DODANE Jean-Baptiste
ROUSSILLON Cyril

TS3

TPE 2001 - 2002

SYNTHESE

Nous avons décidé de choisir comme sujet **La compression graphique informatique**, d'une part parce qu'il correspondait au thème **Image**, et d'autre part parce qu'étant passionnés d'informatique et plus particulièrement de programmation, nous étions très motivés pour étudier des techniques qui nous étaient familières en utilisation, mais dont nous ne connaissions pas le fonctionnement, et surtout nous voulions implémenter nous-même ces méthodes. Notre but était au départ d'étudier une compression conservative, et une compression non conservative. Emportés par le sujet, nous avons traité plusieurs méthodes.

Généraux à tous les types de données ou spécifiques à la compression des images, les algorithmes examinés nous ont livrés leurs secrets. Le détail de toutes ces méthodes avec leur implémentation se trouve dans le dossier, mais voici un résumé de leurs principales caractéristiques et fonctionnement :

- **RLE** (Run Length Encoding) : c'est l'une des méthodes de compression les plus simples qui soit. Le principe consiste à remplacer une séquence de n éléments v, par un couple (n, v), c'est un peu comme remplacer des additions par une multiplication. Cette méthode a l'avantage d'être très simple et rapide, mais n'est réellement efficace que pour des images géométriques avec peu de couleurs. C'est pourquoi elle est peu utilisée seule, mais très souvent en complément d'une autre méthode.
- **Huffman** : le codage de Huffman est un codage statistique, c'est à dire qui se base sur la fréquence d'apparition des caractères pour les coder : plus un caractère apparaît souvent, plus son code sera court et vice-versa. L'avantage du codage de Huffman est qu'il est préfixé : aucun code n'est le préfixe d'un autre, ce qui permet un décodage unique. Il est utilisé notamment dans le JPEG, ou avec le codage prédictif (voir plus loin).
- **Algorithmes à dictionnaire** : aussi appelés algorithmes à substitution de facteurs, le principe est de remplacer des séquences de codes (les facteurs), par un code plus court qui est l'indice de ce facteur dans un dictionnaire. Les plus connus sont LZ77, LZ78 et LZW.
- **Prédictif** : Le codage prédictif consiste à prédire la valeur d'un pixel en fonction de ceux l'entourant. En effet des pixels proches ont de grandes chances d'avoir une valeur voisine. On n'enregistre ensuite que la différence entre la valeur prédite et la valeur effective, qui est généralement petite, et que l'on va ensuite encoder avec un codage de Huffman qui sera efficace, car les nombres étant petits, ils sont plus fréquents. Cette méthode est donc spécifique aux images.
- **JPEG** (Joint Photographic Experts Group) : la norme JPEG est une méthode spécifique aux images, et non conservative, c'est à dire qu'elle perd une partie des données (la qualité de l'image est moins bonne), mais en contrepartie les taux de compression sont beaucoup plus élevés. Le principe est de filtrer l'image (en supprimant les fréquences hautes auxquelles nous sommes moins sensibles, à l'aide d'une transformée en cosinus discrète), puis de l'encoder avec une méthode statistique de type Huffman.

Nous voulions au début étudier les formats, comme le format GIF (utilisant l'algorithme LZW) ou JFIF (JPEG), c'est-à-dire la façon dont les données sont organisées, les paramètres des méthodes, qui sont pour les formats élaborés connus assez complexes, mais qui sont loin d'être la partie la plus intéressante. Nous avons préféré approfondir les méthodes de compression générales, et justement étudier les conséquences de différents paramètres pour chaque méthode, pour trouver les plus efficaces. Nous avons donc créé nos propres formats qui utilisent ces méthodes :

