TP1 – Segmentation par champs de Markov

Rappels de cours

Si on définit N classes gaussiennes ω_k, $k \in [1,N]$, de moyennes μ_k et d'écarts types σ_k, la segmentation d'une image en niveaux de gris $x = (x_s)$ se ramène à un problème de classification au sens du MAP. Le résultat est une matrice d'indices $k = (k_s)$ de mêmes dimensions que x : la valeur de $k_s \in [1,N]$ indique que s est affecté à la classe ω_{k_s}. La segmentation est obtenue en maximisant la probabilité a posteriori. La règle de Bayes s'écrit :

$$p(K=k|X=x) = \frac{p(X=x|K=k)\,p(K=k)}{p(X=x)} \propto p(X=x|K=k)\,p(K=k) \tag{1}$$

où $p(X=x|K=k)$ est la vraisemblance et $p(K=k)$ la probabilité a priori. Grâce à l'hypothèse d'indépendance des données conditionnellement à la carte de segmentation (cf. cours) :

$$p(X=x|K=k) = \prod_{s\in\mathcal{S}} p(X_s=x_s|K_s=k_s) = \prod_{s\in\mathcal{S}} \frac{1}{\sqrt{2\pi}\,\sigma_{k_s}} \exp\left\{-\frac{1}{2}\left(\frac{x_s-\mu_{k_s}}{\sigma_{k_s}}\right)^2\right\} \tag{2}$$

Le *modèle de Potts* définit un a priori qui régularise le résultat ($\beta > 0$ est un paramètre du modèle) :

$$p(K=k) \propto \exp\left\{-\beta \sum_{\{s,t\}\in\mathcal{C}_2} [1-\delta(k_s,k_t)]\right\} \tag{3}$$

où $\mathcal{C}_2$ désigne l'ensemble des cliques de cardinal 2, c'est-à-dire l'ensemble des paires $\{s,t\}$ de pixels voisins. On déduit des équations (1) à (3) la probabilité a posteriori de la configuration k :

$$p(K=k|X=x) \propto \exp\{-U(k)\} = \exp\left\{-\sum_{s\in\mathcal{S}}\left[\ln\sigma_{k_s} + \frac{1}{2}\left(\frac{x_s-\mu_{k_s}}{\sigma_{k_s}}\right)^2\right] - \beta\sum_{\{s,t\}\in\mathcal{C}_2}[1-\delta(k_s,k_t)]\right\} \tag{4}$$

Trouver le maximum de $p(K=k|X=x)$ équivaut donc à chercher le minimum de l'énergie $U(k)$. Pour ce faire, il ne suffit pas de chercher l'optimum de l'énergie locale en chaque pixel $s \in \mathcal{S}$, ce qui s'écrirait :

$$k_s^* = \arg\min_{k_s\in[1,N]}\left\{\ln\sigma_{k_s} + \frac{1}{2}\left(\frac{x_s-\mu_{k_s}}{\sigma_{k_s}}\right)^2 + \beta\sum_{t\in\mathcal{V}(s)}[1-\delta(k_s,k_t)]\right\} \tag{5}$$

car cette expression dépend des indices k_t aux pixels $t \in \mathcal{V}(s)$ voisins de s. On doit chercher le minimum global de l'énergie $U(k)$. Il est impensable de tester toutes les $N^{\mathrm{card}(\mathcal{S})}$ configurations possibles. On peut en revanche utiliser le *recuit simulé*. Cette *méta-heuristique* fait décroître un paramètre T, appelé *température*, en le multipliant par un coefficient $\alpha < 1$ à chaque itération (repérée par l'indice q). L'algorithme s'écrit :

1. $q \leftarrow 1$; $T \leftarrow T_0$; $K \leftarrow$ valeurs entières aléatoires tirées dans $[1,N]$.
2. Pour chaque pixel s de l'image, visitée ligne par ligne et colonne par colonne :
   - Tirer une nouvelle réalisation Nouv de K_s dans $[1,N]\backslash\{k_s\}$ et calculer les énergies locales suivantes :

$$\begin{cases} U_{\mathrm{Cour}} = \ln\sigma_{k_s} + \frac{1}{2}\left(\frac{x_s-\mu_{k_s}}{\sigma_{k_s}}\right)^2 + \beta\sum_{t\in\mathcal{V}(s)}[1-\delta(k_s,k_t)] \\ U_{\mathrm{Nouv}} = \ln\sigma_{\mathrm{Nouv}} + \frac{1}{2}\left(\frac{x_s-\mu_{\mathrm{Nouv}}}{\sigma_{\mathrm{Nouv}}}\right)^2 + \beta\sum_{t\in\mathcal{V}(s)}[1-\delta(\mathrm{Nouv},k_t)] \end{cases} \tag{6}$$

   - Si $U_{\mathrm{Nouv}} < U_{\mathrm{Cour}}$, alors $K_s \leftarrow$ Nouv. Sinon, $K_s \leftarrow$ Nouv avec la probabilité $\exp\{-(U_{\mathrm{Nouv}} - U_{\mathrm{Cour}})/T\}$.
3. Si $q < q_{\max}$, alors : $q \leftarrow q+1$; $T = \alpha\,T$; retour en 2.

