

RAPPORT DE PROJET

COMPRESSION D’IMAGES PAR TRANSFORMEE DE FOURIER

*Projet encadré par M. Boulbe et M. Auroux.*

*Janvier 2020*

DENIS HANSI

MALTESE SALOME

TRAORE YACOUBA

MANDINE LOIC

Introduction

Dans le cadre de notre projet, nous avons étudié le sujet « La compression

d’images par transformée de Fourier ». Afin de le traiter au mieux, nous avons utilisé des outils mathématiques, notamment la Transformée de Fourier discrète mais aussi des outils informatiques, avec l’exploitation du logiciel Python. L’objectif principal est de compresser une image et pour cela, nous procédons en différentes étapes :

Premièrement, nous initialisons l’image en un signal sur lequel on peut appliquer la DCT-II : on convertit en une matrice de triplets (correspondant à la représentation RGB), on dimensionne la matrice en multiples de 8, enfin on recentre les valeurs. Dès lors que l’image correspond à un signal bi-dimensionnel périodique, paire nous allons lui appliquer la transformée de cosinus discrète sur chaque canal d’information de l’image (et ce sur chaque bloc 8x8). On précise que nous utilisons la Transformée de cosinus discrète n°2. Ainsi, nous obtenons une matrice où chaque coefficient traduit la fréquence des changements d’intensité lumineuse (contour=haute fréquence, zone uniforme=basse fréquence). Ensuite, nous quantifions la matrice obtenue précédemment, selon la convention de la norme de compression JPEG ; mathématiquement cela correspond à diviser par une matrice qui va garder que les modes (fréquences) les plus importants et en prendre la partie entière. Finalement, la matrice obtenue sera les données de notre image compressée.

Par la suite, nous expliquerons en détail l’ensemble des différentes étapes pour compresser une image, mais aussi pour la décompresser et comparer l’image source et compressée.

SOMMAIRE

Introduction………………………………………...2

Etape 0 : Importation des modules sur Python…….4

Etape 1 : Préparation de l’image……………….…….5

Etape 2 : DCT-II de l’image…………………………7

Etape 3 : Quantification de l’image………………...9

Etape 4 : Décompression de l’image………………11

Conclusion…………………………………………13

Remerciements……………………………………14

Bibliographie.…………………………………..…..15

Carnet de bord..…………………… ……………….16

Annexe……………………….………………....……17

Etape 0 : Importation des modules sur Python.

Souvent abordée comme étant une tache simple, son apparence est extrêmement trompeuse ! Nous avons consacré une journée entière afin d’importer l’ensemble des modules nécessaires pour le fonctionnement du code. De plus, selon l’environnement de développement intégré (IDE) utilisé, la procédure est différente. En l’occurrence, nous avons importé les librairies *PIL* pour les traitements d’image, *numpy* pour manipuler les tableaux en stockant et en effectuant des opérations sur des données, *linalg* pour avoir accès aux normes et enfin le module *math* pour utiliser les fonctions mathématiques.

Désormais équipés des modules nécessaires, nous pouvons nous lancer dans l’écriture du code.

Etape 1 : Préparation de l’image

**A/ Théorie**

Toute fonction périodique peut se décomposer dans une base de fonctions cosinus et sinus, et si elle est paire seulement avec des fonctions cosinus.

Tout d’abord, nous allons convertir l’image en données numérique : matrice de triplets correspondants aux valeurs RGB.

Ensuite, par symétrisation et périodisation des motifs de l’image, on la rend périodique et paire en deux directions.

Ainsi, nous pouvons décomposer notre image comme une combinaison linéaire de fonctions cosinus, en x et en y.

**B/ Pratique**

On souhaite préparer l’image source telle qu’on puisse utiliser la transformée de Fourier pour étudier et compresser l’information contenue dans l’image.

