Transformations géométriques d'images numériques

Images matricielles avec `python`;

L'objectif de ce premier sprint est de découvrir le laboratoire d'expérimentation numérique `python`/`numpy`/`matplotlib` avec l'IDE spyder, en particulier en ce qui concerne son utilisation pour la manipulation des images matricieles. Le *Guide de démarrage rapide* est le document de référence à étudier pour remplir les exigences de la section **prise en main**.

1 Prise en main

À l'issue de la première semaine de travail, chaque membre du groupe doit

1. être familier avec spyder : la console, l'explorateur de variables, l'historique, les fenêtres graphiques, …
2. Maîtriser l'utilisation interactive de base dans la console : création de variables, affectation, affichage, opérations élémentaires sur les flottants, …
3. Comprendre la notion de tableau `numpy` et leur manipulation : opérations de base, extrations de lignes, de colonnes, de sous–tableaux, leur concaténation selon une dimension, les opérations éléments par éléments, les tableaux particuliers usuels, …
4. Avoir utilisé les fonctions graphiques élémentaires : la fonction `plot`, la sauvegarde, les retouches de base (couleur, style), les fonctions `imread`, `imshow` et `imsave`, …
5. Savoir utiliser efficacement l'éditeur intégré pour l'exécution, la sauvegarde et le partage de code.

2 Images matricielles

Une fois acquises, les connaissances de la section précédente vont être utilisées pour la lecture/écriture/visualisation d'images matricielles. Les compétences à acquérir sont :

1. comprendre la représentation d'une image sous forme matricielle ;
2. Connaître les fonctions `python` permettant de convertir des images en tableaux et vice–versa. Extraire la dimension d'une image, accèder aux plans (rouge, vert, bleu), accéder à la couleur d'un pixel, …

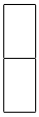
3 Challenges

Les tâches ci–dessous ne doivent pas forcément être toutes réalisées. Pour chacune d'elles, le but n'est pas seulement d'obtenir le résultat demandé, mais plutôt de mettre en œuvre l'approche la plus élégante possible. Le produit matriciel doit être privilégié dès lors qu'on peut l'utiliser. L'usage de boucles et de tests doit être extrêment limité afin d'utiliser au maximum la capacité de `numpy` pour la vectorisation des calculs (manipulation de tableaux). L'algorithme qui produit chaque image doit être clairement détaillé et ses qualités et limites commentées.

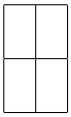
1. Une image d'une hauteur de 50 pixels et d'une largeur de 100 pixels contenant 10 bandes horizontales alternativement blanches et noires.

— À partir de la matrice initiale, une image de hauteur 100 pixels et de largeur 50 pixels ayant 10 bandes verticales alternativement blanches et noires.

2. Une diagonale noire d'épaisseur 1 pixel sur une image de 50 pixels sur 50 pixels. La diagonale commence en haut à gauche.
3. Une image de 60x256 pixels permettant de visualiser tous les niveaux de gris : la première colonne de pixels devra être blanche et la dernière noire.
4. Une image carrée de 80 pixels de côté, représentant un échiquier de 64 cases.
5. À partir d'une image carrée de 100 pixels de côté, produire l'algorithme et le code `python` correspondant pour générer une autre image de type photomaton :
   — deux images identiques superposées

   — quatres images agencées en carré :

6. Une image représentant le drapeau donné en respectant les proportions.

   (a) Le drapeau polonais (5:8)

   (b) Le drapeau français (2:3)

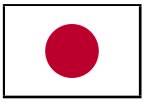
   (c) Le drapeau malgache (2:3)

   (d) Le drapeau tchèque (2:3)

   (e) Le drapeaux japonais (2:3)

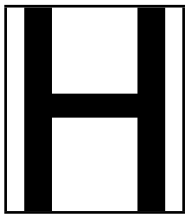
   (f) Le drapeau seychellois (1:2)

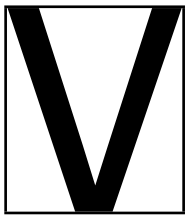
   

   (g) Le drapeau dominicain (sans le motif central) (2:3)

7. Une image en noir et blanc de chacune des lettres majuscules suivantes, selon la police **consolas**

(a) la lettre 

(b) la lettre 

(c) la lettre