



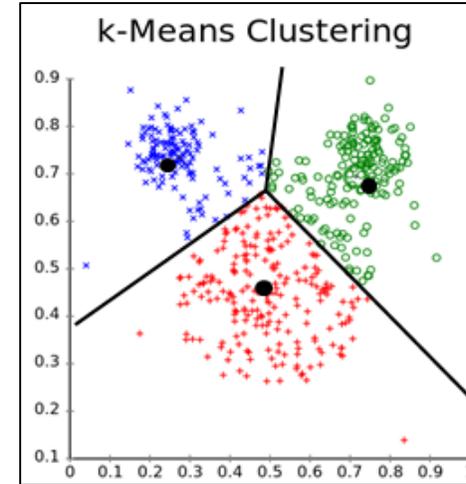
Stéphane GAZUT

Ingénieur-Chercheur

CEA, LIST, Laboratoire Analyse de Données et Intelligence des Systèmes

CEA Saclay – Gif-sur-Yvette

stephane.gazut@cea.fr



TP N°1 - KMEANS

MASTER SETI - INSTN
2021-2022

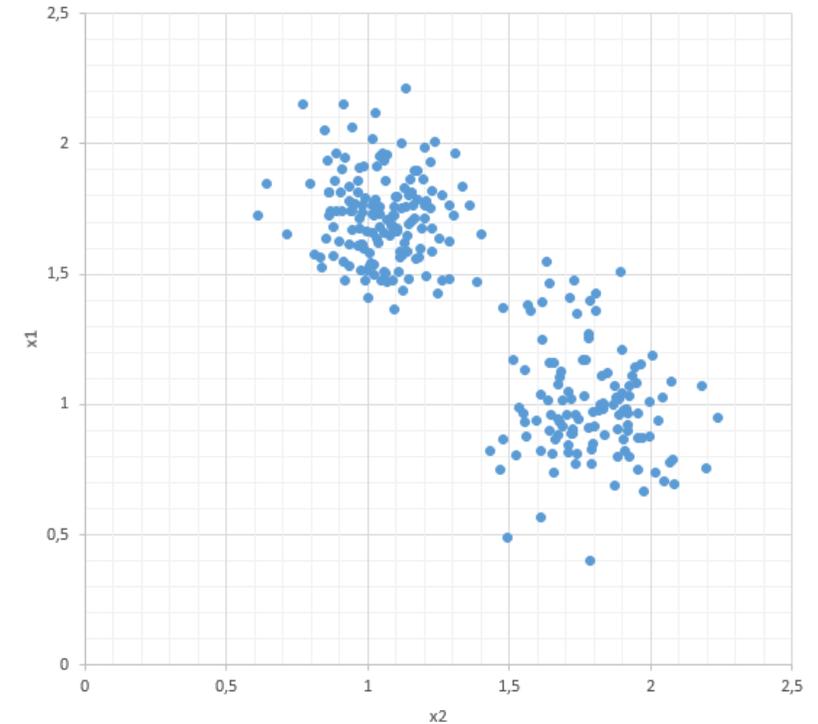


Objectifs du TP:

- Coder entièrement l'algorithme des kmeans
- Utiliser votre algorithme pour faire des tâches de clustering simples (problèmes à 2 variables pour faciliter la visualisation):
 - Avec 2 classes
 - Avec plus de 2 classes
- Utiliser votre algorithme pour faire de la segmentation d'image sur l'information de couleur
- Vous pouvez utiliser le langage que vous voulez. Néanmoins, je vous suggère d'utiliser python qui pourra être utilisé pour les autres TP (avec l'utilisation de la bibliothèque scikit-learn).

