



avril 2022

# Rapport Mini Projet 2022

**Sujet :** Compression sans perte IRM

**Encadré par**

Mr. Adib Abdellah

**Présenté(e) par**

Charraki wissal  
Ebbeiye Salek  
Chlaikhy Salma  
Chair Wiam

# **REMERCIEMENTS**

On tient à remercier toutes les personnes qui ont contribué au succès de notre projet et qui nous ont aidé lors de la rédaction de ce rapport.

Tout d'abord, on adresse nos remerciements à notre professeur, Mr Adib Abdelah de la faculté des sciences et techniques de Mohammadia qui nous 'a beaucoup aidé à comprendre des notions techniques et de les manipuler dans ce projet. Son écoute et ses conseils nous ont permis de concevoir un sens de responsabilité et de sérieuxité.

On tient à remercier vivement notre chef de département, Mr Nourdinne Moumkin pour son travail. Grâce aussi à sa confiance on a pu nous accomplir totalement dans nos missions. Il fut d'une aide précieuse dans les moments les plus délicats.

On remercie également tous les professeurs, qui nous a beaucoup aidé à comprendre des différents domaines que ça soit informatique, Réseau ou Multimédia.

Nos remerciements à Pr. Mustapha Lkhider, Directeur de la FSTM, ainsi que tous les membres de cet établissement pour leur accueil et leur sympathie. Et enfin, nous voudrons adresser nos meilleurs voeux de joie, de réussite et de prospérité à tous les étudiants de la FSTM Mohammedia.

Enfin, On tient à remercier toutes les personnes qui nous ont conseillé et relu lors de la rédaction de ce rapport de stage et notre famille.

# Table of Contents

<b>INTRODUCTION.....</b>	<b>4</b>
<b>Format png.....</b>	<b>6</b>
Caractéristiques et utilisation du format png.....	6
Le gros inconvénient du format Png .....	6
Structure d'un fichier png Composition minimale d'un fichier PNG .....	7
Une signature PNG .....	7
Nommages des chunks.....	7
Composition d'un chunk.....	8
<b>Format TIF.....</b>	<b>9</b>
Propriétés des fichiers TIFF .....	9
Avantage et inconvénient des fichiers Tiff .....	10
Comparaison des fichiers TIFF, JPG, PNG et GIF .....	10
Structure d'un fichier Tiff.....	11
Structure d'entête du fichier.....	11
<b>Extension (.IRM) .....</b>	<b>12</b>
La recherche d'un algorithme .....	12
RLE (Run Length Encoding).....	12
HUFFMAN.....	12
Résoudre du problème.....	13
LZ78 (Lempel-Ziv '78) .....	13
LZW (pour Lempel-Ziv-Welch) .....	13
<b>Conclusion .....</b>	<b>14</b>
<b>Les étapes de compression .....</b>	<b>14</b>
1. La lecture Image.....	14
2. La compression .....	15
3. Le stockage des données .....	15
4. L'entête de fichier IRM .....	15
a. Extensions.....	15
c. Longueur et Largeur .....	16
d. Plan .....	16
e. Pas de lecture.....	16
<b>REALISATION DE L'APPLICATION .....</b>	<b>17</b>
1. Principe de l'interface graphique.....	17
2. Description de l'application .....	17
3. Plan de réalisation .....	17
<b>Conclusion.....</b>	<b>20</b>
<b>Bibliographie.....</b>	<b>21</b>

# INTRODUCTION

De nos jours, la puissance des processeurs augmente plus vite que les capacités de stockage, et énormément plus vite que la bande passante d'Internet, qui, malgré les nouvelles technologies, a du mal à augmenter car cela demande d'énormes changements dans les infrastructures telles que les installations téléphoniques.

La compression d'image est une application de la compression de données sur des images numériques. Cette compression a pour utilité de réduire la redondance des données d'une image afin de pouvoir l'emmager sans occuper beaucoup d'espace ou la transmettre rapidement. Elle peut être effectuée avec perte de données ou sans perte.

Pourquoi peut-on compresser ? Parce qu'une image où chaque point serait parfaitement indépendant des autres n'aurait pour nous pas d'intérêt : une image ne nous est utile que si elle contient des corrélations, qui dès lors qu'elles existent peuvent permettre les compressions en question.

