Compte rendu du TP n° 2

Code d'Huffman et code prédictif

Thibaut Castanié
M2 IMAGINA

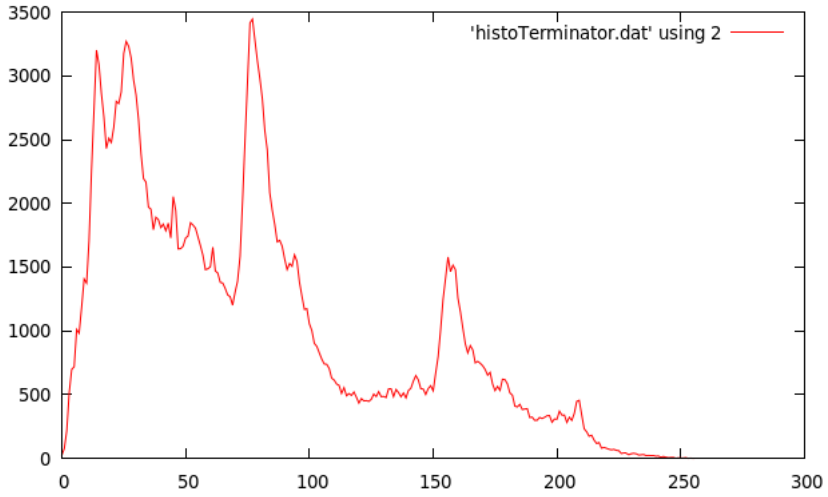
1er novembre 2015

1 Image choisie

L'image .pgm utilisée pour le TP

2 Dans l'espace des pixels

Afin d'avoir une idée des probabilités d'apparition de chaque niveau de gris, on trace l'histogramme de l'image. Pour l'afficher, l'utilitaire `gnuplot` est utilisé.



Histogramme de l'image terminator.pgm

L'algorithme d'Huffman utilisé pour ce TP provient du site communautaire RosetaCode.org (http://rosettacode.org/wiki/Huffman_coding#C++).

L'algorithme utilise une *map*, il suffit alors de la remplir avec les pixels de notre image.

```
for (int i = 0; i < nW; i++){
    for (int j = 0; j < nH; j++){
        frequencies[ImgIn[i*nW+j]]++;
    }
}
```

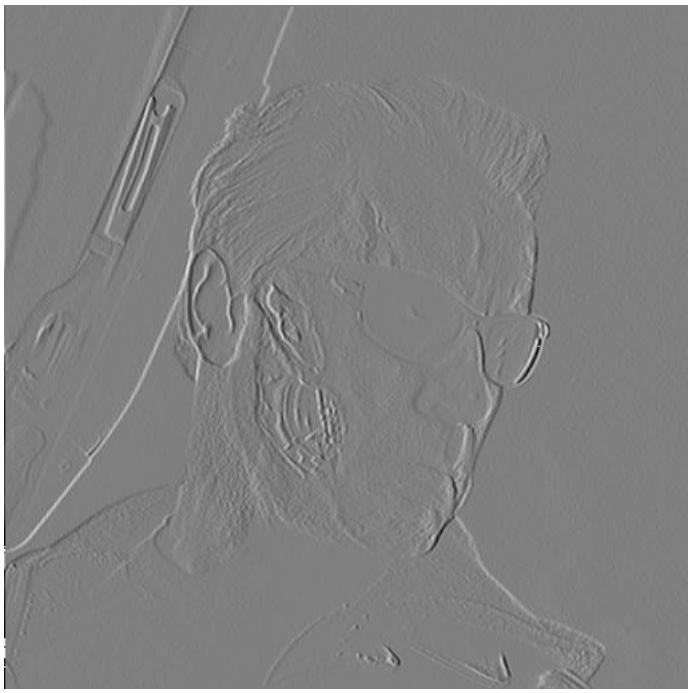
On créé un fichier compressé avec l'algorithme de Huffman, chaque octet étant codé sur 8 bits, on concatène les représentation de chaque symboles. On obtient ainsi un fichier binaire compressé.

- Taille *terminator.pgm* : 262 198 octets.
- Taille *terminatorComp.comp* : 187 011 octets.

Taux de compression : $\tau = \frac{262198}{187011} = 1.402045 \approx 1.4$

3 Dans l'espace de prédilection

En utilisant une méthode de prédiction des voisins, on obtient la carte des différences suivante.



Carte des différences de terminator.pgm

Ensuite, on passe de nouveau l'image dans l'algorithme d'Huffman, et on obtient les chiffres suivants :

- Taille *terminatorDiff.pgm* : 262 159 octets.
- Taille *terminatorDiffComp.comp* : 62 650 octets.

Taux de compression : $\tau = \frac{262159}{62650} = 4.184501 \approx 4.2$

4 Comparaison et conclusion

En utilisant l'espace des pixels, on obtient un taux de **1,4**. En utilisant l'espace de prédilection, on obtient un bon taux de **4,2**. Ainsi ce dernier est bien plus efficace, en raison de la probabilité d'apparition d'un petit nombre de pixels bien plus élevée.

Afin d'obtenir un meilleur taux de compression, nous pouvons utiliser l'algorithme de LZW qui utilise un dictionnaire. Il est même possible d'utiliser Huffman et LZW l'un après l'autre.

En septembre dernier, une équipe travaillant pour Google a sorti l'algorithme Brotli qui est basé sur une version modifiée de LZ77, le codage de Huffman et un modelage du contexte de second ordre. Cet algorithme est 20% plus performant dans son taux de compression que les algorithmes de compression utilisés jusqu'ici.