QUELLES FONCTIONS SERONT UTILES DANS LE SCRIPT ? (1/2)

- **Fonction pour générer la base d'exemple:**
 - Générer des nuages gaussiens. Sous python vous pouvez utiliser la fonction `numpy.random.normal(mean, sd)` qui vous fait des tirages suivant une loi normale de moyenne mean et d'écart-type sd.
 - Cette fonction pourra être utilisée pour générer l'exemple contenant 2 classes et l'exemple contenant plus de 2 classes.
- **Une fonction de calcul de distance (on utilisera la distance euclidienne)**
 - Attention, ne pas se limiter au calcul de distance dans un espace de dimension 2, mais une fonction de distance pouvant être utilisée dans un espace de dimension quelconque.
 - Vous pouvez utiliser la fonction `scipy.spatial.distance.cdist`



QUELLES FONCTIONS SERONT UTILES DANS LE SCRIPT ? (2/2)

- **Une fonction correspondant à votre kmeans.** Elle prendra en paramètres la matrice contenant les exemples et le nombre de clusters souhaités K . Elle devra retourner les clusters et les prototypes.

x1	x2
0.9418643937	1.7401710261
0.8113623936	1.5762078657
0.9011927844	1.6243825542
1.2243903234	1.7544539823
1.0227994548	1.7759683623
1.0689269364	1.7100397125
1.0902117771	1.6799564775
1.0444369488	1.7332221366
1.2038781996	1.9843247469

K

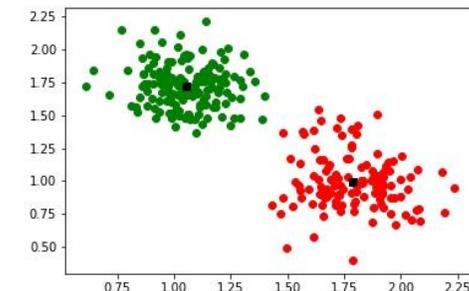
kmeans

- Centre des clusters
- $\{X_k\}$ pour les k clusters
- *les indices (type bool)*

- **Une fonction pour faire de la visualisation.** → Visualiser le résultat du clustering.

- Centre des clusters
- $\{X_k\}$ pour les k clusters

plot



LA FONCTION KMEANS

- **Chaque étape du kmeans doit être implémentée dans la fonction.**
- 1. Initialisation:** Choix aléatoire de K prototypes (ce peut être K points aléatoires de la base d'exemples). Ces points sont stockés (par exemple dans une structure de votre choix P_c – Positions centroïdes). Cette structure doit être indexée sur $k \in \llbracket 1, K \rrbracket$.
 - 2. Calcul de distance** des points aux différents centroïdes. Cette information peut être stockée dans une structure de votre choix, elle aussi indexée sur k .
 - 3. Création des clusters courants c-à-d affectation des points à leur centroïde le plus proche.** Vous pouvez utiliser une structure C indexée sur k qui contient les exemples du cluster k .
 - 4. Mise à jour des prototypes:** les nouveaux centroïdes sont les centres de gravité des clusters courants. Ces points peuvent être stockés dans une structure P_m (Position mobile) indexée sur k .
 - 5. Critère d'arrêt:** $|P_m - P_c| \leq \text{seuil}$? Si oui, retourner C et P_m sinon $P_m \rightarrow P_c$.