Les méthodes les plus importantes de compression d'image sans perte sont :

- La méthode du codage des répétitions, utilisée sur les premiers scanners et télécopieurs ;
- Le codage entropique ;
- Les algorithmes à dictionnaire adaptable tels que LZW, davantage adaptés à l'information de type texte.

L'image matricielle est composée de petits points appelés « pixels » que l'on ne voit pas à l'œil nu. Lors de l agrandissement d'une image matricielle, cette dernière devient floue car les pixels ressortent, ce sont les carrés qui apparaissent sur l'écran.

Les formats d'image de cette catégorie les plus connus sont : JPEG, PNG, GIF et TIFF.

**La compression des données** est l'une des technologies habilitantes de la révolution multimédia. Elle consiste à convertir des fichiers de données en fichiers plus petits pour un stockage et une transmission efficace.

Par définition, la compression est le processus de représentation des informations sous une forme compacte. Ce processus traite les informations sous forme numérique, c'est-à-dire, sous forme de nombres binaires représentés par des octets de données. Ces données contiennent de très grands ensembles de données.

Les techniques de compression sans perte n'entraînent aucune perte d'informations. Si les données ont été compressées sans perte, les données d'origine peuvent être récupérées exactement à partir des données compressées. Cette compression sans perte consiste à reconstruire les données d'origine à partir du fichier compressé sans aucune perte de données.

Ainsi, les informations ne changent pas pendant les processus de compression et de décompression. Ces types d'algorithmes de compression s'appellent des compressions réversibles car le message d'origine est reconstruit par le processus de décompression.

Ces techniques de compression sans perte permettent de compresser des images médicales, du texte et des images [1]. Le codage est une étape essentielle dans la compression. Donc, la compression et le codage réduisent le nombre de bits par pixel à stocker ou à transmettre, en exploitant la redondance informationnelle dans l'image [2].

Notre travail de mini projet s'inscrit dans le cadre de la compression des images sans perte. Il a pour but de créer une nouvelle extension IRM qui utilise les algorithmes de compression sans perte, couramment utilisées en compression de données.

# **Format png**

Le format PNG (Portable Network Graphics, ou format Ping) est un format de fichier graphique bitmap (raster). Il a été mis au point en 1995 afin de fournir une alternative libre au format GIF, format propriétaire dont les droits sont détenus par la société Unisys (propriétaire de l'algorithme de compression LZW), ce qui oblige chaque éditeur de logiciel manipulant ce type de format à leur verser des royalties.

## **Caractéristiques et utilisation du format png**

Le PNG est un format ouvert d'images numériques, c'est-à-dire un format permettant de modifier à volonté une image ou un visuel après sa création. Il a été créé pour remplacer le format GIF qui ne permettait pas de Changer la balance des blancs d'une image, Optimiser la netteté ou en général, de retoucher les images sans dégrader leur qualité.

L'avantage en revanche, c'est que les caractéristiques du format PNG lui permettent d'enregistrer les visuels sans pertes de données même après compression. Il offre donc un rendu de bien meilleure qualité et supérieur à celui des formats JPEG.

Particulièrement adapté pour les visuels, les icônes ou les images représentant des textes et destinées à l'affichage web, Des visuels complexes en géométrie et en couleur, surtout si la question du poids du fichier ne pose pas de problème. Il surpasse également le GIF en ce qui concerne La réduction de la taille des fichiers, avec une compression sans perte de 5 à 25 % meilleure que celle proposée par GIF, et La qualité des visuels.

## **Le gros inconvénient du format Png**

Le gros inconvénient du fichier PNG est son poids. Comparativement au fichier JPEG, il est 7 fois plus lourd. Par exemple, un visuel de dimension 500×190 pixels enregistré au format JPEG et compressé à 60% pèse environ 8 ko. En comparaison, ce même visuel enregistré au format PNG pèse environ 75 ko, soit 10 fois plus.

## **Structure d'un fichier png**

### **Composition minimale d'un fichier PNG**

- Signature PNG - 8 octets
- Chunk IHDR pour l'en-tête - 25 octets
- Chunk IDAT pour les données - longueur variable
- Chunk IEND pour la fin de fichier - 12 octets

Un « chunk » est un gros morceau du fichier, un fragment d'information constituant une entité. Ce terme anglais est utilisé dans de nombreux formats multimédias.

