Logique floue pour la segmentation d'image couleur

Elliot Vanegue

15 décembre 2015

1 Introduction

Nous avons vu dans le TP précédent ce qu'était la logique floue au travers d'exemples ne portant pas sur la segmentation d'image. La logique floue permet en effet d'avoir un niveau d'incertitude sur l'appartenance d'une donnée à une classe. Dans ce TP, nous allons voir comment appliquer cette méthode à la segmentation d'image couleur. Par la suite, nous verrons différentes méthodes dérivées du principe de la logique floue afin de comparer leur performance. Tout au long de ce TP, nous utilisons l'image de la Fig. 1.

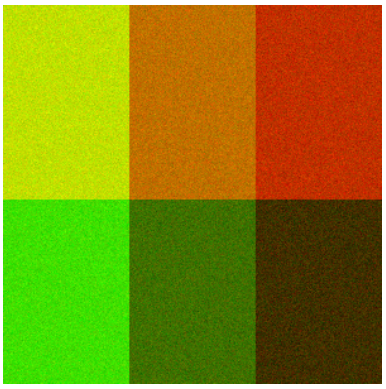

FIGURE 1 – Image de référence du TP

2 FCM (Fuzzy C-Means)

Dans un premier temps, nous utilisons la méthode de la logique floue pour segmenter une image couleur. Cette méthode est très proche de l'algorithme du K-means. La différence entre les deux algorithmes est que dans l'algorithme FCM une donnée n'appartient pas à une classe, mais elle a

une certaine probabilité d'y appartenir. Tout comme pour K-means, l'algorithme a besoin d'un certain nombre de paramètres pour fonctionner. Ces paramètres sont fournis par l'utilisateur.

— c : le nombre de classes à segmenter : ce paramètre détermine le nombre de centroïdes à créer.
— m : le degré d'appartenance d'une donnée : ce paramètre permet d'avoir un plus ou moins grand écart entre les taux d'appartenance aux classes. Plus il sera élevé, plus l'appartenance d'une donnée à une classe est forte. Cependant, si m est trop élevé, la correction des erreurs par les étapes suivantes est plus difficile.
— seuil : le seuil de stabilité à partir duquel l'algorithme peut s'arrêter.

Au début de l'algorithme, les centroïdes sont placés aléatoirement parmi les données. Puis à chaque itération, ces centoïdes se rapprochent du centre d'un ensemble de données, tout en s'éloignant les uns des autres. La détermination de la position des centroïdes se fait grâce au calcul (Eq. 1).

$$\forall i \in \{1, 2, ..., c\}\ v_i = \frac{\sum_{j=1}^{n} u_{ij}^m * x_j}{\sum_{j=1}^{n} u_{ij}^m} \tag{1}$$

Dans cette équation, n est le nombre de pixels. La matrice u représente le degré d'appartenance des pixels pour une classe. Il est possible de calculer cette matrice avec l'équation (Eq. 2).

$$u_{ij} = \left[\sum_{k=1}^{c} \left(\frac{d^2(x_j, v_i)}{d^2(x_j, v_k)} \right)^{\frac{2}{m-1}} \right]^{-1} \tag{2}$$

Ces deux calculs permettent donc de placer les centroïdes au centre des classes détectées. Il faut maintenant permettre à l'algorithme de s'arrêter lorsque le seuil de stabilité est atteint. Pour cela, il faut calculer la performance de l'étape qui a été calculée, c'est-à-dire qu'il faut minimiser le taux d'appartenance des pixels par la distance avec le centroïde (Eq. 3) et comparer ce résultat avec celui de l'étape précédente (Eq. 4).

$$J_{FCM}(P) = \sum_{i=1}^{c} \sum_{j=1}^{n} [u_{ij}]^m * d_{ij}^2 \tag{3}$$

$$J_{FCM}(P) - J_{FCM}(P-1) < seuil \tag{4}$$

Nous obtenons la segmentation de la Fig. 2, qui nous permet d'avoir un résultat très intéressant. On peut voir sur la courbe représentant $J_{FCM}(P)$ (Fig. 3), que l'algorithme fonctionne en très peu d'étapes.

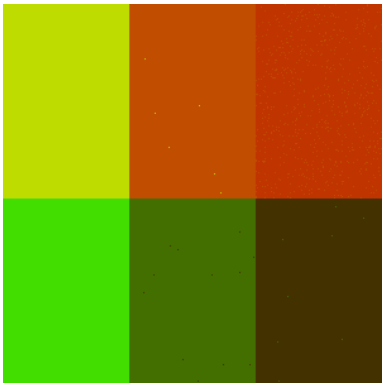

FIGURE 2 – Résultat de la segmentation d'une image couleur avec l'algorithme FCM

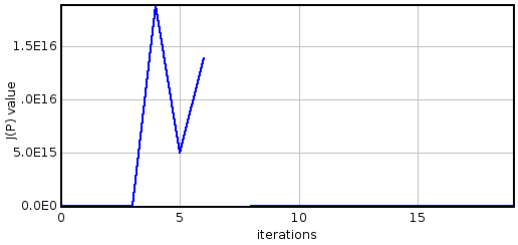

FIGURE 3 – Graphique de $J_{FCM}(P)$

3 HCM (Hard C-Means)

L'algorithme HCM est similaire à l'algorithme K-means. Il n'utilise pas de degré d'appartenance d'une donnée à une classe, soit la donnée appartient à une classe, soit elle n'y appartient pas. Le changement majeur dans l'algorithme du FCM est donc le calcul de u_{ij}. Pour déterminer à quelle classe appartient une donnée, on compare avec la distance de sa coordonnée avec chaque centroïde et on choisit la classe ayant la plus courte distance. Cela nous fournit le résultat de la Fig. 4.