PETIT FOCUS SUR L'ÉTAPE 3

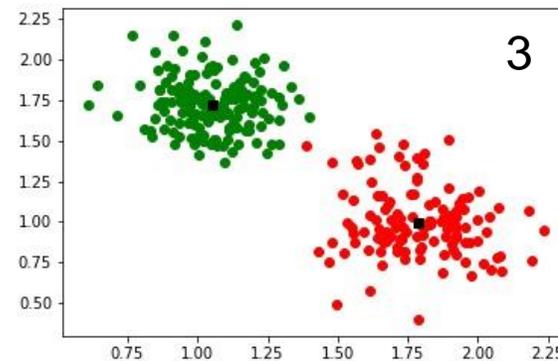
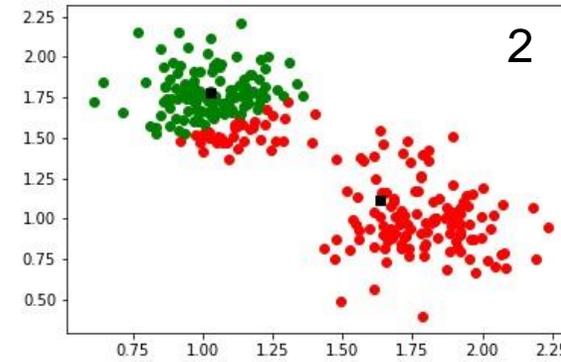
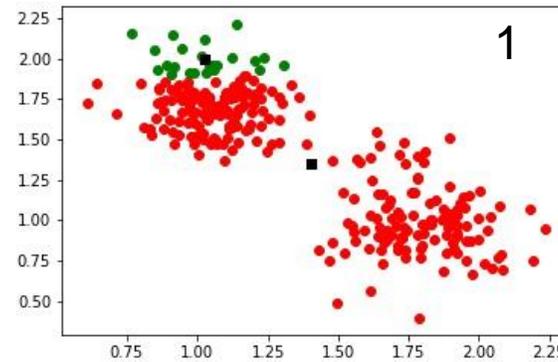
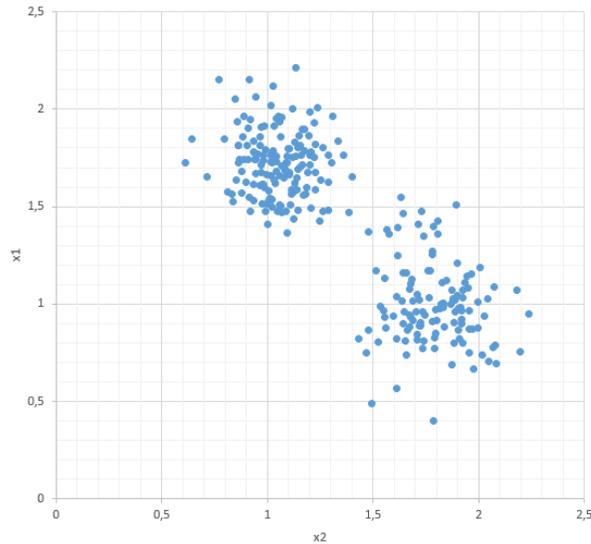
- Pour le $i^{\text{ème}}$ exemple et une structure de distances D (matrice contenant autant de lignes que d'exemples et autant de colonnes que de prototypes):

	K		
	1	2	3
...		...	
$i-1$	0,15	1,82	5,2
i	0,05	0,95	3,45
$i+1$	2,4	0,7	5,2
...		...	

- Le $i^{\text{ème}}$ exemple appartient au cluster 1, sa distance au cluster 1 $D[i,1] == \min(D[i,:])$
- Il faut faire le même type de test en « mode matriciel » et vous pouvez garder une structure booléenne `Index` (indexée sur k) et qui résulte du test logique
 - Vous pouvez utiliser la fonction `numpy.amin`

