

TIM

Compte-rendu de TP

Compression d'images en C

SE2A4

Kevin Doolaeghe

Sommaire

Introduction	3
Compression par quantification uniforme	4
Compression par quantification adaptative	5
Quantification de Floyd-Steinberg	6
Conclusion	6
Annexes	7
Portable pixmap	7
Structure & format	7
1. PGM	7
2. PPM	8
Bibliothèque utilisée	8

1. Introduction

Le TP a pour objectif de réaliser un programme en C permettant par différents moyens de compresser une image. En effet, le programme doit pouvoir lire des images au format PPM et/ou PGM pour compresser l'information qu'elles contiennent (la taille du fichier n'est pas affectée).

On choisit de travailler lors de ce TP avec les images ci-dessous respectivement au format PGM (à gauche) et au format PPM (à droite).

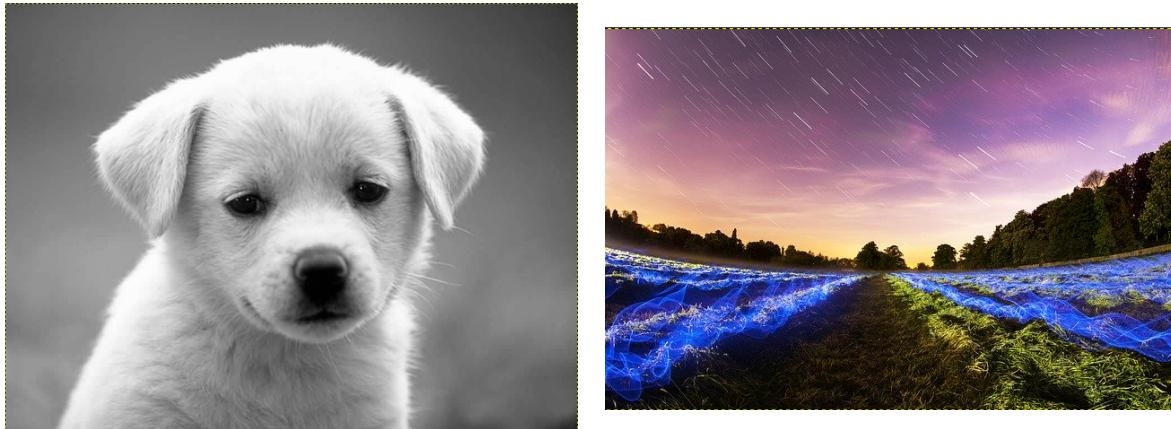


Figure n°1 : Images utilisées lors du TP.

2. Compression par quantification uniforme

Dans un premier temps, on réalise une quantification uniforme de l'image. Pour cela, on utilise un facteur de compression N et pour chaque pixel de l'image, on applique le traitement :

$$\text{pixel} = N \times (\text{pixel} / N)$$

On obtient donc les résultats suivants :



Figure n°2 : Images ayant subi une compression par quantification uniforme.

On remarque un étalement des pixels sur les images compressées. Il y a moins de nuances et de variations des couleurs/niveaux de gris. L'histogramme après compression illustre bien cela :

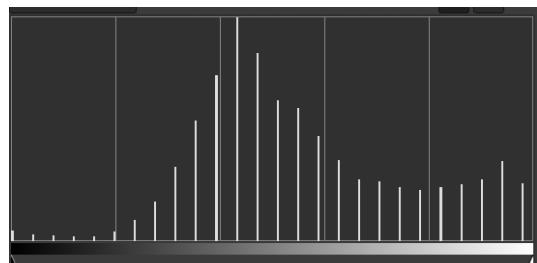
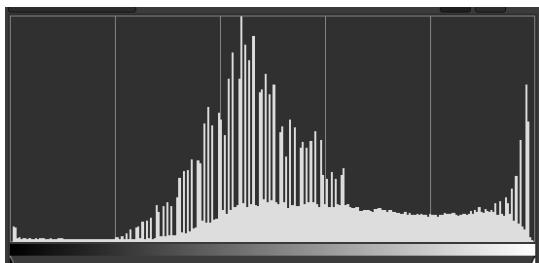


Figure n°3 : Histogramme de l'image originale (à gauche) et après compression (à droite).

3. Compression par quantification adaptative

Ensuite, on réalise une quantification adaptative de l'image. Pour cela, on utilise un seuillage par bloc. On choisit de découper l'image en M blocs et de moyenner la valeur des pixels du bloc. On obtient finalement une matrice de seuillage que l'on applique à l'image (par rapport au niveau maximum).