- **RTPE** : ce format utilise la méthode RLE. Le codage RLE est la compression de base. Elle est d'ailleurs utilisée dans de nombreux algorithmes (Prédictif, Huffman) pour sa rapidité et son efficacité. Mais c'est seule qu'elle compresse les fichiers BITMAP. Bien que répondant à un langage bien particulier, nous avons testé la réponse de notre algorithme en changeant quelques valeurs fondamentales. Ce qui l'a finalement rendu plus puissant que les valeurs Microsoft, pas toujours très logiques. Mais RLE est synonyme de nombreux sens de codages. Nous avons implémenté les 2 principaux et conçu différents autres. Mais sa technique, moins intéressante que les méthodes suivantes, nous a orienté vers les formats suivants :
- **PTPE** : ce format utilise le codage prédictif. Nous avons fait plusieurs essais sur la manière de faire la prédiction, et sur le codage de Huffman, pour voir quels paramétrages donnent les meilleurs résultats. Nous avons retenu pour la prédiction de faire la moyenne entre le pixel juste au-dessus et juste avant. Pour le codage de Huffman, nous avons décidé d'utiliser la variante adaptative, c'est-à-dire que l'on se sert des pixels précédemment compressés pour établir les fréquences d'apparition de chaque caractère. Nous avons également inclus une RLE pour les suites de pixels uniformes.
- **WTPE** : ce format utilise l'algorithme à dictionnaire LZW. Commencé en 1977 par Lempel et Ziv, puis complété 1984 par Welch, il est le seul a être breveté. Chaque séquence, même la plus petite, est inscrite dans un dictionnaire. Puis les séquences suivantes sont exprimées en fonction du contenu du dictionnaire. Ce dernier n'a pas besoin d'être transmis : il est dynamique, ce qui permet un gain considérable de place. De même, le nombre de bits sur lequel les informations sont codées est variable : l'espace gaspillé est réduit au minimum. Toutefois, le principal inconvénient est le temps : notre lutte contre le chronomètre s'est achevée avec un bilan mitigé, mais en divisant quand même la durée initiale par 9.
- **JTPE** : ce format utilise la méthode JPEG. Nous avons testé nos propres tables de quantification (qui servent pour le filtrage, précisant sur quelles fréquences on va diminuer la précision), pour étudier le meilleur rapport entre qualité et taux de compression. Nous avons comme pour le format PTPE choisi la variante adaptative du codage de Huffman, couplée à une RLE, et choisit les paramétrages optimums en essayant plusieurs combinaisons.

Nous avons également étudié le fonctionnement de la transformée en cosinus discrète (DCT) pour le JPEG, ses différences avec la transformée de Fourier et quels sont ses avantages. Le JPEG nous a aussi permis d'étudier les pertes engendrées par la compression, les éléments caractéristiques et leur explication.

Nous avons implémenté tous ces algorithmes sous Borland Delphi, en pascal. Notre production consiste en plus du dossier, en l'écriture d'un logiciel (avec ses sources complets sur le cd-rom accompagnant le dossier) qui permet de gérer nos formats, et d'étudier en détail et en pratique toutes les méthodes.

Mais pour réaliser ce programme il nous a fallu nous perfectionner dans le domaine de l'informatique graphique. Les couleurs RGB, YUV ou HSB, les standards Windows ou Internet ou encore le traitement des bits en pascal nous ont donné du fil à retordre, mais les connaissances acquises nous sont maintenant extrêmement précieuses.

L'écriture de ce logiciel nous a pris une très grande partie du temps passé sur nos TPE, et nous avons passé beaucoup plus de deux heures par semaine dessus. Mais c'est un sujet qui nous intéressait naturellement, et nous ne regrettons pas du tout ce travail car le temps que l'on a passé à écrire ce programme, de toute manière nous l'aurions passé à programmer en loisir, et peut-être de manière moins constructive.

Finalement, nous pouvons dire que nous avons exploré 50 ans de compression informatique, secteur qui suit l'expansion des moyens de stockage. Car si la capacité de nos disques durs augmente incroyablement, les données à stocker sont toujours plus nombreuses. Et un fichier compressé est en plus un fichier plus rapide : a travers les réseaux, ou à l'enregistrement sur support, le temps n'est pas négligeable.

Le domaine de l'imagerie satellitaire ou de la photographie aérienne ne trouve pas de difficultés pour l'inspiration. Mais pour le stockage des informations, les gains de 95% fournis par les compressions que nous avons étudiées ne sont en aucun cas superflues.