Un fichier peut contenir plusieurs chunks de données IDAT ainsi qu'un chunk PLTE pour la palette à utiliser s'il s'agit d'une image dont les couleurs sont indexées.

Un fichier peut également contenir d'autres chunks secondaires, dont des informations textuelles.

### **Une signature PNG**

Un fichier PNG commence par une signature de 8 octets représenté par les valeurs décimales suivantes : 137 80 78 71 13 10 26 10, ou en hexadécimal : 89 50 4E 47 0D 0A 1A 0A.

La suite du fichier est décomposée en plusieurs parties de longueurs variables, appelées chunk.

### **Nommages des chunks**

Il existe 18 chunks officiels, dont 4 principaux et 14 secondaires [3].

Les chunks sont étiquetés (nommés). La casse est importante dans les noms des chunks. Chaque étiquette est définie par quatre caractères successifs, définissant un code mnémonique, sous forme de fourCC. Pour chaque chunk, si la première lettre de son nom est en capitale il s'agit d'un chunk critique, sinon c'est un chunk auxiliaire [4].

Voici un tableau regroupant les chunks les plus utilisés (Les quatre principaux en tête) :

Nom	Description	Contenu	Importance	Occurrence
IHDR	<i>Image header</i> En-tête du fichier	Largeur de l'image en pixels Hauteur de l'image en pixels Profondeur de bits (1, 2, 4, 8 ou 16) Type de couleur (0, 2, 3, 4, ou 6) Méthode de compression (0) Méthode de filtrage (0) Méthode d'entrelacement : 0 (sans) ou 1 (avec Adam7)	Obligatoire	Après la signature PNG
PLTE	<i>Palette</i> Palette de l'image	Table des couleurs	Facultatif	Entre IHDR et le 1 <sup>er</sup> chunk IDAT
IDAT	<i>Image data</i> Bloc de données	Données de l'image	Obligatoire	Entre IHDR ou PLTE et IEND
IEND	<i>Image trailer</i> Fin du fichier	néant	Obligatoire	En dernier
tIME	<i>Image last-modification time</i> Horodatage		Facultatif	N'importe où
iTXt	<i>International textual data</i> Info textuelle internationale (peut-être compressée zlib)		Facultatif	N'importe où
tEXt	<i>Textual data</i> Info textuelle non-compressée		Facultatif	N'importe où
zTXt	<i>Compressed textual data</i> Info textuelle compressée (zlib)		Facultatif	N'importe où

## Composition d'un chunk

Un chunk est composé de 4 parties :

LENGTH	TYPE	DATAS	CRC
Longueur des données	Type de chunk	Données dont la longueur en octets est spécifiée dans LENGTH	Contrôle
4 octets	4 octets	n octets	4 octets

**LENGTH** : La taille en octets du chunk, seulement ses datas. On ne prend pas en compte la taille, le type, ni le CRC.

**TYPE** : Le nom du chunk (ex : IHDR, IDAT, IEND, etc.).

**DATAS** : Les informations relatives au chunk sur n octets (relatif à LENGTH).

**CRC** : 4 octets de contrôle généré en utilisant l'algorithme suivant

# Format TIF

**TIFF** ou **TIF**, Tagged Image File Format, Il est capable de décrire des données d'image à deux niveaux, en niveaux de gris, en palette de couleurs et en couleurs dans plusieurs espaces colorimétriques.

TIFF est principalement utilisé sur le Web pour fournir des graphiques de haute qualité pour une impression **sans perte**.

L'objectif principal du format de fichier TIFF était de fournir un format de fichier d'image numérisée commun pour tous les fournisseurs de scanners de bureau. En commençant par la prise en charge du format d'image binaire uniquement, le format a évolué vers la prise en charge des images en niveaux de gris et en couleur au fil du temps.

## Propriétés des fichiers TIFF

Contrairement au format de fichier JPG, la compression ou la décompression d'un fichier TIFF se fait généralement sans perte. La taille du fichier est réduite sans affecter négativement la qualité d'origine

En raison de cette propriété, les TIFF conviennent à l'archivage et à l'impression d'images et de graphiques haute résolution. Un autre avantage du format de fichier est qu'il est indépendant de la plateforme, ce qui signifie que le format est également adapté à l'échange de fichiers, quel que soit le système d'exploitation utilisé. Les systèmes d'exploitation Windows et Mac ont tous deux un logiciel intégré pour ouvrir les fichiers TIFF. Cependant, pour éditer un fichier TIFF, il est généralement nécessaire de le convertir d'abord dans un autre format.

