Compte rendu 3

Emilien Rey
Juan José Parra Díaz

semaine 3

1 Superpixels

Les superpixels sont utilisés dans différents domaines comme la reconnaissance d'objets, la segmentation d'images médicales ou encore comme dans notre cas la compression d'images.
Il existe principalement 2 différents types d'algorithmes pour les superpixels. Il y a les algorithmes basés sur les graphes et ceux sur l'ascension de gradient (Gradient-ascent-based algorithms).
La première méthode utilise un graphe où les nœuds représentent les pixels et les arêtes le poids proportionnel à la similarité entre les pixels des 2 nœuds. La deuxième méthode fait des clusters de pixels et itère sur ces clusters afin de les raffiner jusqu'à convergence. SLIC, Watersheds et Turbopixels appartiennent à cette catégorie d'algorithmes.

2 SLIC

Les différents algorithmes de superpixels souffrent de plusieurs problèmes comme un haut coût de calcul, des segmentations de mauvaise qualité, des formes et tailles incohérentes et plusieurs paramètres difficiles à régler. SLIC (Simple Linear Iterative Clustering) propose une approche rendant des superpixels de bonne qualité, compacts et presque uniformes tout en permettant à l'utilisateur de paramétrer le nombre de superpixels de l'image. Le seul inconvénient de l'algorithme est son haut coût de calcul.

SLIC réalise des clusters de pixels dans un espace en 5D défini par les valeurs L, a et b de l'espace couleur CIELAB et les coordonnées x et y des pixels. CIELAB est un espace couleur défini par L la clarté et a et b deux valeurs exprimant l'écart de la couleur par rapport à celle d'une surface grise de même clarté.

SLIC procède de la manière suivante, on regroupe les pixels basés sur leur similarité de couleur et proximité dans le plan de l'image. On fait cela dans l'espace en 5D [labxy]. Afin de faire ce clustering dans cet espace, on fait une mesure de distance qui considère la taille du superpixel.
L'algorithme prend en entrée K le nombre souhaité de superpixels et une image avec N pixels. La taille d'un superpixel est donc théoriquement de $\frac{N}{K}$ pixels. Pour des superpixels de taille à peu près similaire, il devrait y avoir un centre d'un superpixel à un intervalle de $S = \sqrt{\frac{N}{K}}$.

On crée K centres de superpixels définis comme suit sur une grille régulière avec S comme intervalle entre les centres :

$$C_k = \left[l_k, a_k, b_k, x_k, y_k\right]^T, \quad k \in [1, K]$$

Comme les superpixels ont une taille moyenne de S^2, on limite la recherche des pixels qui leur sont assignés à une zone de $2S \times 2S$ autour de leur centre.

La mesure de distance D_s utilisée est la suivante :

$$d_{lab} = \sqrt{(l_k - l_i)^2 + (a_k - a_i)^2 + (b_k - b_i)^2}$$

$$d_{xy} = \sqrt{(x_k - x_i)^2 + (y_k - y_i)^2}$$

$$D_s = d_{lab} + \frac{m}{S} d_{xy}$$

D_s est la somme des distances lab et des distances dans le plan xy normalisée par l'intervalle S. La variable m permet de gérer la compacité d'un superpixel. Plus m est grand, plus la proximité spatiale est importante et les clusters sont plus compacts. m est compris entre 1 et 20.

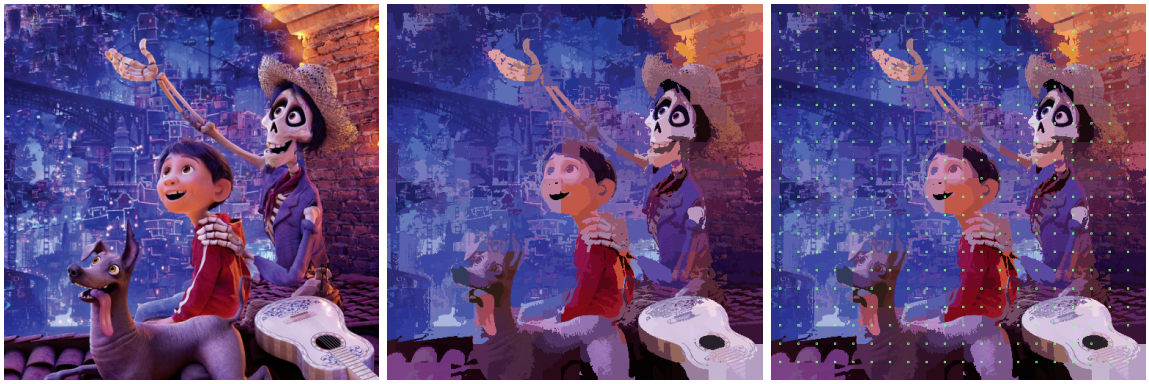
L'algorithme commence par calculer K centres de clusters de pixels. On place ces centres à la position du plus petit gradient dans un voisinage de 3 par 3 afin d'éviter que le centre se trouve sur une arête et éviter de choisir du bruit comme centre. Le gradient est calculé comme ceci :

