

---

## TP 1 - Vision artificielle et traitement des images

---

Algorithme de detection



Benoît Faure FAUB20060100  
Basil Courbayre COUB30119800  
Ali Ghammaz GHAA27070100  
Novembre 2023

## Table des matières

|          |  |           |
|----------|--|-----------|
| <b>1</b> | <b>Introduction</b>                    | <b>1</b>  |
| <b>2</b> | <b>Traitements</b>                     | <b>1</b>  |
| <b>3</b> | <b>Extraction de contours</b>          | <b>1</b>  |
| <b>4</b> | <b>Watershed</b>                       | <b>4</b>  |
| 4.1      | Érosions . . . . .                     | 4         |
| 4.2      | Transformation par distances . . . . . | 4         |
| <b>5</b> | <b>SLIC</b>                            | <b>5</b>  |
| 5.1      | Feature Extraction . . . . .           | 5         |
| 5.2      | Clustering . . . . .                   | 5         |
| 5.3      | Conclusion . . . . .                   | 9         |
| <b>6</b> | <b>Conclusion</b>                      | <b>9</b>  |
| <b>7</b> | <b>Références</b>                      | <b>11</b> |

## Table des figures

|   |  |    |
|---|--|----|
| 1 | Differents type de traitements sur l'image . . . . .                           | 2  |
| 2 | Délimitation de l'arrière-plan. . . . .  | 3  |
| 3 | Extraction des bords . . . . .   | 3  |
| 4 | Opération de calcul de distance des pixels environnant . . .                   | 5  |
| 5 | Watershed sur image LAB . . . . .  | 6  |
| 6 | SLIC segmentation . . . . .  | 7  |
| 7 | Pixel clustering . . . . .   | 8  |
| 8 | SLIC superpixels and clustering by neighborhood and feature distance . . . . . | 10 |

## 1 Introduction

Le but de ce projet était de faire de la détection de grains de minéraux dans une image pour réaliser des opérations dessus. Pour ce faire, nous avions à notre disposition 9 images représentant des minéraux de couleurs, tailles et formes différentes sur fond noir, et nous devions trouver un moyen de détecter les différents types de minéraux. Ainsi, le travail consistait en deux tâches. La première était une étape de segmentation, où nous devions trouver un moyen de détecter et délimiter les minéraux dans l'image, ce à l'aide de divers filtres et transformations de l'image d'origine. La seconde étape consistait en l'évaluation de chaque minéral en faisant la moyenne de chacun des pixels dans le minéral, et ce pour chaque canal.

Ce rapport détaille les étapes du traitement d'image mises en œuvre pour relever ces défis, les méthodes et algorithmes employés, ainsi que les résultats obtenus.

## 2 Traitements

Avant de procéder à la segmentation, il était nécessaire d'identifier quelles images sources pourraient être intéressantes. Nous avons examiné les images dans les espaces de couleur HSV (1.d), Lab (1.e) et XYZ (1.f), ainsi qu'après avoir équilibré les histogrammes des canaux RGB (1.c). Les quatre traitements semblaient être discriminants, mais nous avons commencé par tester l'égalisation de l'histogramme sur l'image Lab. Les résultats sont présentés dans la figure 1. De plus, nous avons également essayé la méthode CLAHE, cependant, nous avons constaté que la différence n'était pas significative.

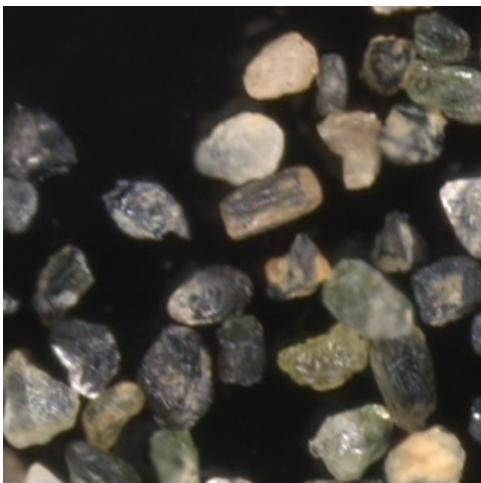
Nous avons dû extraire l'arrière-plan. Pour ce faire, nous avons examiné l'histogramme des valeurs grises (figure 2.c) et constaté que les valeurs sombres s'arrêtent à 50. En conséquence, nous avons appliqué une limite (threshold) de 50 pour extraire les images qui ne font pas partie de l'arrière-plan (figure 2.b).