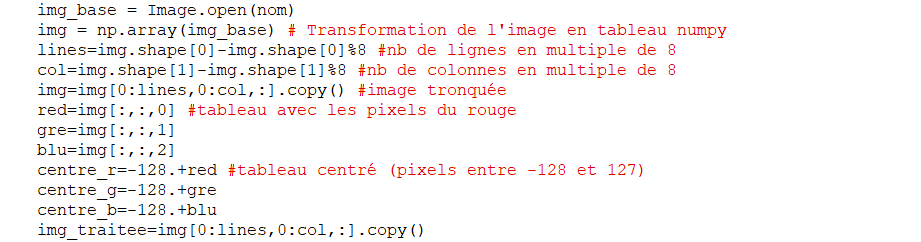
Tout d’abord, nous stockons les valeurs des informations de l’image via la fonction open(image\_source) du module Image de PIL, et donc nous obtenons le tableau des valeurs RGB de chaque pixel de l’image.

Ensuite, on dimensionne notre matrice de données associée à notre image source telle que son nombre de lignes et de colonnes soient des multiples de 8.

Puis, on extrait et on stocke les composantes rouges, vertes et bleues de notre images (respectivement la première composante des triplets, etc…).

Enfin, on soustrait 128 aux intensités de chaque pixel, aﬁn de traiter des entiers compris entre −128 et 127.

Ainsi, notre image s’apparente à un signal bi-dimensionnel paire, périodique et centré en zéro ; est donc prête à être traité avec la DCT.



Etape 2: DCT-II de l’image

**A/ Théorie**

Nous utilisons l’expression de la transformée de cosinus discrète n°2. Sous forme matricielle, cette transformée nous permet d’obtenir une matrice orthogonale et correspond, d’un point de vue algébrique, à un changement de base de notre matrice associée à notre image source en une matrice « en mode fréquentiel ».

On rappelle aussi qu’on effectue la transformée sur des blocs 8x8.

La matrice P de changement de base associée à cette transformée est donc une matrice de coefficients constants qui sont principalement des cosinus correspondants les coefficients de la DCT.

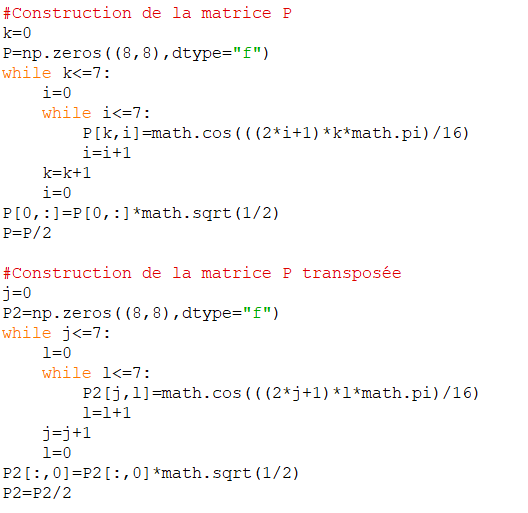
**B/ Pratique**

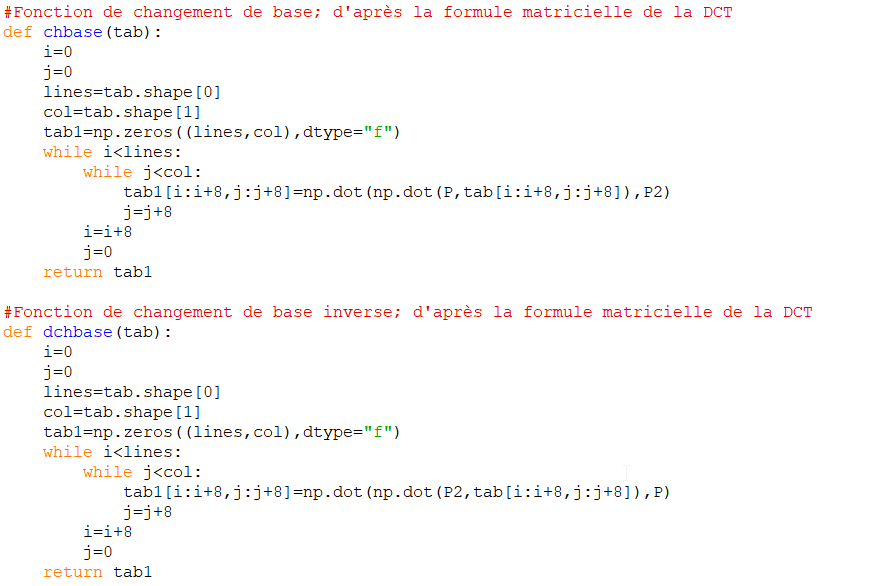
Le premier enjeu est de déterminer la matrice P. Nous partons d’une part de la formule matricielle que nous identifions d’autre part avec la formule analytique de la DCT. Cette étape fut particulièrement compliquée !!

