BE – Segmentation d'image supervisée

Le but de ce BE est d'appliquer et de combiner des méthodes d'Analyse de Données pour segmenter une image.

Vous déposerez les codes matlab sous format .zip (BE_nom.zip) à la fin de chaque séance de TP.

Première étape : les données et les textures en présence

La figure 1 présente une image (ImageGris.mat) en niveaux de gris où apparaissent 3 textures : grains, bois et pierres.

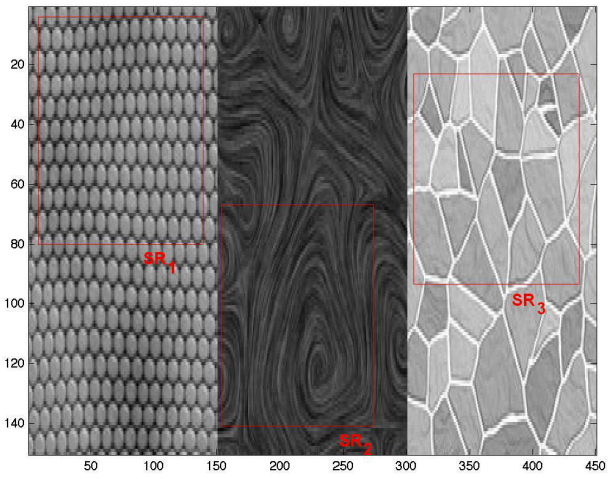
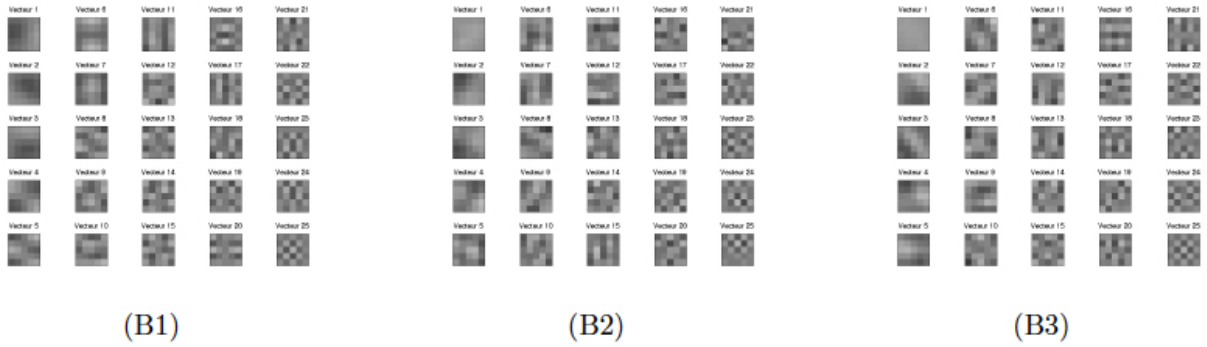


FIGURE 1 – Image de travail et les 3 sous-régions sélectionnées dans chaque texture

- Désignez de manière intéractive (à la souris) 3 sous-régions rectangulaires de l'image (1 sous-région par texture). Dans la figure 1, les sous-régions sont respectivement notées SR_1, SR_2 et SR_3.
  La fonction matlab `getrect.m` est utilisée pour sélectionner les sous-régions SR_i pour $i \in \{1, 2, 3\}$.
  La fonction fournie `split_into_blocs.m` permet de décomposer l'image en blocs :
  Par exemple, `split_into_blocs(...SR1...,ones([w,w]))` pour découper chaque sous-région en p petits blocs de taille $w \times w$ (ici $[w, w] = [5, 5]$).

- Complétez le script `PremiereEtape.m`, de manière à effectuer l'Analyse en Composantes Principales (ACP) de l'ensemble d'apprentissage stocké dans la matrice X de taille $w^2 \times p$. L'objectif est de construire grâce à 3 ACP, 3 bases à partir des blocs extraits des 3 sous-régions. La base construite sera notée B_1 (resp. B_2 et B_3) à partir de SR_1 (resp. SR_2 et SR_3).

La figure 2 suivante illustre les trois bases obtenues en visualisant les vecteurs propres de la matrice de variance/covariance. Notez que ces bases sont différentes : chaque base Bi dépend d'une texture particulière (la texture n°i).



(B1) (B2) (B3)

FIGURE 2 – 3 bases obtenues : caractéristiques des 3 textures

Deuxième étape : reconstructions par les différentes bases

Le but de cette partie est de détecter les 3 textures. En d'autres termes, il s'agit de segmenter/décomposer l'image en trois régions associées aux trois textures grâce à l'étape de supervision précédente où on a désigné des sous-régions caractéristiques de chaque texture. Bien entendu, le résultat attendu est connu ("on dispose d'une vérité terrain") puisque les 3 régions associées aux 3 textures forment une partition évidente de l'image en 3 bandes verticales (composée chacune de 150 pixels).

On propose d'utiliser les 3 bases précédentes pour reconstruire partiellement l'image originale (figure 1). Complétez le script `DeuxiemeEtape.m` en réalisant les étapes suivantes. Le résultat à obtenir est illustré par la figure 3.