La **compression sans perte** entraîne également des quantités de données plus importantes que les autres formats d'image. Sur le Web cependant, des temps de chargement courts sont d'une grande importance afin d'obtenir une expérience utilisateur positive et au bout du compte un meilleur classement Google. Dans le domaine du Cloud également, l'utilisation de l'espace de stockage est un facteur important dans le choix du bon format de fichier.

Bien que la taille d'un fichier TIFF soit limitée à un maximum de **quatre gigaoctets**, d'autres formats de fichiers sont préférés sur Internet. La perte de qualité associée et les résolutions inférieures sont à peine visibles en ligne et sont un moindre mal par rapport à la taille des fichiers. Afin de fournir des images haute résolution en qualité imprimable, le format TIFF peut être utilisé sur Internet

## Avantage et inconvénient des fichiers Tiff

Le tableau suivant compare les propriétés les plus importantes d'un fichier TIFF et présente les avantages et les inconvénients du format.

Avantages	Inconvénients
✓ indépendant de la plate-forme	✗ importante quantité de données
✓ prend en charge les niveaux	✗ traitement difficile
✓ compression sans perte de données	✗ grande complexité
✓ transparencies via le canal alpha	✗ limité à un maximum de quatre gigaoctets par fichier
✓ haute sécurité des données	
✓ idéal pour l'impression	

## Comparaison des fichiers TIFF, JPG, PNG et GIF

Les formats d'image les plus courants sont JPG, PNG, GIF et TIFF. Le tableau suivant compare les propriétés les plus importantes de ces différents formats.

	TIFF	JPG	PNG	GIF
Utilisation	Impression	Web	Web ; Images avec des transparencies et de nombreuses nuances de couleurs	Animations
Compression sans perte de données possible	oui	non	oui	oui
Taille des fichiers	maximum 4 Go	minime	minime	infime
Spectres de couleurs	CMJN, RVB et CIELAB ; spectre de couleurs complet	CMJN et RVB ; spectre de couleurs complet	seulement RVB ; spectre de couleurs complet	seulement RVB; Limité à 256 couleurs
Transparencies possibles	oui	non	oui	oui
Convient pour l'impression	oui	oui	non	non

## Structure d'un fichier Tiff

Un fichier TIFF commence par les deux caractères ASCII MM pour big endian ou II pour little endian. Les deux octets suivants représentent le nombre 42, en big endian ou little endian.

Les images au format TIFF sont limitées à une taille de 4 Go (taille de l'offset sur 32 bits). La bibliothèque LibTIFF 4.0 (parue en 2007) a introduit le format BigTIFF, permettant des images d'une taille supérieure grâce à des offsets de 64 bits. Ce format ne fait pas l'objet d'un standard officiel.

## Structure d'entête du fichier

L'entête TIFF a une longueur de 8 octets et contient trois champs. Il occupe toujours les 8 premiers octets d'un fichier de format TIFF. Il n'y a aucun moyen fiable de décoder un fichier TIFF sans inspecter d'abord l'entête, donc la lecture de l'entête devrait toujours être la première tâche d'un décodeur. Voici la structure d'entête du fichier «.TIF » ou «.TIFF »

L'en-tête du fichier TIFF de 8 octets contient les informations suivantes :

Déplacement	Taille	Description						
0	2 octets	Ce champ permet d'indiquer le format de signature. Voici la chaîne de caractères de 2 octets correspondant aux formats possibles : <table border="1"><thead><tr><th>Valeur</th><th>Description</th></tr></thead><tbody><tr><td>"II"</td><td>Cette valeur permet d'indiquer le format des nombres Intel.</td></tr><tr><td>"MM"</td><td>Cette valeur permet d'indiquer le format des nombres Motorola.</td></tr></tbody></table>	Valeur	Description	"II"	Cette valeur permet d'indiquer le format des nombres Intel.	"MM"	Cette valeur permet d'indiquer le format des nombres Motorola.
Valeur	Description							
"II"	Cette valeur permet d'indiquer le format des nombres Intel.							
"MM"	Cette valeur permet d'indiquer le format des nombres Motorola.							
2	2 octets	Ce champ permet d'indiquer la version du format TIFF.						
4	4 octets	Ce champ permet d'indiquer le déplacement à effectuer vers le premier bloc « IFD »						