Ainsi, nous calculons, à l’aide de deux boucles, les matrices P et transposée de P.

Dès lors, nous pouvons appliquer le changement de base pour chaque bloc de 8x8 de notre matrice associée à notre image source.

La fonction np.dot (matrice1, matrice2) permet la multiplication matricielle entre deux matrices.





Etape 3 : Quantification de l’image

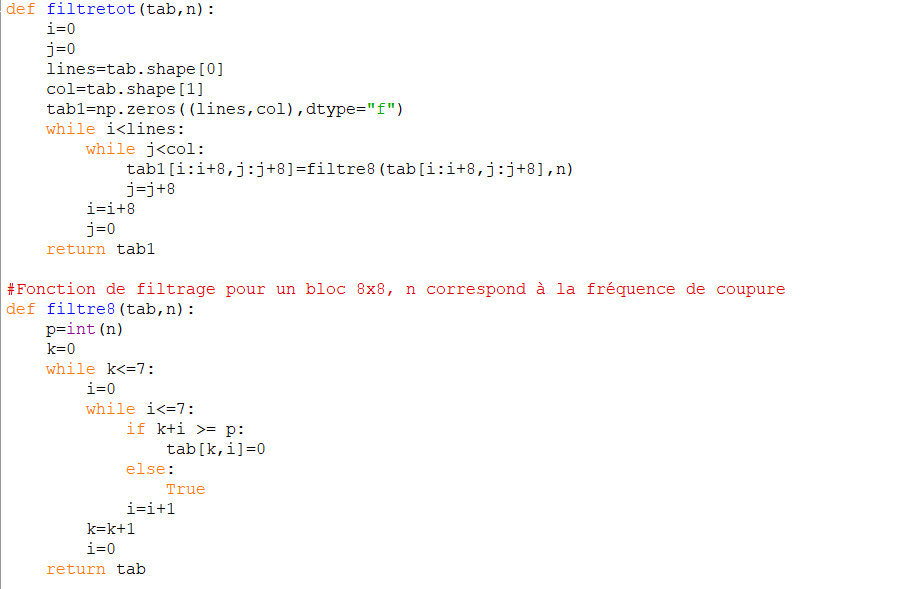
Le but de cette étape est de garder les modes les plus importants. Donc, nous allons diviser chaque bloc de 8 de notre matrice précédente issue de notre transformée par une matrice 8x8 de Quantification.

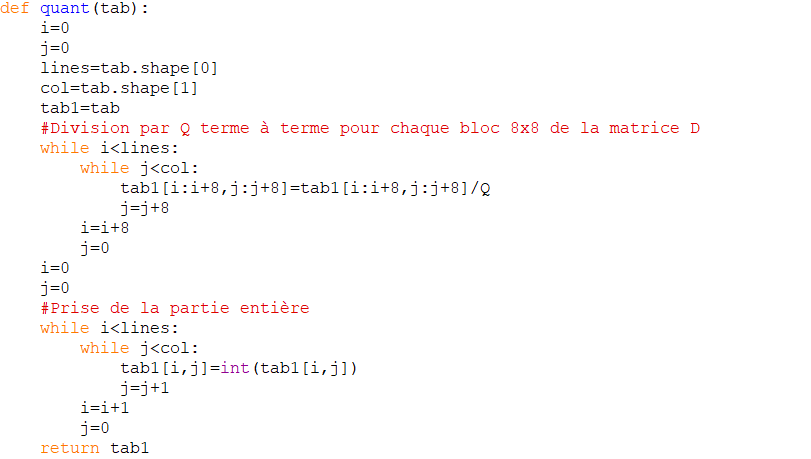
Cette dernière est déterminée pour la norme JPEG. Elle est construite telle que la division va annuler les termes de hautes fréquences et garder les basses fréquences ; car l’essentiel de l’information se trouve dans les coefficients de basses fréquences.