## 3 Extraction de contours

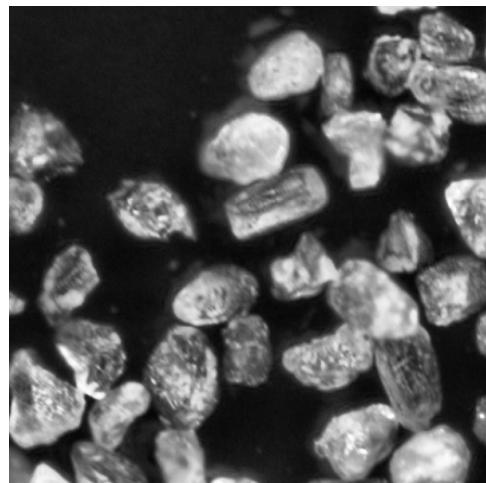
Ensuite, nous avons examiné la manière d'extraire les contours des différents grains afin de déterminer si nous pouvions obtenir des informations supplémentaires manuellement.

Nous avons d'abord essayé une méthode consistant à effectuer une détection de contours en floutant l'image à l'aide d'un filtre moyen de taille 100 x 100, puis en sous-trayant l'image floutée de l'image originale. Pour améliorer la qualité des contours, nous avons appliqué une opération de seuillage à la différence d'images. Ensuite, nous avons éliminé l'arrière-plan de l'image (voir la figure 3.a). Malgré cela, un remplissage (voir la figure 3.b) a été appliqué, mais les résultats n'étaient pas satisfaisants.

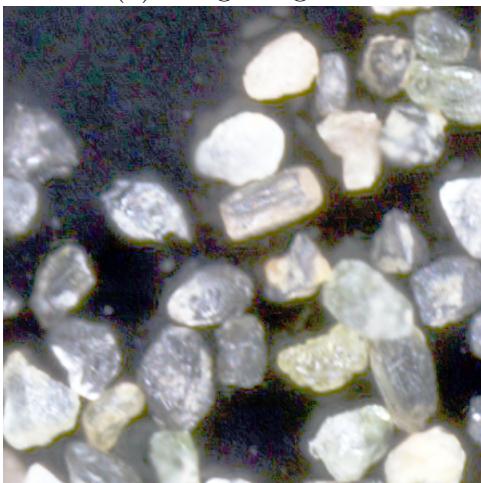
Face à ces résultats peu concluants, nous avons opté pour une autre approche de détection de contours : la méthode de détection des contours Canny. Comme illustré dans la figure 3.c, les contours détectés se sont avérés beaucoup plus complexes, voire trop. Une fois de plus, le remplissage (voir la figure 3.d) a produit des résultats de qualité médiocre.



