HAI809I — Projet Image et Compression
Parcours Imagine 2024-2025

Mathis Duban (mathis.duban@etu.umontpellier.fr)
Paul Deligne (paul.deligne@etu.umontpellier.fr)

March 23, 2025

# Compression basée super-pixels

Compte rendu du 23 mars 2025

# 1 Travail déposé dans le dépot GitHub

Durant cette semaine de travail, de nouveaux éléments ont été déposés dans différents dossiers :

- Le dossier **Image** contient les images d'entrées et de sorties utilisées pour l'algorithme.
- Le dossier **Code** contient tout le code (fichiers .cpp/.h)

Voici l'adresse de notre repository GitHub contenant tous les éléments actuels du projet:

`https://github.com/Akkuun/Super-Pixel-Project`

# 2 Travail effectué cette semaine

Le travail de cette semaine a été dédié à implémenter une méthode de compression de l'image modifiée avec notre méthode de Super pixel. Nous avons également implémenté une fonction qui calcule automatiquement le PSNR de l'image compressée.

## 2.1 Travail effectué auparavant

A la fin de la semaine dernière, nous avions terminé d'implémenter l'algorithme SLICC ainsi que les différentes optimisations pour permettre de calculer l'image transformée plus rapidement. Cependant, nous n'avions pas encore de méthode de compression.

## 2.2 Compression de l'image Super Pixel

Dans cette partie, nous allons discuter de la fin de l'implémentation d'une méthode pour la compression de l'image après l'algorithme SLICC.

Notre méthode choisie pour la compression fonctionne de la manière suivante : On transforme notre image super pixel au format RGB en une image au format YUV où Y est la luminance et U et V la chrominance. Pour la luminance, on ne modifie rien, mais pour la chrominance, on réduit le nombre de bits sur lesquels sont codées les valeurs de chrominance. Le nombre de bits est donné en paramètre par l'utilisateur lors de l'appel du code.

Voici le résultat que l'on obtient après la compression et décompression pour différentes valeurs de quantification sur l'image LezardHD ($3456 \times 5184$) avec k=5000 et m=20:

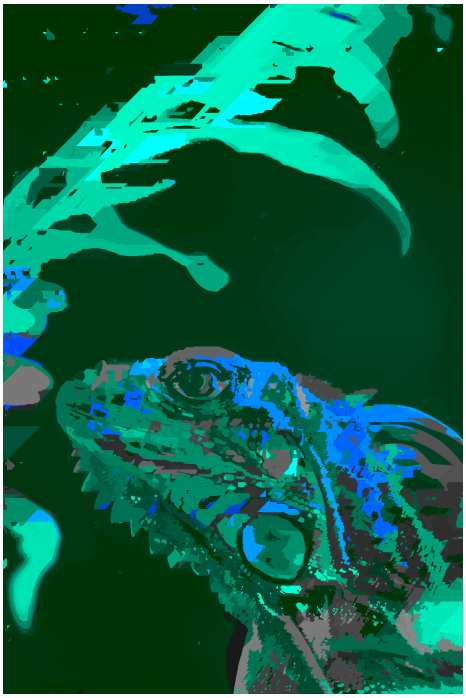

Figure 1: LezardHD après SLICC et 1 bit pour la chrominance

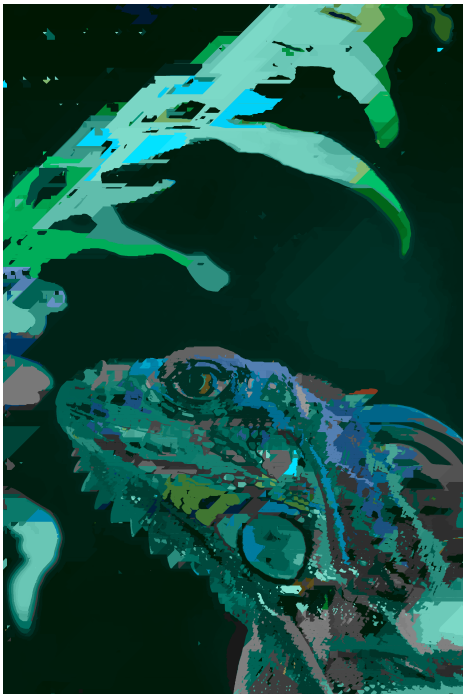

Figure 2: LezardHD après SLICC et 3 bit pour la chrominance

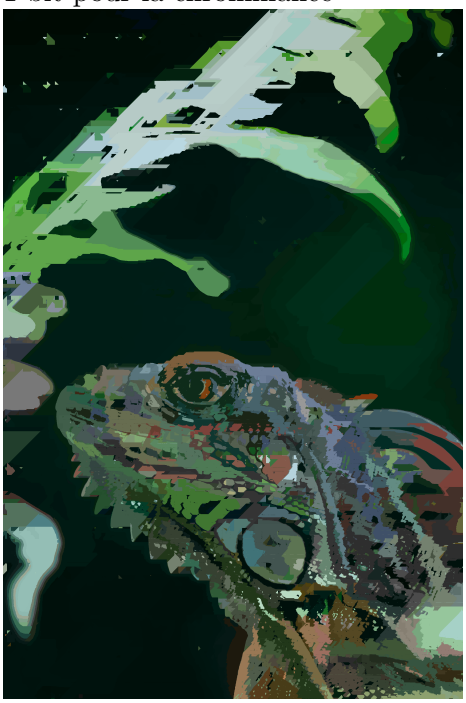

Figure 3: LezardHD après SLICC et 5 bit pour la chrominance

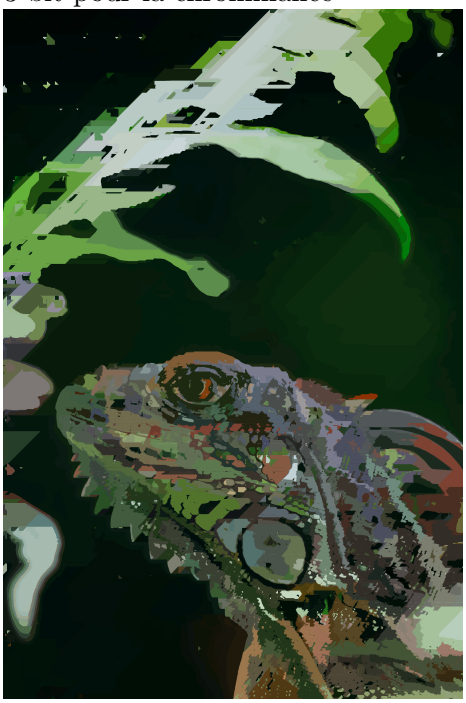

Figure 4: LezardHD après SLICC et 8 bit pour la chrominance

Enfin nous avons implémenté une fonction pour calculer le PSNR entre l'image de départ et l'image compressée ainsi qu'un programme qui calcule la courbe de distorsion en fonction du nombre de bits où est codée la chrominance :