# **Extension (.IRM)**

Le format IRM (Informatique réseau et Multimédia) est une extension d'une compression sans perte s'inspirer des autres formats "TIFF" ET "PNG" qui utilisent les algorithmes de compression sans perte (Huffman, LZW, LZ78, RLE, LZ77)

## **La recherche d'un algorithme**

L'étape la plus importante dans une compression est l'application d'algorithme qui permet de réduire la taille des données. Chacun des algorithmes a ces propres états d'utilisations par exemple lorsque la répétition des valeurs dépasse trois la meilleure chose et d'appliquer RLE.

Au cours de la préparation d'une compression sans perte, on a testé les différents algorithmes. Pendant chaque test, on découvre des problèmes différents mais on essaye de chercher la solution la plus convenable.

## **RLE (Run Length Encoding)**

Un algorithme très efficace pour une répétition successive qui dépasse trois fois de façon qu'à la place de l'écrire trois fois il suffit d'écrire l'élément et le nombre d'occurrences. Mais lorsque l'élément ne se répète pas plus que trois fois on tombe sur une augmentation d'espaces est ce n'est pas du tout notre but et si la répétition est de deux on ne va rien gagner.

### **EXEMPLE :**

La phrase suivante : mammi

Avec RLE : (1, m) (1, a) (2, m) (1, i)

Du coup à la place que le mot soit écrit sur 5 octets on va l'écrire sur 8 octets

## **HUFFMAN**

Un algorithme qui est basé sur l'occurrences des éléments de façon que l'élément apparaît le plus prend un code plus petit et vice versa.

au cours de traitement des images avec l'algorithme Huffman on a trouvé des bonnes résultats au niveau de la compression et les résultats vont de plus mieux lorsque on applique algorithmes RLE sur les données récupérer par le code Huffman .mais la seule chose qui nous a empêché de ne l'est pas l'utiliser c'est qu'on a besoin de transmettre l'information en forme binaire et cette étape besoin de fixer nombre de bits réservé pour chaque élément et cette étape est impossible pour le dictionnaire du Huffman car le zéro est signifiant et si on veut l'utilise après on doit éliminer les zéros mais on savait pas le nombre de zéros qu'il faut à chaque fois.

## EXEMPLE :

010 : £

10 : a

Lorsqu'on fixe la taille par l'ajout de zéro à droite on tombe sur le même code alors que ne revient pas au même caractère.

## Résoudre du problème

Pour éviter ce conflit on a décidé que dans l'entête on envoie le pas de lecture pour chaque élément du dictionnaire mais ça à causer une réduction performante du programme ainsi il a pris plus de temps qu'avant. Donc on compare le taux obtenu avec le temps qu'a pris le programme lors de la compression et la décompression on a constaté que cette méthode n'est pas convenable à l'utiliser.

## LZ78 (Lempel-Ziv '78)

Le principe de LZ78 consiste à parcourir la source en apprenant les suites de symboles les plus fréquentes. Ces suites de symboles, appelées préfixes sont stockées dans un dictionnaire, construit au fur et à mesure que l'on parcourt la source. Plus des suites de symboles sont fréquentes, plus elles donneront lieu à des longs préfixes dans le dictionnaire.

Lors de l'utilisation de cet algorithme on a eu des bons résultats parce qu'on n'a pas besoin d'écrire l'élément à chaque fois, il suffit qu'il existe déjà dans le dictionnaire et on ajoute juste le dernier élément.

## LZW (pour Lempel-Ziv-Welch)

LZW est un algorithme de compression de données sans perte. Il s'agit d'une amélioration de l'algorithme LZ78 inventé par Abraham Lempel et Jacob Ziv en 1978. LZW fut créé en 1984 par Terry Welch.

La meilleure chose dans LZW qui n'est pas dans LZ78 que le dictionnaire adaptif alors on n'a pas besoin d'envoyer le dictionnaire et même pas on n'a pas besoin d'ajouter les éléments déjà existant en dictionnaire et continuer la numération à partir du dernier élément dictionnaire de base.

