INFO — IUT GRAND OUEST NORMANDIE

R 5.A.06

2023 – 2024

Sensibilisation à la programmation multimédia

TP n° 2
Images

ANNE Jean-François
D'après le TD de F. DOIDY

Sensibilisation à la programmation multimédia

Le but de ce TP est de se familiariser avec la programmation Python pour les images.

A. Echantillonnage et quantification d'une image

Dans ce TP nous allons analyser l'impact d'une diminution de la quantification ou de l'échantillonnage d'une image.
Nous verrons ensuite comment suréchantillonner une image.

1°) Exercice 1 : Compression par diminution du nombre de bits

Ecrire un programme python qui prend une image (Lena.png) qui affiche le nombre de bits dans lequel est codée l'image et qui diminue le nombre de bits de chaque pixel pour compresser l'image. Par exemple passer de 8 bits à 4 bits

- Afficher l'une à côté de l'autre l'image originale et l'image compressée.
- Afficher la taille en octets de l'image originale, de l'image compressée et le taux de compression.

2°) Exercice 2 : Compression par diminution du nombre de lignes

Ecrire un programme python qui prend une image qui réduit la taille de l'image en ne prenant qu'une ligne sur 2

- Afficher l'une à côté de l'autre l'image originale et l'image compressée.
- Afficher la taille en octets de l'image originale, de l'image compressée et le taux de compression.

3°) Exercice 3 : Compression par diminution du nombre de colonnes

Ecrire un programme python qui prend une image qui réduit la taille de l'image en ne prenant qu'une colonne sur 2

- Afficher l'une à côté de l'autre l'image originale et l'image compressée.
- Afficher la taille en octets de l'image originale, de l'image compressée et le taux de compression.

4°) Exercice 4 : Compression par diminution du nombre de lignes et de colonnes

Ecrire un programme python qui prend une image qui réduit la taille de l'image en ne prenant qu'une ligne sur 2 et qu'une colonne sur 2

- Afficher l'une à côté de l'autre l'image originale et l'image compressée.
- Afficher la taille en octets de l'image originale, de l'image compressée et le taux de compression.

5°) Exercice 5 : Duplication des lignes et des colonnes pour retrouver l'image d'origine

Ecrire un programme python qui prend une image qui réduit la taille de l'image en ne prenant qu'une ligne sur 2 et qu'une colonne sur 2, puis dupliquant chaque ligne et chaque colonne, pour retrouver une image de la taille égale à celle de l'image originale.

- Afficher l'une à côté de l'autre l'image originale et l'image compressée et l'image décompressée. Mettre un titre à chaque image.

- Afficher la taille en octets de l'image originale, de l'image compressée, de l'image reconstituée et le taux de compression.

6°) Exercice 6 : Duplication de la moyenne des pixels des lignes et des colonnes pour retrouver l'image d'origine

Ecrire un programme python qui prend une image qui réduit la taille de l'image en ne prenant qu'une ligne sur 2 et qu'une colonne sur 2, puis en créant les lignes et les colonnes correspondant à la moyenne de 2 lignes et des colonnes contiguës, pour retrouver une image de la taille égale à celle de l'image originale.

- Afficher l'une à côté de l'autre l'image originale et l'image compressée et l'image décompressée. Mettre un titre à chaque image.
- Afficher la taille en octets de l'image originale, de l'image compressée, de l'image reconstituée et le taux de compression.

L'image finale sera de taille 511 x 511

7°) Exercice 7 : Lisser une image

Ecrire un programme python qui prend une image et qui lisse cette image en prenant une matrice 3x3 pour le lissage.

- Afficher l'une à côté de l'autre l'image originale et l'image lissée. Mettre un titre à chaque image.
- Afficher la taille en octets de l'image originale, de l'image lissée, et le taux de compression.

B. Pour aller plus loin :

1°) Détections de voitures :

Les images images1png et image2.png représentent deux photographies prises par une caméra située sur l'autoroute Paris-Lyon. Ces photographies ont été prises à quelques minutes d'intervalle. On va chercher à faire ressortir l'affichage des voitures.

Méthode :

❖ Créer une image Somme contenant l'ensemble des véhicules présents sur l'image1.png et sur l'image2.png en faisant la somme des 2 images.

❖ Créer une image Diff1 qui est la soustraction de l'image 1 et l'image 2.

❖ Effectuer l'opération Diff1 = Diff1 * (Diff1 si >40)

Augmenter la valeur 40 du test si la différence est peu visible.