$$G(x, y) = \|I(x + 1, y) - I(x - 1, y)\|^2 + \|I(x, y + 1) - I(x, y - 1)\|^2$$

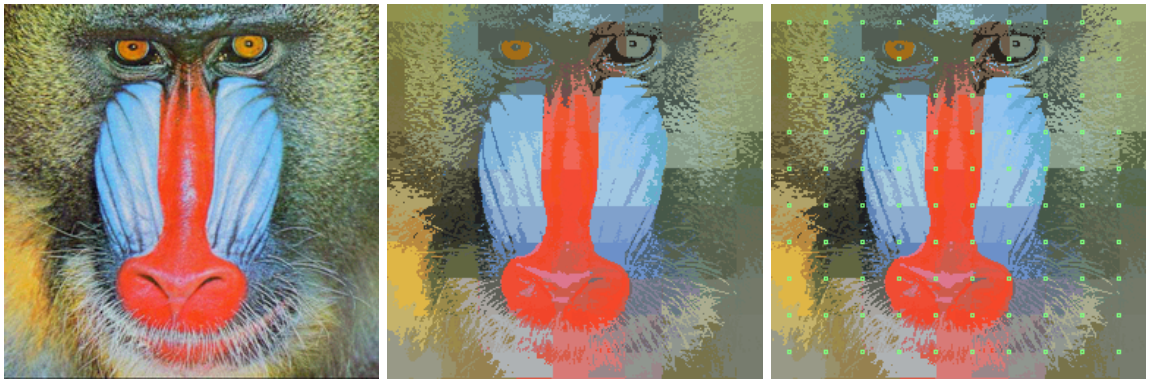
Ensuite chaque pixel est associé au cluster le plus proche dont l'espace de recherche à partir de son centre $(2S \times 2S)$ contient le pixel courant. Après que tous les pixels aient été associés à un cluster, un nouveau centre est calculé comme étant la moyenne du vecteur labxy de chaque pixel du cluster. On répète cela jusqu'à convergence.

Il existe une amélioration de l'algorithme SLIC appelée PREEMPTIVE SLIC qui réduit le temps de calcul tout en préservant la qualité des résultats. La principale différence est que SLIC utilise un unique critère d'arrêt pour le recalcul des clusters, ce qui fait qu'on recalcule des clusters qui n'ont pas de changements, ce qui augmente le temps de traitement. PREEMPTIVE SLIC propose d'utiliser des critères locaux afin d'éviter de recalculer des clusters où il n'y a pas eu de changements majeurs entre les itérations. De base, SLIC s'arrête soit lorsqu'un certain nombre d'itérations a été dépassé ou alors lorsque plus aucun changement n'est survenu entre 2 itérations.

Nous allons dans un premier temps nous concentrer sur l'implémentation de cet algorithme. Nous allons faire l'implémentation en C++ comme ça nous pourrons réutiliser les bibliothèques de traitement d'images que nous avons utilisées durant les TP. Pour l'instant, nous avons une première version qui fonctionne mais qui n'utilise pas encore le calcul de la distance avec des couleurs dans l'espace CIELAB et notre algo ne fait qu'une seule passe et ne recalcule pas les centres et les clusters. Notre code est surtout une base sur laquelle travailler par la suite.



(a) Image de base (b) 400 superpixels (c) Superpixels avec les centres



(d) Image de base (e) 100 superpixels. (f) Superpixels avec les centres

FIGURE 1 – Exemple de ce qu'on obtient avec notre version. On voit bien le découpage uniforme en grille, ce qui fait que les superpixels sont placés toujours à la même distance. On voit aussi que plus on a de superpixels et plus le résultat qu'on obtient est proche de l'image de base.

La prochaine étape consiste à finir l'implémentation de SLIC.