## EXEMPLE :

Message à envoyer : véridique ! Dominique pique-nique en tunique

Le code à envoyer : 118 101 114 105 100 105 113 117 101 32 33 32 100 111  
109 105 110 261 263  
32 112 273 264 272 262 264 101 110 32 116 117 279 263

Alors si on envoie le message sans application du LZW on doit envoyer 45 éléments et si on utilise LZW on envoie juste 33 éléments

## Conclusion

Le choix de l'algorithme est basé sur le taux ainsi que le temps des opérations de compression et décompression. On fin après des plusieurs tests, on a décidé d'adopter l'algorithme qui s'exécute dans une durée courte et nous permet de gagner plus d'espace. Comme nous avons déjà vue que RLE rapide mais ne réduit pas la taille, HUFFMAN et RLE donne un meilleur résultat mais il prend un temps énorme pour la décompression.

Finalement, on a adopté à utiliser LZW et LZ78 qui prend un temps acceptable avec une réduction assez remarquable.

## Les étapes de compression

La compression d'une image se passe par plusieurs étapes différentes.

Dans un premier lieu, la lecture de l'image ensuite la compression puis le stockage de cette dernière .et au cours de ces étapes on prépare parallèlement l'entête de l'image.

### 1. La lecture Image

La lecture d'image a aussi un effet sur la compression de façon que le mode de lecture (horizontal, vertical, zigzag) de l'image et impose l'ordre des éléments pour certaine image. Le choix de mode de lecture dépend de répétitions des pixels et leurs positionnements dans l'espace.

#### *1. La première étape :*

On commence par la lecture de l'image avec la fonction `Image. Open ()` de la bibliothèque PIL, qui nous retourne une liste ligne, chaque ligne contient une liste de colonne qui contient des valeurs de pixel (EXEMPLE: pour le plan rgb [rr, gg, bb])

#### *2. La deuxième étape :*

Les données résultantes par la dernière fonction, on les prend et on les lit avec l'un des méthodes Verticale ou Horizontale ou bien ZIG-ZAG et on transforme sous forme d'une liste et on garde la valeur des dimensions de l'image plus le mode de lecture pour l'utiliser au niveau d'entête.

#### *3. La troisième étape :*

La liste obtenue dans la dernière étape, on la transforme en chaîne de caractères avec la fonction `Chr ()` et `". Join(liste)`.

## **2. La compression**

On prend les données obtenues à partir de la dernière étape et on fait l'appel à la fonction déjà prédéfinie de compression (LZ78) cette dernière nous retourne une liste de valeur prête à envoyer.

## **3. Le stockage des données**

On prend cette dernière liste des valeur et on cherche la plus grande valeur dans la liste avec la fonction max() et grâce des deux fonction Len() et format() en obtenu le nombre de bit besoin pour coder le plus grand numéro et on garde le nombre de bit nécessaire et on l'utilise comme pas de lecture et on ajoute dans l'entête.

En premier temps, on doit commencer par l'entête qu'on va la détailler dans la partie suivante mais maintenant on parle juste de codage des données.

Après l'écriture d'entête, on commence la lecture du dernière liste et on prend chaque élément et l'écrire en format binaire avec le nombre de bit déjà fixé avec la fonction zfill().

## **4. L'entête de fichier IRM**

L'entête c'est la première chose qu'on lire dans un fichier, et c'est la clé pour ouvrir n'importe quel fichier, c'est là où on trouve les différentes informations sur les données stocker la taille, l'extension, le mode de lecture et le pas de lecture.

Le fichier IRM est compris un entête de taille 8 octets et demi, contient les éléments ci-dessous :

### **a. Extensions**

On a réservé pour l'extension 3 octets ; chaque lettre va être codée sur un octet

### **b. Mode de lecture :**

On a choisi de réserver 2bit pour le mode de lecture car on a trois modes et pour deux bits nous donne 4 états possible donc c'est assez suffisant.

### **c. Longueur et Largeur**

En premier temps, on a décidé de laisser un seul octet pour la longueur et la largeur mais on a trouvé que ce n'est pas suffisant car le nombre maximal donnée par un octet est 256 alors la plupart des images dépasse ça. C'est pour ça on a laissé deux octets pour la longueur et la largeur.