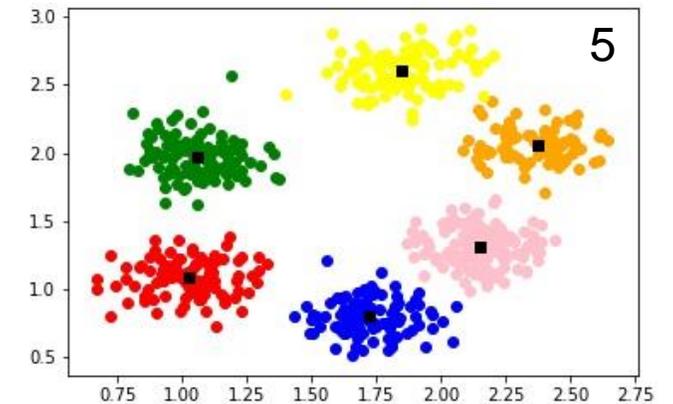
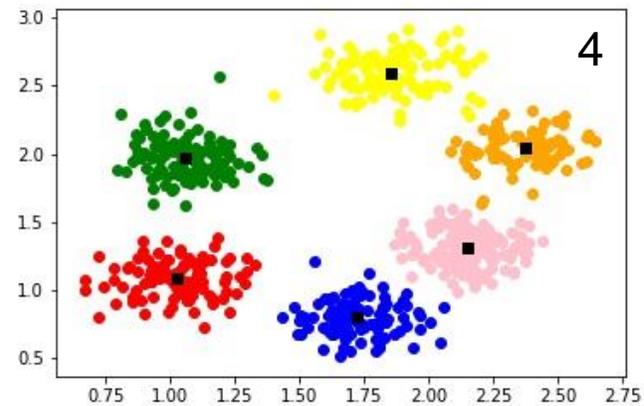
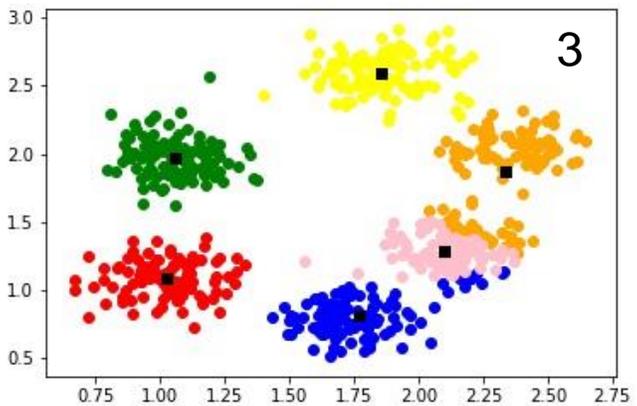
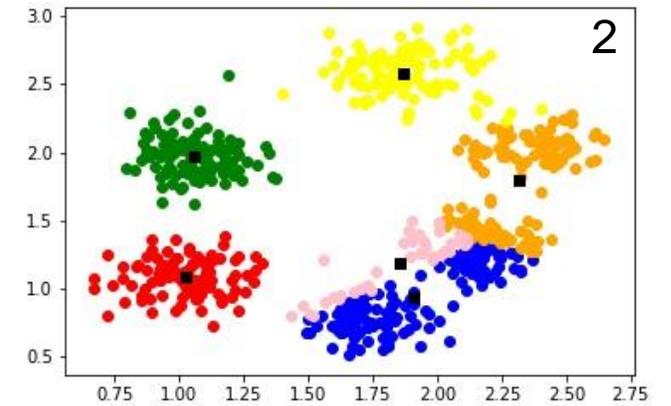
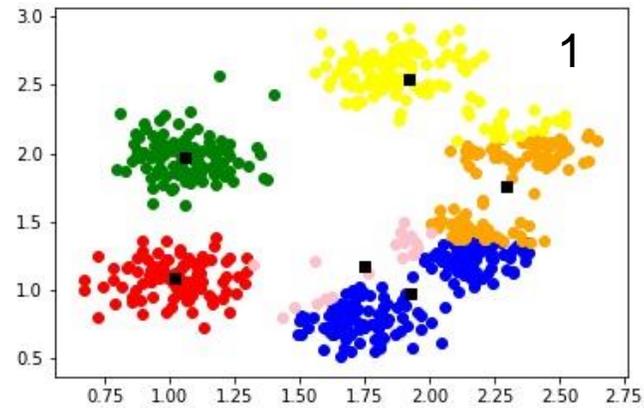
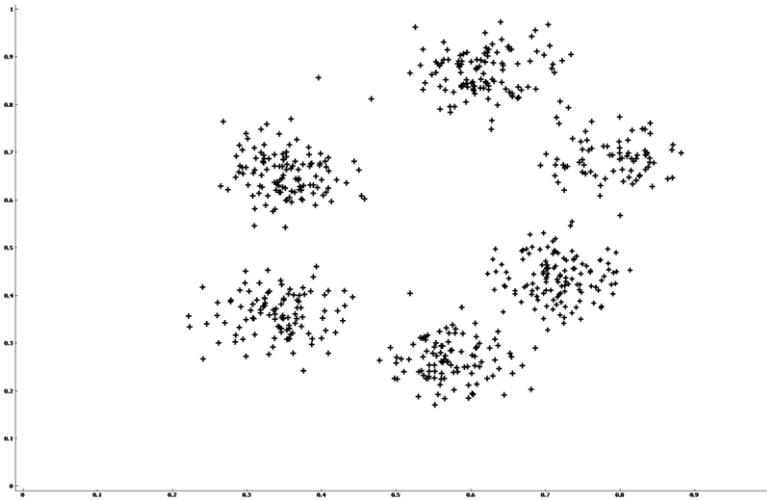
EXEMPLE À 2 CLUSTERS