On obtient donc les résultats suivants :

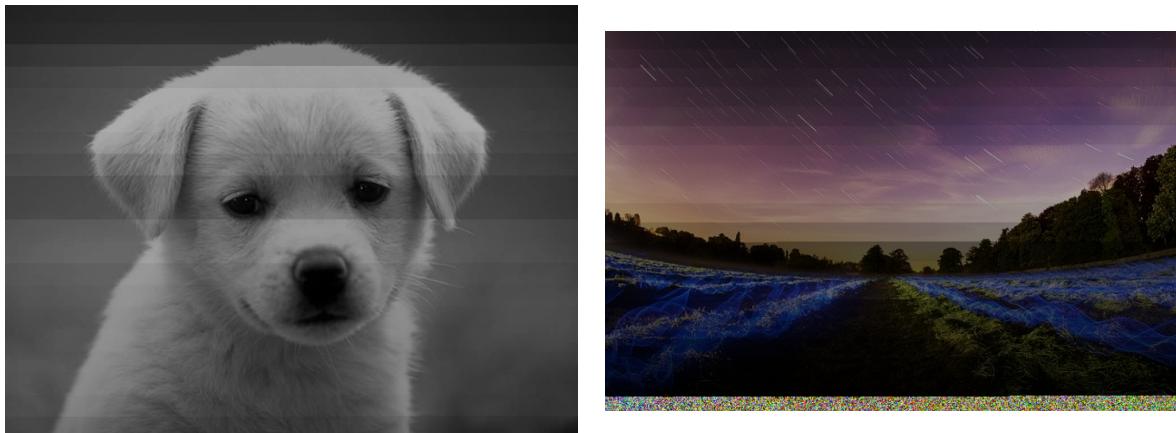


Figure n°4 : Images ayant subi une compression par quantification adaptative.

La compression adaptative donne des images plus nettes que la quantification uniforme. Pourtant, on constate sur ces images l'apparition de lignes et également un assombrissement. Enfin, on remarque que l'histogramme de l'image est l'original mais compressé :

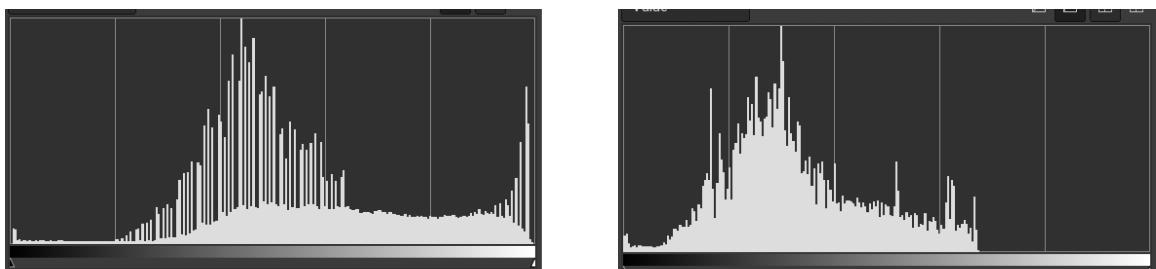


Figure n°5 : Histogramme de l'image originale (à gauche) et après compression (à droite).

4. Quantification de Floyd-Steinberg

Enfin, on réalise une quantification par l'algorithme de Floyd-Steinberg des images précédentes. Cet algorithme quantifie pixel par pixel l'image et répartie l'erreur d'échantillonnage aux pixels suivants (cf. illustration ci-dessous).

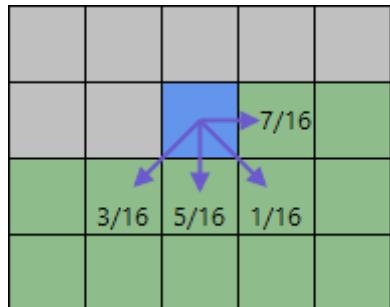


Figure n°6 : Représentation de l'algorithme de Floyd-Steinberg.

