TP1 – Corrélation entre pixels dans une image

Initiation à Matlab

Notions générales

- Le résultat d'une affectation = est affiché, sauf si cette affectation se termine par un caractère ;
- Les commandes `format short` et `format long` permettent de modifier le format d'affichage des variables.
- Les fonctions `who` et `whos` permettent d'afficher l'ensemble des variables utilisées.
- La fonction `clear` efface le contenu de toutes les variables utilisées.
- Il est fortement déconseillé d'utiliser des mots-clés de Matlab comme noms de variables.
- La commande `help <fonction>` affiche la description de `<fonction>`.

Manipulation de vecteurs et de matrices

- Les composantes d'un vecteur ligne sont séparées par des virgules ou des espaces : `v1 = [x1 y1 z1];`
- Les composantes d'un vecteur colonne sont séparées par des points-virgules : `v2 = [x2 ; y2 ; z2];`
- Vecteur à incrément constant : `v3 = x_min:dx:x_max;` (vecteur ligne de dimension variable, qui contient les valeurs `x_min+i*dx`, où `i` est un entier positif ou nul tel que `x_min+i*dx` est inférieur à `x_max`).
- Les matrices utilisent la même syntaxe que les vecteurs : `M = [m11 m12 m13 ; m21 m22 m23];`
- La sous-matrice de `M` constituée par les lignes de numéros pairs et les colonnes de numéros impairs s'écrit : `N = M(2:2:end,1:2:end);`
- Vectorisation d'une matrice (les colonnes de `M` sont concaténées) : `v = M(:);`
- L'instruction `[nb_lignes,nb_colonnes] = size(M);` permet d'affecter le nombre de lignes de la matrice `M` à la variable `nb_lignes`, et le nombre de colonnes de `M` à la variable `nb_colonnes`.

Quelques matrices utiles

- `zeros(m,n)` : matrice nulle de taille $m \times n$.
- `ones(m,n)` : matrice de taille $m \times n$ dont tous les éléments sont égaux à 1.
- `eye(m,n)` : matrice de taille $m \times n$ dont les éléments diagonaux sont égaux à 1, les autres à 0.
- `rand(m,n)` : matrice de taille $m \times n$ d'éléments tirés aléatoirement selon la **loi uniforme** sur $[0, 1]$.
- `randn(m,n)` : matrice de taille $m \times n$ d'éléments tirés aléatoirement selon la **loi normale** centrée réduite.
- Appeler ces fonctions avec un seul argument équivaut à les lancer avec deux arguments identiques.

Opérations sur les matrices

- Addition `A+B` ; soustraction `A-B` ; produit `A*A'` ; puissance `A^3` ; transposition `A'` ou `transpose(A)`.
- Inverse `inv(A)` ; pseudo-inverse `pinv(A)`.
- Multiplication **élément par élément** `A.*B` (chaque élément `A(i,j)` est multiplié par l'élément `B(i,j)`) ; division **élément par élément** `A./B` (chaque élément `A(i,j)` est divisé par l'élément `B(i,j)`) ; puissance **élément par élément** `A.^3` (chaque élément de `A` est élevé à la puissance 3).

Quelques conseils utiles

- La fonction `mean` permet de calculer la moyenne des éléments d'une matrice, colonne par colonne. Dans le cas où la matrice comporte une seule ligne, `mean` calcule la moyenne de la ligne.
- La fonction `figure` permet de créer et de configurer la fenêtre de visualisation, alors que la fonction `imagesc` permet d'afficher une image dans la figure. N'oubliez donc pas de modifier les titres des figures, afin de rendre ces titres cohérents (ce conseil vaut également pour les noms des axes d'un graphique).
- Sachant que l'incrément par défaut est égal à `1`, il est plus lisible d'écrire `v = 1:10;` que `v = 1:1:10;`
- De même, il est plus lisible d'écrire `M(:,:,1)` que `M(1:end,1:end,1)`.

Exercice d'initiation à Matlab

Lancez l'exécution du script `exercice_Matlab` : ce script lit une image en niveau de gris, la stocke dans la matrice bidimensionnelle `image_originale` et l'affiche. Les niveaux de gris des pixels de cette image sont obtenus après traversée d'une mosaïque de filtres colorés, appelée *matrice de Bayer*, qui est placée devant le récepteur photosensible des appareils photographiques :

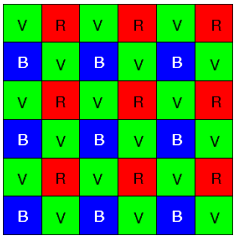


FIGURE 1 – Mosaïque de filtres colorés en rouge, vert et bleu (matrice de Bayer).