- Une base d'exemples constitués en 2 groupes est disponible dans l'archive
- La fonction de visualisation peut être utilisée à chaque itération de l'algorithme



EXEMPLE À 6 CLUSTERS

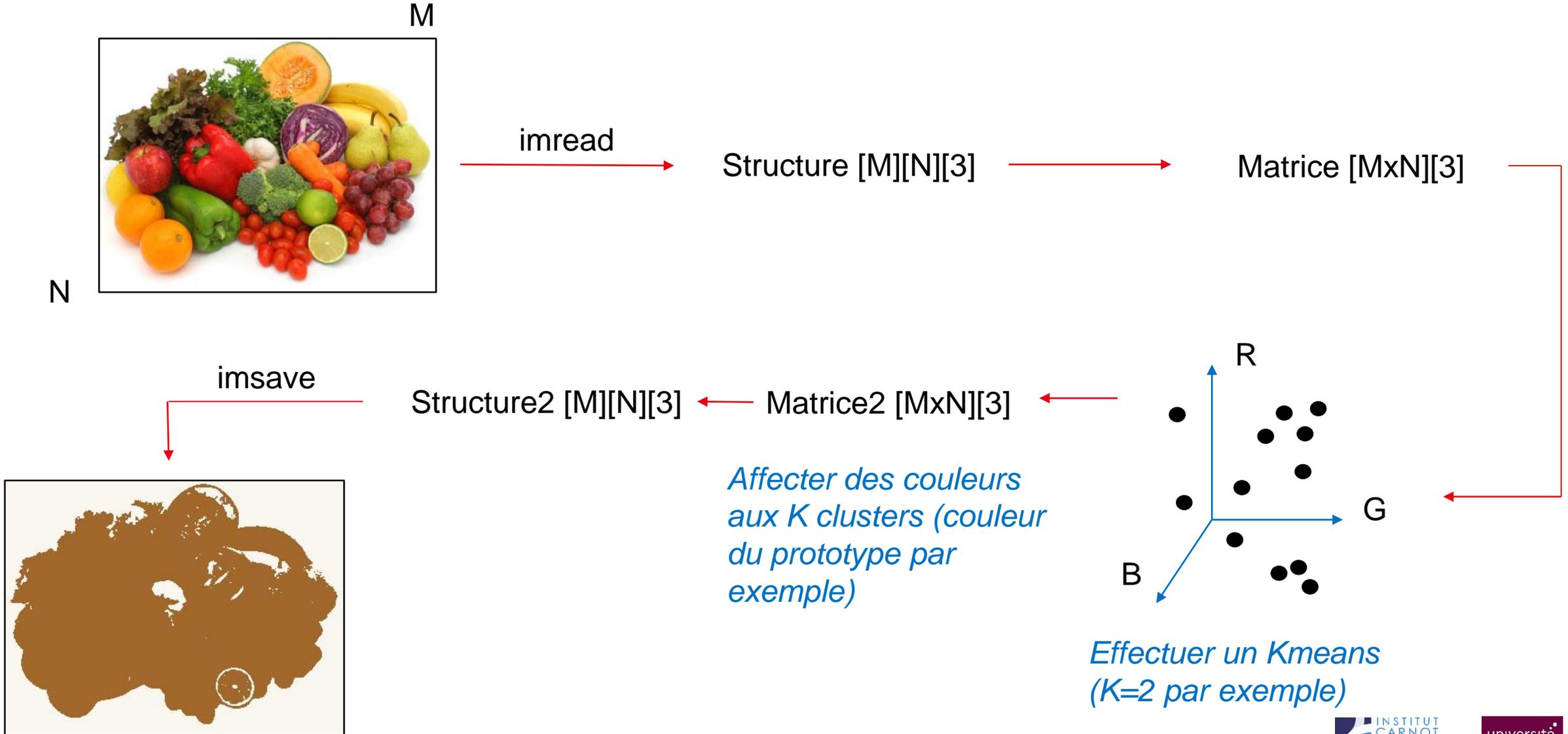
- Une base d'exemples constitués en 6 groupes est disponible dans l'archive