Encore une fois, la segmentation est efficace et on voit que $J_{HCM}(P)$ fournit des résultats dès la première itétation. Cependant, l'algorithme met plus de temps à trouver la solution qu'avec FCM.

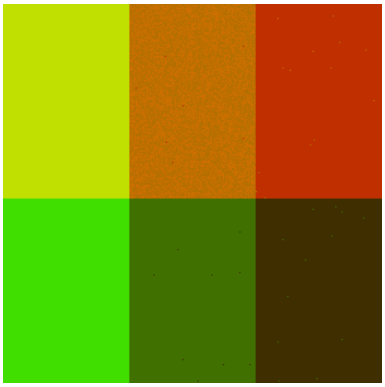

FIGURE 4 – Résultat de la segmentation d'une image couleur avec l'algorithme HCM

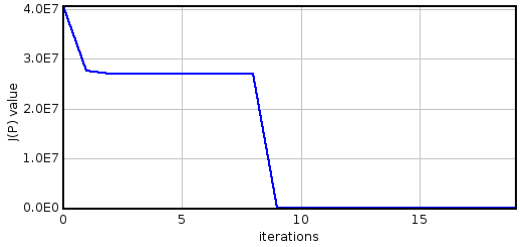

FIGURE 5 – Graphique de $J_{HCM}(P)$

4 PCM (Possibilistic C-Means)

L'algorithme PCM fonctionne sur le même principe que le FCM tout en introduisant une valeur de pénalité sur l'appartenance d'une donnée à une classe. Cette pénalité exclut les points trop éloignés de la classe. Le calcul de cette pénalité est donné dans l'Eq. 5.

$$\eta_i = \frac{\sum_{j=1}^{n} u_{ij}^m * x_j}{\sum_{j=1}^{n} u_{ij}^m} \tag{5}$$

Cette pénalité est appliquée au calcul de u_ij (Eq. 6) et à la performance de l'étape courante (Eq. 7).

$$\eta_i = \frac{1}{1 + \left(\frac{d_{ij}^2}{\eta_i}\right)^{\frac{1}{m-1}}} \tag{6}$$

$$J_{FCM}(P) = \sum_{i=1}^{c}\sum_{j=1}^{n}[u_{ij}]^m * d_{ij}^2 + \sum_{i=1}^{c}\eta_i \sum_{j=1}^{n}[1-u_{ij}]^m \tag{7}$$

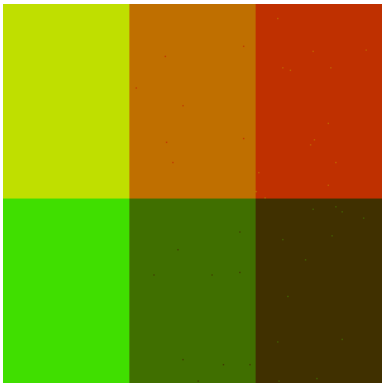

FIGURE 6 – Résultat de la segmentation d'une image couleur avec l'algorithme PCM

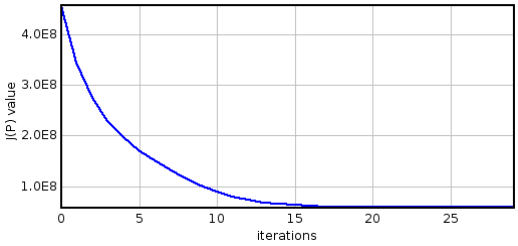

FIGURE 7 – Graphique de $J_{PCM}(P)$

Le résultat de l'algorithme est le même, tout en obtenant des résultat pour $J_{PCM}(P)$ dès les premières itération. Cependant, cet algorithme est un peu plus lent que HCM.

5 Algorithme de Davé

L'algorithme de Davé fonctionne a peu près de la même façon que l'algorithme PCM, sauf qu'il exclut les pixels trop éloignés de la classe en considérant que ceux-ci sont du bruit.

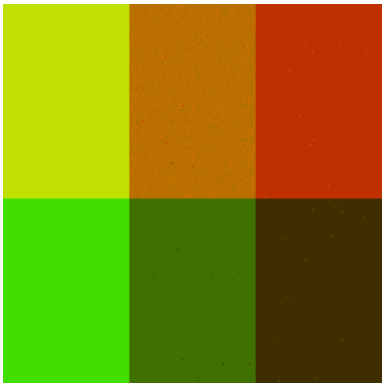

FIGURE 8 – Résultat de la segmentation d'une image couleur avec l'algorithme de Davé

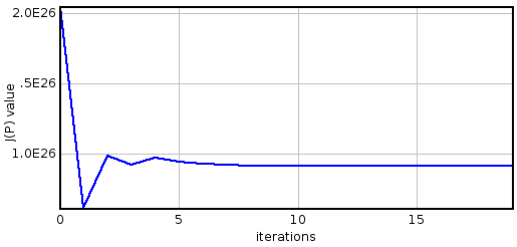

FIGURE 9 – Graphique de $J_{PCM}(P)$

6 Conclusion

Comme nous avons pu le voir au travers de ce TP, la logique floue permet d'avoir une segmentation équivalente à une méthode binaire, tout en trouvant le résultat plus rapidement. De plus, les résultats obtenus avec ce type de méthodes sont très intéressants, bien que ces méthodes aient besoin de nombreux paramètres.