Exercice 1 : segmentation d'une image en classes prédéfinies

Complétez le script `exercice_1.m`, de façon à segmenter l'image `image.bmp` selon la méthode décrite ci-dessus. Dans ce premier exercice, les paramètres (moyenne et écart type) des classes de pixels sont prédéfinis. En revanche, les quatre autres paramètres de la méthode de segmentation (T_0, $q_{\max}$, α, β) doivent être ajustés de façon à maximiser le pourcentage de bonnes classifications. Testez plusieurs jeux de valeurs pour ces paramètres, par exemple : $(1.0, 150, 0.99, 1.0)$; $(2.0, 250, 0.99, 5.0)$; $(1.0, 150, 0.9, 2.0)$; $(1.0, 250, 0.995, 2.0)$.

Faites une copie de ce script, de nom `exercice_1_bis.m`, que vous modifierez de façon à prendre en compte un modèle de Potts 8-connexe. Trouvez le jeu de paramètres qui optimise la segmentation de l'image `image.bmp`. Comparez ce résultat avec celui du modèle de Potts 4-connexe.

Pour le jeu de paramètres donnant le meilleur résultat avec le modèle 8-connexe, diminuez le nombre N de classes (qui vaut 6 par défaut) et observez la répercussion sur la segmentation.

Exercice 2 : segmentation par classification supervisée

On se propose maintenant d'apprendre les paramètres (moyenne et écart type) des différentes classes de pixels. Ceci est possible grâce à la fonction `estimation`, qui permet de sélectionner dans l'image des échantillons rectangulaires caractéristiques des différentes classes, en cliquant deux points considérés comme les sommets opposés d'un rectangle.

Complétez le script `exercice_2.m` en recopiant le morceau manquant dans le script `exercice_1_bis.m`. Montrez que les résultats sur l'image `image.bmp` sont comparables à ceux de `exercice_1_bis.m`, lorsque les échantillons des différentes classes sont correctement sélectionnés. Que se passe-t-il lorsque les échantillons sont mal sélectionnés ?

Faites une copie du script `exercice_2.m`, de nom `exercice_2_bis.m`, que vous modifierez de manière à lire l'image couleur `guadeloupe.jpg` et à la convertir en niveaux de gris (grâce à la fonction `rgb2gray`, cf. TP1 de TAV). Veillez à supprimer les dernières lignes du script `exercice_2.m`, car la « vérité terrain », c'est-à-dire le résultat attendu de la segmentation, n'est pas disponible pour cette image réelle. Testez le script `exercice_2_bis.m` pour différentes valeurs du nombre N de classes. Que pensez-vous des résultats obtenus ?

Exercice 3 : utilisation des trois canaux RVB

Pour segmenter une image telle que `guadeloupe.jpg`, il est dommage de ne pas utiliser l'information de couleur. Faites une copie du script `exercice_2_bis.m`, de nom `exercice_3.m`, que vous modifierez de manière à utiliser les trois canaux RVB. Il est rappelé que la loi normale en dimension d s'écrit :

$$p(x) = \frac{1}{(2\pi)^{d/2}\sqrt{\det\Sigma}}\,\exp\left\{-\frac{1}{2}\,(x-\mu)^\top\Sigma^{-1}(x-\mu)\right\} \tag{7}$$

où $x \in \mathbb{R}^d$, $\mu \in \mathbb{R}^d$ désigne la moyenne et $\Sigma \in \mathbb{R}^{d\times d}$ est la matrice de variance/covariance. La fonction `estimation` doit être remplacée par `estimation_RGB`, qui retourne pour chaque classe : la moyenne, l'inverse de la matrice de variance/covariance et le logarithme de son déterminant.

Quelques remarques pour la réalisation de cet exercice :
- La méthode devient très lente à cause de l'utilisation des $d = 3$ canaux RVB.
- Pour cette même raison, il vaut mieux afficher la carte de segmentation toutes les 2 itérations.
- L'affichage d'une image couleur avec la fonction `imagesc` nécessite que les niveaux de couleur soient normalisés entre 0 et 1.
- En revanche, il devient facultatif de spécifier une palette des couleurs (à l'aide de la fonction `colormap`).