SEGMENTATION D'IMAGE SUR L'INFORMATION DE COULEUR

- Décrire les pixels d'une image dans l'espace colorimétrique à 3 dimensions R, G, B
- Réaliser un kmeans (identifier K prototypes)
- Affecter aux pixels la couleur de leur prototype le plus proche. La couleur d'un prototype est la couleur moyenne du cluster qu'il constitue.
- Reconstruire l'image avec ces nouveaux pixels. L'image ne contient alors que K couleurs.

SEGMENTATION D'IMAGE SUR L'INFORMATION DE COULEUR



STRUCTURE DE LA FONCTION

- Vous pouvez utiliser la bibliothèque skimage <https://scikit-image.org/> pour obtenir la structure MxNx3 à partir d'une image avec la fonction `skimage.io.imread()` ou pour sauvegarder une image à partir de la structure MxNx3 avec la fonction `skimage.io.imsave()`.

```
def kmeans_image(path_image, K):  
    my_img = io.imread(path_image)  
  
    # Transformation de la structure my_img NxMx3 en matrice pixels x 3  
    Mat = img_2_mat(my_img)  
  
    C, Index, Pc = kmeans_seti(Mat, K)  
  
    # Affecter aux pixels la couleur de leur prototype  
    for k in range(K):  
        Mat[Index[k],:] = Pc[k,:]  
  
    # Reconstruire la structure MxNx3 à partir de ces nouveaux pixels  
    img_seg = mat_2_img(Mat, my_img)  
  
    io.imsave(path_image.split('.')[0] + "_%d.jpg" % K, img_seg)
```

2 fonctions à définir

CE QUI EST ATTENDU

- **Vous devrez faire un compte rendu de TP contenant le code source et des illustrations des résultats de clustering pour:**
 - Le cas à 2 clusters en dimension 2
 - Le cas avec plus de 2 clusters en dimension 2
 - Choisir une image (pas trop grosse) et faire la segmentation pour différentes valeurs de K
- **Quelques pistes d'exploration**
 - Prendre une valeur de K qui ne correspond pas au nombre réel de groupes dans les données
 - Générer des nuages de points plus ou moins proches et/ou de densité différentes

Commissariat à l'énergie atomique et aux énergies alternatives
Institut List | CEA SACLAY NANO-INNOV | BAT. 861 – PC142
91191 Gif-sur-Yvette Cedex - FRANCE
www-list.cea.fr

Établissement public à caractère industriel et commercial | RCS Paris B 775 685 019