L'algorithme recréé en langage C permet d'obtenir les résultats suivants :

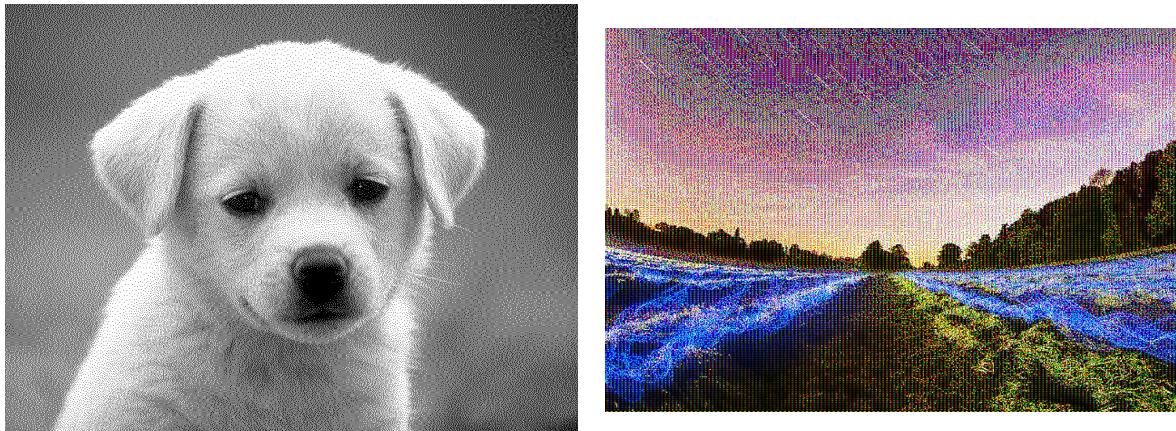


Figure n°7 : Images ayant subi une compression par quantification de Floyd-Steinberg.

5. Conclusion

En conclusion, la quantification de Floyd-Steinberg permet de compresser l'information en gardant l'image la plus semblable à l'originale. La quantification adaptative réalisée (seuillage par bloc) est aussi performante mais a l'inconvénient de former des lignes sur l'image. Enfin, la quantification uniforme est la plus simple à mettre en place mais est cependant la moins efficace car beaucoup de détails de l'image sont perdus.

6. Annexes

Portable pixmap

Le portable pixmap file format (PPM), le portable graymap file format (PGM) et le portable bitmap file format (PBM) sont des formats de fichier graphique utilisés pour les échanges. Ils ont été définis et sont utilisés par le projet NetPBM. Ils proposent des fonctionnalités élémentaires et sont utilisés pour convertir les fichiers de type pixmap, graymap et bitmap entre différentes plateformes. Plusieurs applications désignent cet ensemble de trois formats comme le format PNM (portable anymap).

Wikipedia | <https://fr.wikipedia.org/wiki/PortablePixmap>

Structure & format

Les fichiers PBM, PGM ou PPM sont composés sur la même base :

- le nombre magique du format (deux octets) : il indique le type de format (PBM, PGM, ou PPM) et la variante (binaire ou ASCII)
- un caractère d'espacement (espace, tabulation, nouvelle ligne)
- la largeur de l'image (nombre de pixels, écrit explicitement sous forme d'un nombre en caractères ASCII)
- un caractère d'espacement
- la hauteur de l'image (idem)
- un caractère d'espacement
- les données de l'image : succession des valeurs associées à chaque pixel
- l'image est codée ligne par ligne en partant du haut
- chaque ligne est codée de gauche à droite.

Toutes les lignes commençant par un croisillon # sont ignorées (lignes de commentaires).

Wikipedia | <https://fr.wikipedia.org/wiki/PortablePixmap>

1. PGM

Une image au format PGM est constituée de nuances de gris.

- Un fichier pgm binaire a pour nombre magique P5.
- Un fichier pgm ASCII a pour nombre magique P2.

```
P2
# Affiche le mot "GG"
12 7
15
0 0 0 0 0 0 0 0 0 0 0 0
0 3 3 3 3 0 0 7 7 7 7 0
0 3 0 0 0 0 0 7 0 0 0 0
0 3 0 3 3 0 0 7 0 7 7 0
0 3 0 0 3 0 0 7 0 0 7 0
```

```
0 3 3 3 3 0 0 7 7 7 7 0  
0 0 0 0 0 0 0 0 0 0 0 0
```

2. PPM

Ce format est utilisé pour des images en couleur.

- Un fichier ppm binaire a pour nombre magique P6.
- Un fichier ppm ASCII a pour nombre magique P3.

```
P3  
# P3 = RGB ASCII  
3 2  
# 3 colonnes x 2 lignes  
255  
# 255 pour valeur maximum  
255 0 0 0 255 0 0 0 255  
255 255 0 255 255 255 0 0 0
```

Bibliothèque utilisée

Afin de réaliser le traitement d'images au format PPM ou PGM, ce TP utilise la bibliothèque suivante :

<https://github.com/nkkav/libpnmio>