précédemment, transformer ce vecteur en une matrice de la taille de l'image. Utiliser la fonction « reshape » de matlab
- i) Afficher le résultat de la décision :

```
pas=floor(256/K) ;  
ng=0 :pas :255 ;  
for k=1 :length(Ng)  
    dec(dec==k)=ng(k) ;  
end  
  
figure(2)  
clf;  
imshow(uint8(dec))
```

Que fait ce bout de programme? Décrivez chaque étape ?

- j) Analyser vos résultats