Vous constatez également qu'un message d'erreur s'affiche dans la fenêtre d'exécution. Cela vient du fait que le script `exercice_Matlab` appelle la fonction `ecriture_RVB`, qui n'existe pas encore, mais que vous devez écrire dans un fichier de nom `ecriture_RVB.m`. L'en-tête de cette fonction s'écrit :

```
function image_RVB = ecriture_RVB(image_originale)
```

Elle doit créer une matrice `image_RVB` à 3 dimensions contenant deux fois moins de lignes et deux fois moins de colonnes que `image_originale`. Chaque pixel de `image_RVB` correspond à un ensemble de quatre pixels de `image_originale` : V_1, R, V_2, B. Les valeurs R et B sont recopiées telles quelles dans les canaux rouge et bleu de `image_RVB`. En revanche, la valeur dans le canal vert est égale à la moyenne des valeurs V_1 et V_2.

Exercice 1 : mise en évidence des corrélations entre pixels voisins

Cet exercice constitue une illustration du cours de probabilités consacré à un couple de variables aléatoires. Ici, chaque pixel d'une image numérique est considéré comme une variable aléatoire. On s'intéresse à la corrélation entre pixels voisins.

Le script Matlab de nom `exercice_1` affiche une image interne à Matlab (cf. figure 2-a) ainsi que son histogramme (cf. figure 2-b). Il doit aussi afficher les paires de niveaux de gris d'un pixel et de son voisin de droite sous la forme d'un nuage de points. Écrivez la fonction `vectorisation` appelée par ce script, d'en-tête :

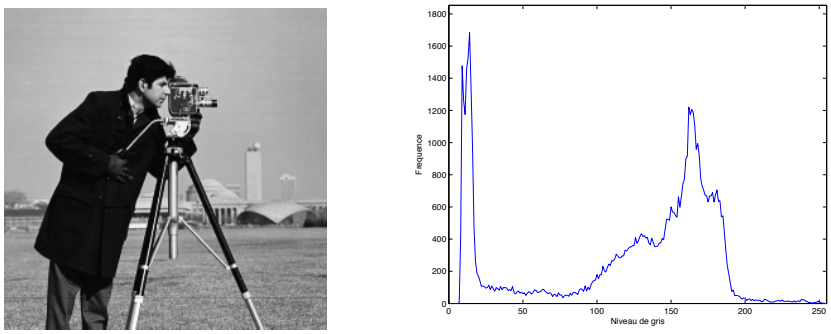
```
function [I_gauche,I_droite] = vectorisation(I)
```

dont les deux paramètres de sortie `I_gauche` et `I_droite` doivent être deux sous-matrices de `I` vectorisées, c'est-à-dire deux vecteurs colonnes.

Écrivez ensuite la fonction `calcul_parametres`, qui calcule le coefficient de corrélation linéaire r des données, ainsi que les deux paramètres (a, b) de la droite de régression d'équation $y = ax + b$.

Rappels

- Moyenne : $\bar{x} = \dfrac{1}{n}\sum_{i=1}^{n} x_i$
- Variance : $\sigma_x^2 = \dfrac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2 = \dfrac{1}{n}\sum_{i=1}^{n} x_i^2 - \bar{x}^2$
- Écart-type : $\sigma_x = \sqrt{\sigma_x^2}$
- Covariance : $\sigma_{xy} = \dfrac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y}) = \dfrac{1}{n}\sum_{i=1}^{n} x_i\, y_i - \bar{x}\,\bar{y}$
- Coefficient de corrélation linéaire : $r = \dfrac{\sigma_{xy}}{\sigma_x\,\sigma_y}$
- Équation de la droite de régression : $y = \dfrac{\sigma_{xy}}{\sigma_x^2}\,(x - \bar{x}) + \bar{y}$



(a) (b)

FIGURE 2 – (a) Exemple d'image interne à Matlab. (b) Histogramme de l'image (a).

Attention

- Cet exercice doit être résolu sans boucle !
- L'écriture `I(1:nb_lignes-1,:)` revient à calculer `I_haut`, et non pas `I_gauche`.

Exercice 2 : décorrélation des niveaux de gris d'une image

La décorrélation des niveaux de gris consiste, par exemple, à soustraire au niveau de gris d'un pixel le niveau de gris de son voisin de gauche. Faites une copie du script `exercice_1`, de nom `exercice_2`, que vous modifierez de manière à remplacer l'image de la figure 2-a par une version décorrélée `I_decorrelee` de cette image. Pour cela, initialisez `I_decorrelee` par duplication de `I`, conservez la première colonne et modifiez les autres colonnes de cette matrice.

Attention

- Il est nécessaire de conserver la première colonne de l'image d'origine dans `I_decorrelee`, sans quoi l'opération de décorrélation ne serait pas inversible (il serait impossible de recalculer l'image d'origine).
- Il est nécessaire de modifier la valeur de la variable `I_min`, qui n'est plus égale à 0 puisque `I_decorrelee` peut comporter des valeurs négatives.