De plus, une étape facultative est de filtrer les hautes fréquences en tronquant à une fréquence de coupure.

En pratique, on va diviser terme à terme la matrice fréquentielle par la matrice de quantification et prendre la partie entière de chaque coefficient.

Et la procédure pour tronquer les valeurs fréquentielles reste aussi assez simple.





Etape 4 : Décompression de l’image

Pour chaque bloc de 8x8 de la matrice compressée, on va multiplier terme à terme par la matrice de quantification (que si lors de la compression on a divisé par la matrice de quantification).

Ensuite, nous allons effectuer le changement de base inverse pour passer de la matrice fréquentielle à la matrice des intensités. Ce changement de base se traduit par la formule de changement de base inverse usuelle. On précise que nous devons garder la même matrice de changement de base que pour la compression.

Conclusion et limites

Lors de ce projet, nous avons répondu au sujet demandé : nous devons compresser l’information d’une image. Nous avons découvert et manipulé un langage nouveau pour nous Python et son calcul formel, mais également approfondi quelques connaissances théoriques en mathématiques liées à la Transformation de Fourier.

Ce travail a été très intéressant et bénéfique pour nous, il nous a permis d’atteindre un objectif tout en étant autonome. Grace à notre cohésion de groupe, nous sommes arrivés à un résultat correct tout en partant de zéro, en obtenant une qualité d’image assez correcte. Répartir ce travail au sein du groupe pendant une semaine a été un véritable challenge pour nous.

Quant aux limites, nous constatons que lorsqu’on essaye de compresser une image avec beaucoup de contours (discontinuité), le résultat du code n’est pas en adéquation avec le résultat attendu. De plus, le temps nécessaire pour l’exécution du code est relativement long (vingtaine de secondes) et dépend du format de l’image et de sa complexité texturée.

Remerciements

Nous souhaitons remercier :

. Monsieur Didier AUROUX : directeur du département Mathématiques Appliquées et Modélisation.

. Monsieur Cédric Boulbe : responsable des troisièmes années.

Bibliographie

- <http://sorciersdesalem.math.cnrs.fr/Vulgarisation/JPEG/jpeg-DCT.html>

- <https://fr.wikipedia.org/wiki/Transform%C3%A9e_en_cosinus_discr%C3%A8te>

- <http://www.tsi.telecom-paristech.fr/pages/enseignement/ressources/beti/test_JPEG/mti.html>

- <http://documents.irevues.inist.fr/bitstream/handle/2042/2003/nondispo.pdf?sequence=1>