Avec ces deux octets, on peut avoir une valeur maximale de 65 536.

### **d. Plan**

Le nombre de plan des images peut prendre les valeurs suivantes : 1, 3, 4  
C'est pour ça on a laissé 2 bits.

### **e. Pas de lecture**

Puisque on a changé le nombre de bit écrit pour chaque élément et qu'il va varier d'une image à une autre, ça dépend de la valeur maximale on a laissé un octet pour le pas de lecture lors de la décompression

# REALISATION DE L'APPLICATION

## 1. Principe de l'interface graphique

Le but de cette application est la **compression d'une image** en réduisant sa taille originale pour le fait de **stockage** et de **transmission**.

## 2. Description de l'application

Notre Application est faite sous **Tkinter** dans **Jupyter**, le Thème de couleur choisi désigne la passion, l'énergie, la force et la volonté.

Le **logo** crée reflète l'objectif de la réalisation du notre mini projet.

L'application est adressée aux **utilisateurs** qui veulent minimiser la taille d'une telle image.



## 3. Plan de réalisation

L'application est constituée d'un **Panel** qui contient **trois cadres**.

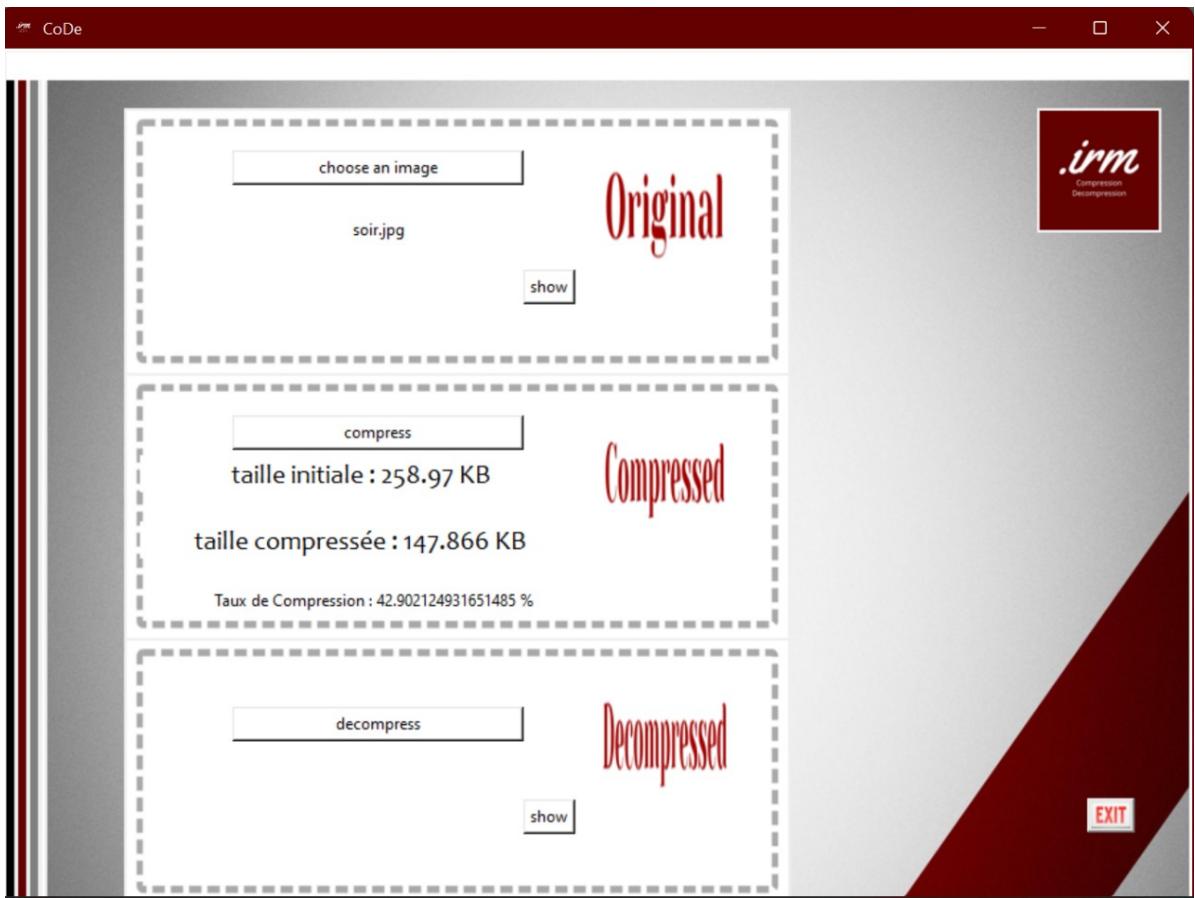
Le **premier cadre 'Original'** est consacré par le fait qu'un utilisateur clique sur le bouton **choisir une image** et son nom va être afficher avec son extension d'origine dans un **Label**.