❖ Créer une image Diff2 qui est la soustraction de l'image 2 et l'image 1.

❖ Effectuer l'opération Diff2 = Diff2 * (Diff2 si >40)

Augmenter la valeur 40 du test si la différence est peu visible.

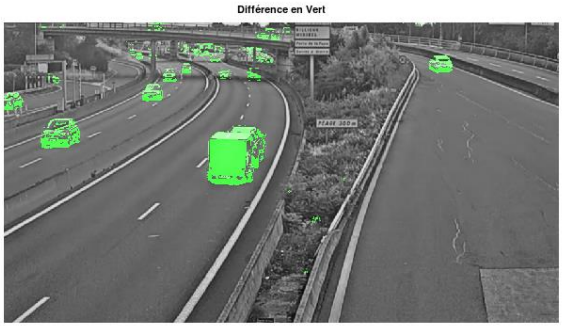
❖ Ajouter les deux images Diff1 et Diff2

❖ Créer une copie de l'image Somme dans laquelle vous modifierez le canal vert en ajoutant 255* diff aux valeurs de somme.

❖ Vous obtenez alors une image dans laquelle les véhicules en circulation seront mis en vert.

❖ Afficher, sur une même figure côte à côte :

- Les 2 images originales
- En dessous l'image contenant tous les véhicules et l'image contenant les véhicules en circulation qui sont en vert.



C. Compression d'image :

1°) Compression et décompression « sans perte » :

a) Compression :

Le « Run Length Encoding » (RLE) est le plus simple des algorithmes de compression de données. Il remplace les successions de deux ou plusieurs caractères identiques par un nombre représentant la longueur du passage, suivi du caractère d'origine.

Par simplification, dans ce TP, vous coderez de la même manière le fait que le caractère soit isolé ou qu'il existe une succession de caractères identiques.

Algorithme de compression :

- Recherche des caractères répétés plus de n fois (vous prendrez n=1).
- Remplacement de l'itération de caractères par :

1.un caractère spécial identifiant une compression. Exemple : @.

2.le nombre de fois où le caractère est répété.

3.le caractère répété

Exemple : AAAAARRRRRROLLLLBBBBBUUTTTTTT.

On choisit comme caractère spécial : @

et comme seuil de répétition : 3.

Après compression : @5A@6RO@4L@5BUU@6T

gain : 11 caractères soit 38%

Il faudrait utiliser cet algorithme sur une image n'ayant pas trop de changement de gris ! (image ayant des bandes de couleurs horizontales de part et d'autre et en format PNG sans compression ou en BMP, et en codant sur 8 bits).

b) Décompression :

Algorithme de décompression :

Durant la lecture du fichier compressé, lorsque le caractère spécial est reconnu, on effectue l'opération inverse de la compression tout en supprimant ce caractère spécial.

Exemple : @5A@6RO@4L@5BUU@6T

On choisit comme caractère spécial : @ et comme seuil de répétition : 3.

Après décompression : AAAAARRRRRROLLLLBBBBBUUTTTTTT

gain : 11 caractères soit 38%

c) Compression d'une image

Appliquer l'algorithme de compression précédent à une image BMP.

d) Décompression d'une image

Appliquer l'algorithme de décompression précédent pour réobtenir l'image BMP.

D. Webographie :

- http://nsinfo.yo.fr/cours_python/python_10_images.html
- http://tvaira.free.fr/dev/python/python-images.html
- https://www.cours-gratuit.com/tutoriel-python/tutoriel-python-les-bases-de-traitement-dimages-en-python-bibliothque-numpy
- https://www.cours-gratuit.com/tutoriel-python/tutoriel-python-les-bases-de-traitement-dimages-avec-scipy
- https://www.cours-gratuit.com/tutoriel-python/tutoriel-python-les-bases-de-traitement-dimages-avec-scikit-image
- https://www.cours-gratuit.com/tutoriel-python/tutoriel-python-les-bases-de-traitement-dimages-en-python-opencv
- https://ensip.gitlab.io/pages-info/ressources/transverse/tuto_images.html
- https://www.codingame.com/playgrounds/53303/apprendre-python-dans-le-secondaire/manipulations-dimages-i
- https://compression.fiches-horaires.net/la-compression-avec-perte-1/le-compression-jpeg/
- https://scikit-image.org/skimage-tutorials/lectures/three_dimensional_image_processing.html
- http://mp01.free.fr/comp/comp.htm
- https://web.maths.unsw.edu.au/~lafaye/CCM/video/compimg.htm