MASTER M2TI

PROJET DE SEGMENTATION D'IMAGES

2016-2017

Segmentation couleur avec les K-means

Vous devrez rendre un programme écrit en Python ainsi qu'un petit rapport présentant votre travail, les résultats obtenus, et toute remarque qui vous paraitrait pertinente.

Les K-means sont une méthode d'apprentissage non supervisé permettant de trouver des groupes parmi un nuage de points.

On peut la résumer ainsi :

```
Données :  Un ensemble de E de points de l'espace,

           K, un entier désignant le nombre de groupes que l'on
souhaite faire


1. Soit N un ensemble de k points placés au hasard dans l'espace

2. Associer chaque point de E au point de N qui lui est le plus
proche

3. Déplacer chaque point de N sur le barycentre des points de E qui
lui sont associés

4. Répéter les étapes 2-4 jusqu'à stabilité des points de N
```

Comme nous l'avons vu en cours, cet algorithme peut être utilisé pour regrouper les pixels d'une image couleur, et diminuer au final le nombre de couleurs présentes dans l'image. Le but de ce projet sera d'implémenter cet algorithme afin de réaliser cette tâche.

1. Le fichier **KMeans-projet.py** contient du code pour vous aider à démarrer le projet. Une image est lue et transformée de façon à obtenir, pour chaque pixel de l'image, trois tableaux (*bleu*, *vert*, *rouge*) qui seront les coordonnées des pixels de l'image (lignes 38 à 40 du fichier).

Ensuite, dans les lignes 46 à 48 du fichier, on créée trois tableaux (*cluster_bleu*, *cluster_vert*, *cluster_rouge*) qui contiendront les coordonnées des clusters de l'images (les centres des groupes que l'on cherche à créer). On place ces centres au hasard dans les lignes 53 à 60 du fichier, en leur rattachant au hasard des pixels de l'image (à l'aide du tableau *groupe*).

Votre code devra être inséré à partir de la ligne 68, et réaliser en boucle les étapes 2 à 4 du pseudo-code précédent, en inversant les étapes 2 et 3 : vous devrez tout d'abord déplacer chaque cluster sur le barycentre des pixels qui lui sont associés, puis associer chaque pixel (à l'aide du tableau *groupe*) au cluster qui lui est le plus proche.

2. Il est possible d'utiliser un autre espace de couleurs que l'espace RVB. Essayez votre algorithme en utilisant l'espace de représentation Lab. Vous trouverez ici des explications sur comment convertir une image de BVR vers HSV à l'aide d'OpenCV :

http://opencv-python-tutroals.readthedocs.org/en/latest/py_tutorials/py_imgproc/py_colorspaces/py_colorspaces.html

A vous de trouver comment modifier légèrement le code afin de convertir l'image vers l'espace Lab au lieu de HSV.

Grâce aux images exemples données, trouvez-vous des différences de résultats entre l'utilisation de BVR et Lab (tentez de diminuer le nombre de clusters, et voir quel espace semble donner les résultats les plus élégants) ?

3. Testez votre algorithme sur des images noir et blanc (par exemple, celles vue lors du TP sur la méthode d'Otsu). Que donne votre algorithme lorsque vous tentez de trouver deux clusters seulement (est-ce mieux ou moins bien qu'Otsu) ?

4. **Question de recherche expérimentale** Un problème avec cet algorithme est qu'il faut connaître le nombre de clusters que l'on souhaite obtenir… Pour tenter d'évaluer si le nombre de clusters est bon ou pas, on pourrait répéter plusieurs fois l'algorithme et voir si les attributions de groupes sont stables d'une itération à une autre. Si elles ne le sont pas, il y a de fortes chances que le nombre de clusters soit trop bas/élevé.

Testez cette hypothèse sur les images Mimi1 (à peu près 10-12 couleurs), Mimi2 (entre 10 et 12 couleurs), Mimi3 (entre 8 et 11 couleurs) et Caillou (entre 8 et 10 couleurs). Les résultats sont-ils stables lorsque le nombre de clusters est très bas/très grand/proche de la réalité ? Pouvez-vous proposer une mesure de stabilité permettant de quantifier si un nombre de cluster choisi est pertinent ?

Pour différentes images, tracez pour différentes valeurs de K, la valeur de votre mesure de stabilité : lorsque le nombre K vaut la valeur attendue, que donne votre mesure de stabilité ?