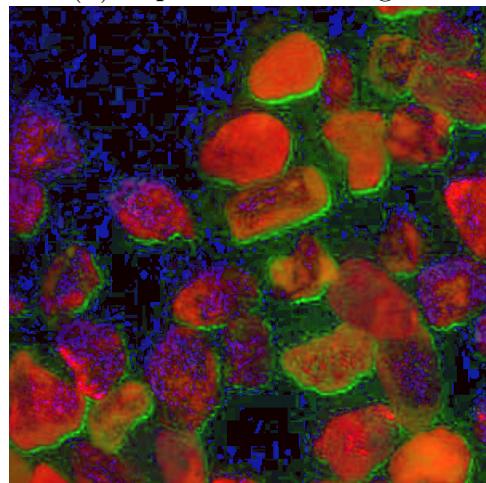
(a) Image originale



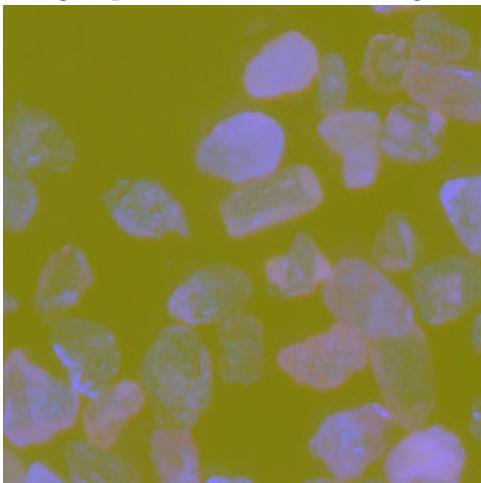
(b) Espace de couleur gris



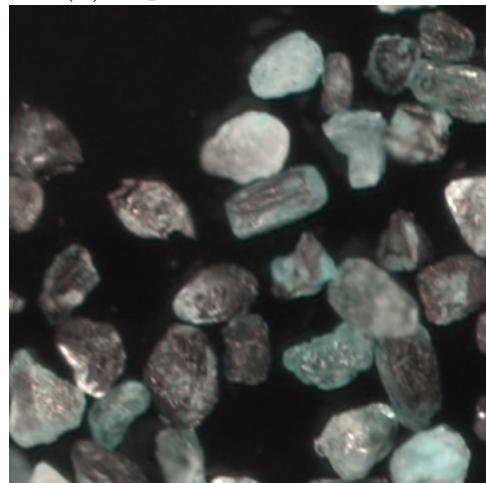
(c) Image après balance des histogrammes