- Décomposez l'intégralité de l'image en petits blocs (de même taille que celle utilisée précédemment).
- Reconstruisez approximativement tous ces blocs grâce aux k premiers vecteurs de B_1
  (si les blocs sont 5×5, on a 25 vecteurs dans B_1 et k, à choisir par vos soins, est donc inférieur à 25).
- Reformez une image reconstruite (approximativement) en replaçant les blocs reconstruits grâce à la fonction fournie `replace_blocs.m`.
- Procédez de même pour B_2 et B_3.

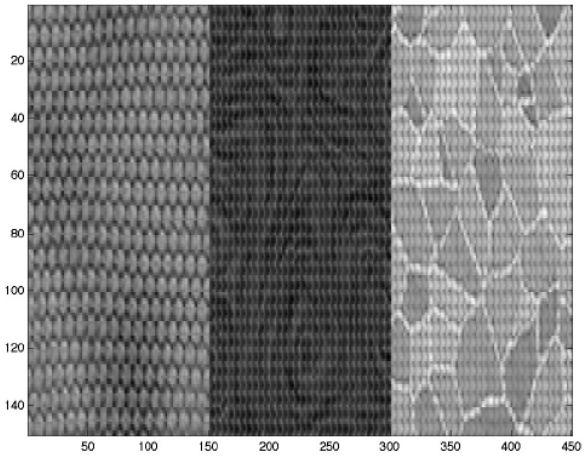


FIGURE 3 – L'image reformée après reconstruction de l'ensemble des blocs par B1

On peut alors comparer les 3 images reconstruites (approximativement) grâce aux 3 bases avec l'image originale en introduisant des mesures d'erreurs. Soient $Err_1(b)$ (resp. $Err_2(b)$, $Err_3(b)$) les erreurs commises (en norme L^2) en reconstruisant le bloc b avec la base B_1 (resp. B_2, B_3).

- Calculez, pour chaque bloc b, les mesures d'erreurs $Err_i(b)$, $i \in \{1, 2, 3\}$.
- Vérifiez que l'erreur de reconstruction d'un bloc obtenue avec la base B_i est d'autant plus faible si ce bloc appartient à la texture n°i échantillonnée par la sous-région SR_i.

La figure 4 suivante montre un résultat possible à l'issue des étapes précédentes. Les 3 niveaux de gris codent l'appartenance aux trois classes (c'est-à-dire aux 3 textures). Chaque pixel de cette image correspond à un bloc (5×5 ici). On peut observer que la texture centrale (N°2 texture bois) est correctement segmentée. Il y a davantages d'erreurs de classification pour la texture de droire (N°3 texture pierres) et encore plus pour la texture située à gauche (N°1 texture grains).

FIGURE 4 – Un résultat de segmentation possible à partir des analyses des textures précédentes

Troisième étape : intégration de la couleur

Afin d'améliorer la classification (et donc la segmentation obtenue), on ajoute pour chaque bloc une mesure de couleur en plus d'une caractéristique de texture.
On définit ainsi pour chaque bloc b les erreurs normalisées $\overline{Err_i}$ suivantes, pour $i \in \{1, 2, 3\}$:

$$\overline{Err_i}(b) = \frac{Err_i(b)}{\sum_{i=1}^{3} Err_i(b)}.$$

Soient $R(b)$ et $V(b)$ les caractéristiques de couleur du bloc b après la normalisation des couleurs des pixels. Rappelons que la normalisation des couleurs d'un pixel est définie par :

$$R = \frac{R}{R+V+B}, \quad V = \frac{V}{R+V+B},$$

(cf TP sur la classification bayésienne).

On note $\overline{R}(b)$ et $\overline{V}(b)$ la moyenne pour chaque bloc b des couleurs normalisées de chaque pixel du bloc b. Pour chaque bloc b, on définit un indice caractéristique de dimension 5, noté IC, qui inclut des informations de texture et de couleur :

$$IC(b) = [\overline{Err_1}(b), \overline{Err_2}(b), \overline{Err_3}(b), \overline{R}(b), \overline{V}(b)]^T.$$

Complétez le script `TroisiemeEtape.m` en réalisant les étapes suivantes :

- Construisez pour chaque bloc et chaque texture ce nouvel ensemble de données IC.
- Mettez en œuvre un mécanisme de classification bayésienne supervisée de chaque bloc b de l'image selon cette caractéristique multidimensionnelle. On continuera de superviser la segmentation grâce aux choix interactifs des sous-régions notées SR_i dans ce qui précède. Ces choix initiaux permettront de modéliser les distributions (supposées gaussiennes) des caractéristiques pour chacune des 3 textures.
- Définissez un taux d'erreur ou de reconnaissance pour le classifieur bayésien et comparer les résultats avec ceux établis grâce à la texture seule (sans la couleur).
- Evaluez votre algorithme de segmentation supervisée (avec k et w bien choisis) sur une image naturelle. L'image "`plage.jpg`" suivante (figure 5) présente 3 (voire 4) régions (rochers, eau, végétations, plage) dont les textures et les couleurs permettent d'espérer une segmentation correcte.
  La fonction `imread` permet de lire une image `.jpeg`.

FIGURE 5 – Une image naturelle "Plage Piana" utilisable pour tester votre algorithme