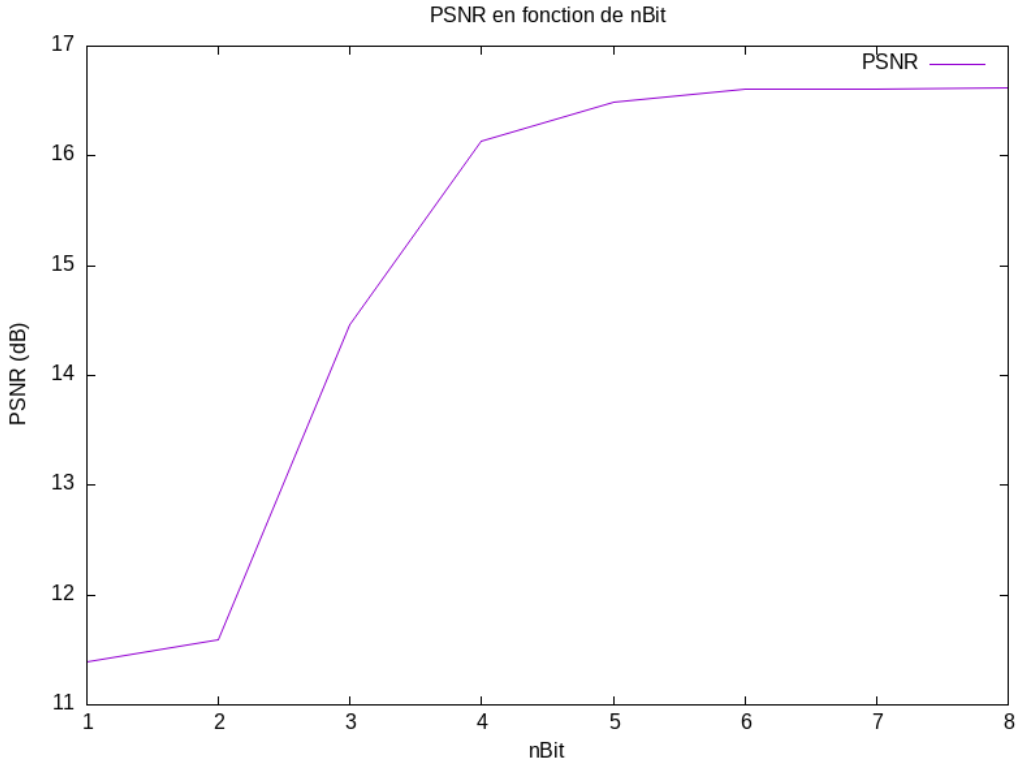


Figure 5: Courbe du PSNR de LézardHD en fonction du nombre de bits pour la chrominance. On peut remarquer que le meilleur choix en terme de comprimis entre le PSNR et le taux de compression serait 4 ou 5 bit pour la chrominance.

Enfin nous avons tracé une courbe où l'on calcule le PSNR entre l'image de départ et l'image compressée en fonction de k et m. La figure suivante a été réalisée à partir de notre image de lézard pour un k allant de 10 000 à 100 000 et un m allant de 10 à 50. L'axe des abscisses varie en fonction de k et celui des ordonnées en fonction du PSNR ainsi qu'une courbe pour chaque valeur de m.

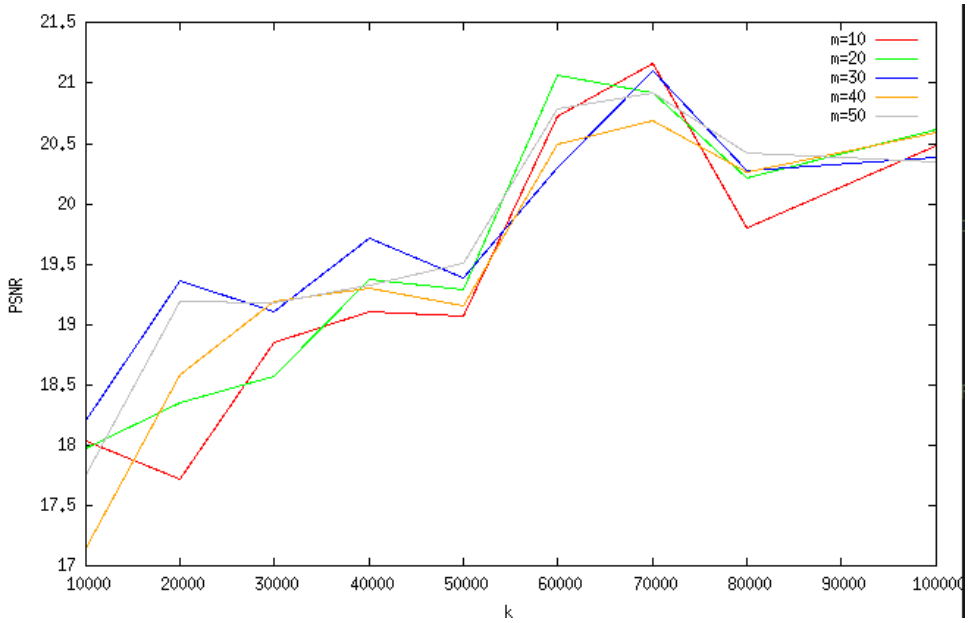


Figure 6: Courbe du PSNR de LézardHD en fonction de k (nombre de clusters) et de m (la résolution spatiale). On peut observer que la valeur de m pour laquelle on a le meilleurs PSNR est en général 30 ou 50 et le meilleur a été trouvé pour k=60 000.

# 3 Travail à venir

Pour le travail à venir, nous allons essayer d'implémenter une autre méthode pour calculer les superpixels ainsi que de chercher d'autres méthodes de compression.

# 4 Références

# References

[1] Bibilothèque d'image Unsplash,

https://unsplash.com/fr/s/photos/reptile

[2] jflalonde,

http://vision.gel.ulaval.ca/~jflalonde/cours/4105/h17/tps/results/projet/111063028/index.html *Segmentation d'images en superpixels via SLIC*

[3] jflalonde, epfl

https://www.epfl.ch/labs/ivrl/research/slic-superpixels/#SLICO *SLIC Superpixels*