Ensuite, le bouton **Show** a deux conditions ; la première et s'il effectue un clique avant de choisir une image, elle va affichée un **Show Message** que l'image n'est encore choisie, la deuxième et s'il effectue un clique après l'action de bouton choisir, elle va afficher l'image pour qu'il puisse la visualiser et la comparer avec celle après décompression.

Le **deuxième cadre ‘Compressed’** est consacré par le fait qu'un utilisateur clique sur le bouton “**Compresse**” et il va être affiché trois **Labels** ; le premier affiche la taille originale du fichier et le deuxième affiche la taille du fichier compressé ; et le dernier fait calculer le taux de compression.

Enfin, le **dernier cadre ‘Decompressed’** est consacré par le fait qu'un utilisateur clique sur le bouton “**Decompress**” et il va remarquer qu'il va attendre une certaine durée qui dépend du nombre de pixel pour faire la reconstruction de l'image et pour qu'il puisse voir si les données sont les mêmes que l'originale, il y a un bouton **Show** qui va afficher l'image reconstruite et faire la comparaison entre les deux images.

L'application va être visualiser comme la suivante lors de l'ouverture :



## **Conclusion**

La performance de compression d'un fichier est la transmission des données dans une courte durée, plus que la taille est réduite plus que la vitesse de transmission va être rapide ainsi que la qualité du fichier va être excitée.

Les algorithmes de compression des fichiers permettent non seulement de réduire la taille de fichier mais aussi la charge de traitement et de transmission des données.

Dans ce mini projets, nous avons étudié et comparé les performances de compression sans perte de quartes techniques de codage source, couramment utilisées en compression de données, à savoir : le codage de Huffman, RLE ,LZ78 et LZW sur des images RGB.

## Bibliographie

- [1] S. Melwin, A. S. Solomon, M. N. Nachappa, "A survey of compression techniques," 2(1) : 152-6, Int. J. Recent. Technol. Eng., 2013.
  - [2] D. Mekimah, H. Sadmi, "Compression d'image par la technique EZW," Département d'Électronique, Mémoire de Master, Université de Jijel, 2018.
  - [3] (En) ISO/IEC, « Portable Network Graphics (PNG) Specification (Second Edition) » [archive], sur w3.org, 10 novembre 2003.
  - [4] ↑ Revenir plus haut en : a et b (en) ISO/IEC, « Portable Network Graphics (PNG) Spécification (Second Edition) — §5.4 - Chunk naming conventions » [archive], sur w3.org, 10 novembre 2003.
- 
- <https://www.ionos.fr/digitalguide/sites-internet/web-design/quest-ce-que-le-format-tiff/>
- <https://compression.fiches-horaires.net/la-compression-sans-perte/lz78-et-lzw-la-compression-par-dictionnaire/>
- [https://fr.m.wikipedia.org/wiki/Compression\\_d%27image#:~:text=La%20compression%20sans%20perte%20est,qualit%C3%A9%20de%20l'image%20restitu%C3%A9e.](https://fr.m.wikipedia.org/wiki/Compression_d%27image#:~:text=La%20compression%20sans%20perte%20est,qualit%C3%A9%20de%20l'image%20restitu%C3%A9e.)
- : <http://turrier.fr/articles/uncompressed-tiff-structure-file/structure-fichier-tiff-non-compresse.php>
- <https://compression.fiches-horaires.net/la-compression-sans-perte/deflate-lalgorithme-que-vous-retrouvez-partout/#3-le-fonctionnement-de-deflate>
- <https://www.commentcamarche.net/contents/1195-compression-de-donnees>