- <https://lehollandaisvolant.net/science/jpg/>

Carnet de bord

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | Hansi  Salomé | |  |  |  |  | Yacouba |  |  |  |  |  |  | Loic |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | Lundi 22/01 |  | Découverte du sujet. | | | |  |  | Découverte du sujet. | | | | |  |  | Découverte du sujet. | | | |  |  |
|  |  |  |  |  |  |  |  |
|  | (Matinée) |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | |  |  | | |  | |  |  | | | | | |  |  | |  | | |  |
|  | |  |  | | |  | |  |  | | | | | |  |  | |  | | |  |
|  | Lundi 22/01 |  | Installation de | | | Python | |  | Installation de Python. | | | | |  |  | Installation | | de Python | |  |  |
|  | (Après-midi) |  | . | | |  |  |  | Travail sur le shell. | | | | |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | |  |  |  |  | | |  |  |  |  | | | |  |  |  | | | |  |
|  | Mardi 23/01 |  | Fin | de | l’installation, | | |  | Fin | de | l’installation, | | |  |  | Fin de | l’installation, | | |  |  |
|  | (Matinée) |  | avec |  | l’ajout | des | |  | avec |  | l’ajout | | des |  |  | avec | l’ajout | | des |  |  |
|  |  |  | modules. | | |  |  |  | modules. | | |  |  |  |  | modules. | |  |  |  |  |
|  | |  |  | | | |  |  |  | | | | | |  |  | | | | |  |
|  | |  |  | | | |  |  |  | | | | | |  |  | | | | |  |
|  | Mardi 23/01 |  | Recherches internet. | | | |  |  | Recherches internet. | | | | |  |  | Recherches internet. | | | |  |  |
|  | (Après-midi) |  | Se | familiariser | | | de |  | Se | familiariser | | | de |  |  | Apprendre | |  | à |  |  |
|  |  |  | nouveau | | au | langage | |  | nouveau | | au | langage | |  |  | programmer en Python. | | | |  |  |
|  |  |  | Python. | |  |  |  |  | Python. | |  |  |  |  |  | Questions | |  | aux |  |  |
|  |  |  | Questions | | | aux | |  | Questions | | |  | aux |  |  | professeurs. | |  |  |  |  |
|  |  |  | professeurs. | | |  |  |  | professeurs. | | |  |  |  |  |  |  |  |  |  |  |
|  | |  |  | | |  |  |  |  | | | |  |  |  |  | | |  |  |  |
|  | Mercredi 24/01 |  |  | | |  |  |  |  | | | |  |  |  |  | | |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  | (Matinée) |  |  | | |  |  |  | Recherches | | | | |  |  | Recherches | | | |  |  |
|  |  |  | Recherches internet. | | | |  |  | internet. | |  |  |  |  |  | internet. |  |  |  |  |  |
|  |  |  | Choix de l’image. | | |  |  |  | Choix de l’image. | | |  |  |  |  | Choix de l’image. | | |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | |  |  | | |  |  |  |  | | | | | |  |  | | | | |  |
|  | Mercredi 24/01 |  | Avancement | | | sur | le |  | Théorie du rapport. | | | | |  |  | Pratique du rapport. | | | |  |  |
|  | (Après-midi) |  | code étape 3 et 4. | | | |  |  | Travail | | sur | le | code |  |  | Travail | sur | le | code |  |  |
|  |  |  |  |  |  |  |  |  | étape 4. | |  |  |  |  |  | étape 4. |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | |  |  | | |  |  |  |  | | | | | |  |  | | | | |  |
|  | Jeudi 25/01 |  | Avancement | | | sur | le |  | Rédaction du rapport. | | | | |  |  | Rédaction du rapport. | | | |  |  |
|  | (Matinée) |  | code. |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | |  |  | | | |  |  |  | | | | | |  |  | | | | |  |
|  | |  |  | | | |  |  |  | | | | | |  |  | | | | |  |
|  | Jeudi 25/01 |  | Finalisation du code. | | | |  |  | Finalisation du code. | | | | |  |  | Finalisation du code. | | | |  |  |
|  | (Après-midi) |  | Essai sur des images | | | |  |  | Essai sur des images. | | | | |  |  | Essaie sur des images. | | | |  |  |
|  |  |  | Questions | | | aux | |  | Questions | | |  | aux |  |  | Questions | |  | aux |  |  |
|  |  |  | professeurs. | | |  |  |  | professeurs. | | |  |  |  |  | professeurs. | |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Annexe

Lors de notre projet, nous avons eu besoin de plusieurs modules python afin d’accéder à certaines de leurs fonctions, et nous avons eu besoin d’une journée entière pour bien comprendre comment les installer !

L’objectif de cette annexe est de faciliter l’installation et l’importation des modules python (seule étape pour exécuter le code, il suffira de le lancer normalement après). Les modules dont nous avons eu besoin sont :

* Math
* Numpy
* PIL
* Image
* Linalg
* Time
* Scipy

Pip est un système de gestion de paquets utilisé pour installer et gérer des librairies écrites en Python. Ainsi, pour installer un module, il suffit de :

-Ouvrir l’invite de commande (cmd)

-écrire pip install nomdumodule

(En effet la procédure est simple, mais nous avons passé beaucoup de temps avant de la trouver, nous avons auparavant essayé de télécharger les bibliothèques par d’autres moyens)

Il se peut que l’invite de commande retourne « Requirement already satified » et que donc le module soit déjà installé.