Informatique PCSI
TP 7 suite : manipulation d'une image

Énoncé des exercices

Exercice 1

Écrire une fonction `echange_lignes` qui prend en paramètre une matrice représentant une image carrée de côtés n pixels, avec n pair, au format pbm. La fonction modifie la matrice en plusieurs étapes.

Première étape : la première ligne (indice 0) et la dernière (indice $n-1$) sont fixes. Pour les autres lignes, dans chaque couple de lignes consécutives les deux lignes sont échangées, la ligne d'indice 1 avec celle d'indice 2, la ligne d'indice 3 avec celle d'indice 4, et ainsi de suite jusqu'à l'échange de la ligne d'indice $n-3$ avec celle d'indice $n-2$.

Deuxième étape : le même processus est reproduit sauf que les deux premières lignes et les deux dernières sont fixes.

On continue le processus d'échanges, en laissant fixes les trois premières et les trois dernières lignes, puis les quatre premières et les quatre dernières, etc. À la dernière étape, seule la ligne d'indice $n/2-1$ est échangée avec la ligne suivante.

Tester la fonction avec une matrice représentant une image. Exécuter à nouveau la fonction sur l'image obtenue et recommencer plusieurs fois en observant le résultat.

Exercice 2

Transformation du photomaton

On considère une image carrée de côté n pixels. La transformation à appliquer pour créer une nouvelle image est décrite pour chaque pixel repéré par un couple d'indices (i, j) (les indices vont de 0 à $n-1$) :

- si i et j sont pairs, le pixel est envoyé sur le pixel d'indice $(i/2, j/2)$;
- si i est impair et j pair, le pixel est envoyé sur le pixel d'indice $((n+i)/2, j/2)$;
- si i est pair j impair, le pixel est envoyé sur le pixel d'indice $(i/2, (n+j)/2)$;
- si i et j sont impairs, le pixel est envoyé sur le pixel d'indice $((n+i)/2, (n+j)/2)$.

Écrire une fonction récursive `photomaton` qui prend en paramètres une matrice carrée et un entier correspondant au nombre de répétitions de cette transformation. Tester la fonction avec neuf répétitions par exemple.

Exercice 3

Chiffrement d'une image bitmap. (Utiliser la matrice définissant l'image `chat.pbm`).

1. Écrire une fonction `hasard` qui prend en paramètres les dimensions d'une image `n` et `p`, (`n` est la hauteur et `p` la largeur), et renvoie une matrice représentant une image aléatoire. Chaque pixel vaut 0 ou 1 de manière équiprobable.

   Créer une image aléatoire avec les mêmes dimensions que l'image `chat.pbm`.

   On nomme cette image `alea.pbm`. Visualiser cette image.
2. Écrire une fonction `chiffre` pour chiffrer l'image `chat.pbm`. La fonction prend en paramètres une image à chiffrer et une image clé sous la forme de matrices de mêmes dimensions. Elle crée une nouvelle image de même dimension. Pour déterminer chaque bit de l'image, on utilise l'opérateur *ou exclusif* entre les bits correspondant de l'image à chiffrer et de l'image clé. Le *ou exclusif* se note `^` en Python. Si `b1` et `b2` sont deux bits, `b1 ^ b2` vaut 0 si `b1` et `b2` sont égaux et vaut 0 sinon.

   Exécuter la fonction avec l'image `chat.pbm` et l'image clé `alea.pbm`. Visualiser le résultat.

   L'image clé s'appelle un *masque*. Comment déchiffrer l'image chiffrée ?
3. Faire une copie de la fonction précédente, la renommer, et remplacer l'opérateur *ou exclusif* par l'opérateur *ou*. L'opérateur se note $|$, mais on peut utiliser `or`. Que peut-on dire de l'image obtenue si on exécute ce programme avec l'image chiffrée et l'image clé ?