MPSI - ITC - TP 21 - 02/03/2026

Images matricielles

Notions d'image matricielle et de pixel

Une façon d'encoder les images est de les voir comme des matrices rectangulaires M dans lesquelles chaque élément $M_{i,j}$ est un carré de couleur unie appelé *pixel*.
Ici, pour simplifier, les pixels seront des nombres entre 0 et 255 où chaque valeur représente un niveau de gris différent (0 étant noir et 255 étant blanc).

Code fourni

Vous est fourni du code avec deux fonctions :

- Une fonction `fichier_vers_matrice` qui prend en argument le nom d'un fichier contenant une image, et renvoie la matrice de niveaux de gris correspondante.
- Une fonction `matrice_vers_fichier` qui prend en entrée une matrice de niveaux de gris et un nom de fichier, et enregistre l'image correspondante dans le fichier correspondant.

Sauf mention contraire explicite, dans toutes les questions de ce TP, quand on dit qu'une fonction prend une image en entrée on entend par là une matrice de niveaux de gris (une liste de listes de nombres entre 0 et 255).
De même, quand on dit qu'une fonction renvoie une image en sortie, on entend par là une matrice de niveaux de gris.

Exercice 1 (Transformations géométriques)

1. Écrire une fonction `rotation_antihoraire` qui prend une image en entrée et renvoie en sortie la même image tournée de 90° en sens anti-horaire. Utiliser votre fonction (et les fonctions fournies) pour créer un fichier « `question_1_corrige.jpg` » à partir du fichier « `question_1.jpg` ». Bonus : modifier votre fonction pour que l'angle soit un deuxième argument de la fonction, et ne soit pas forcément multiple de 90°.
2. Écrire une fonction `transposee` qui transpose une image (au sens matriciel du terme « transposer »). Utiliser votre fonction (et les fonctions fournies) pour créer un fichier « `question_2_corrige.jpg` » à partir du fichier « `question_2.jpg` ».
3. Écrire une fonction `mirroir` qui applique une symétrie axiale par rapport à la verticale au milieu de l'image. Utiliser votre fonction (et les fonctions fournies) pour créer un fichier « `question_3_corrige.jpg` » à partir du fichier « `question_3.jpg` ».

Exercice 2 (Transformations colorimétriques)

1. Écrire une fonction `negatif` qui prend en entrée une image et renvoie en sortie la même image où chaque niveau de gris x a été remplacé par son opposé $255 - x$. Utiliser votre fonction (et les fonctions fournies) pour créer un fichier « `question_4_corrige.jpg` » à partir du fichier « `question_4.jpg` ».
2. *(Question difficile)* Écrire une fonction `posterisation` qui prend en arguments une image et un nombre cible de couleurs $k \in \mathbb{N}$, et qui renvoie une image qui n'utilise que k niveaux de gris différents, et qui ressemble le plus possible à l'image de départ.

Exercice 3 (Autres transformations)

1. Écrire une fonction `cadre_interieur` qui prend en entrée une image et un nombre k, et qui renvoie en sortie la même image mais avec un cadre noir de k pixels d'épaisseur sur tout le contour de l'image. Les nouveaux pixels noirs écraseront des pixels de l'image d'origine, la taille de l'image ne changera pas.
2. Écrire une fonction `cadre_exterieur` qui prend en entrée une image et un nombre k, et qui rajoute un cadre noir de k pixels d'épaisseur tout autour de l'image. Les nouveaux pixels n'ont pas le droit d'écraser des pixels de l'image d'origine.
3. Écrire une fonction `pixeliser` qui prend en arguments une image de taille $n \times m$ et un entier $r \in \mathbb{N}^*$ et renvoie une image de taille $\left\lfloor \frac{n}{r} \right\rfloor \times \left\lfloor \frac{m}{r} \right\rfloor$ où chaque groupe de $r \times r$ pixels a été remplacé par un seul pixel (la moyenne des pixels remplacés).

Exercice 4 (Analyse)

En utilisant matplotlib, tracer une courbe qui indique, pour chaque niveau de gris, combien de pixels ont ce niveau de gris.