(d) Espace de couleur HSV

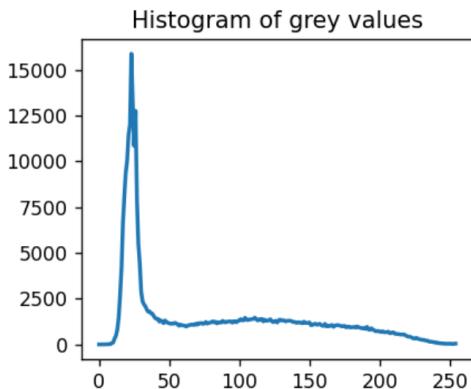


(e) Espace de couleur LAB

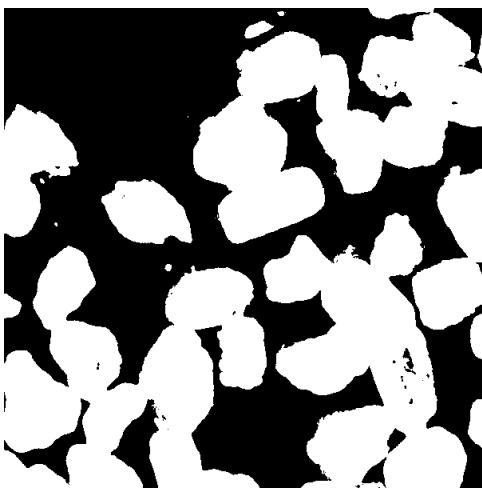


(f) Espace de couleur XYZ

FIGURE 1 – Differents type de traitements sur l'image

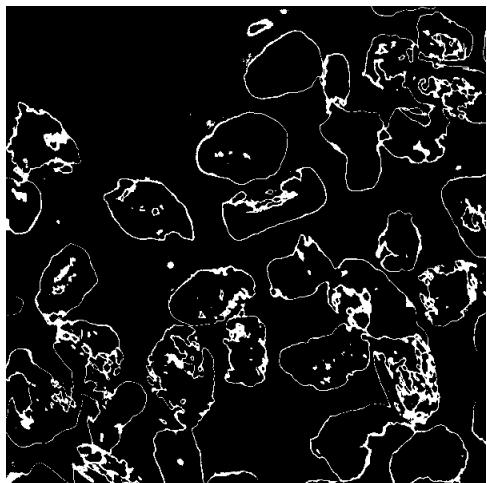


(a) Histogramme des valeurs grises

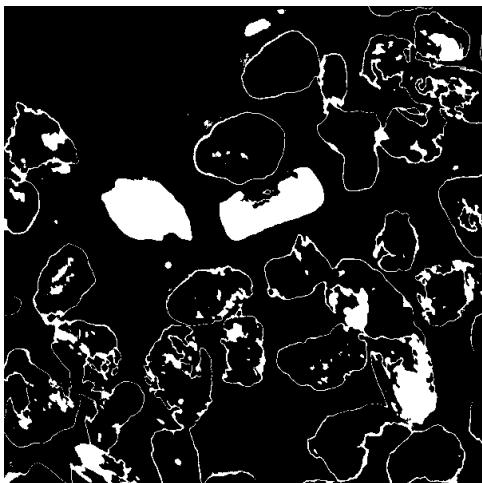


(b) Threshold

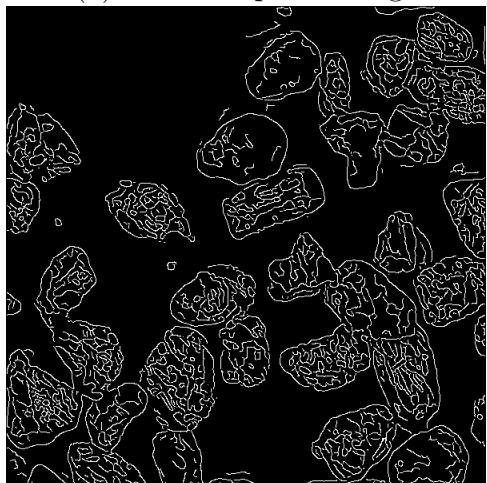
FIGURE 2 – Délimitation de l'arrière-plan.



(a) Méthode par floutage



(b) Méthode par floutage - remplis



(c) Méthode Canny



(d) Méthode Canny - remplis

FIGURE 3 – Extraction des bords

## 4 Watershed

Ensuite, nous avons souhaité tester l'algorithme Watershed. Cet algorithme simule l'écoulement d'un liquide sur l'image en partant de points d'origine marqués tout en évitant les surfaces de l'arrière-plan. Pour utiliser cet algorithme, il est nécessaire de définir un arrière-plan, que nous avons utilisé comme précédemment déclaré lors du pré-traitement de l'image et élargie par une opération de dilatation. De plus, il est essentiel de définir les points d'origine de l'écoulement. Nous avions l'intention d'utiliser une version érodée par une opération morphologique de l'extraction de bords remplis. Cependant, les résultats de ces remplissages étaient insatisfaisants. Par conséquent, nous avons essayé les méthodes suivantes.

### 4.1 Érosions

La première méthode que nous avons utilisée pour définir les points d'origine a été de simplement réduire considérablement le foreground défini par l'arrière-plan. Cependant, cette approche n'était pas optimale car elle ne permettait pas de mettre en évidence les séparations entre les objets. Par conséquent, nous avons décidé d'utiliser l'extraction des contours définis précédemment, que nous avons mis en œuvre pour créer les bordures et, par conséquent, effectuer le remplissage en utilisant la définition de l'arrière-plan. Ensuite, nous avons réalisé une opération d'érosion pour réduire l'origine de l'écoulement. Cependant, les résultats n'étaient pas satisfaisants.

### 4.2 Transformation par distances

En prenant l'arrière-plan, nous nous sommes définis une zone sûre du background en élargissant chaque minéral (dilatation avec un noyau de  $3 \times 3$ , 1 itération). Cela nous a permis d'avoir une idée des zones où nous étions sûrs de ne pas avoir de minéraux. Après ces opérations, nous avons utilisé la fonction `distanceTransform` pour mettre en valeur les zones d'origine des différents sédiments afin d'essayer de les séparer individuellement en un point (figure 4.a). Ensuite, nous avons appliqué un seuil à 40% de la valeur maximale de cette carte de distance, ce qui nous a permis d'isoler ces points d'origine (figure 4.b).

Nous avons appliqué l'algorithme à une version de l'image d'origine, et nous avons constaté que la version en espace de couleur LAB fonctionne le mieux. Les résultats de la segmentation sont présentés dans la figure 5. On peut remarquer que les contours (en bleu) sont bien définis, bien que non parfaits. Il est fréquent de constater que certains minéraux ont fusionné. Cela est principalement dû au fait que certaines zones de foreground sûres de minéraux différents se chevauchent. En conséquence, l'algorithme Watershed suppose que ces minéraux appartiennent à une seule structure, ce qui n'est pas le cas. Nous avons réalisé diverses expérimentations avec les opérations morphologiques sur le foreground, le choix des seuils, les espaces de couleurs et les traitements, mais cela n'a pas amélioré les résultats.

Nous pensions que les problèmes venaient du fait que certains points d'origine n'étaient pas suffisamment lumineux. Nous avons essayé de corriger cela en utilisant un flou gaussien, mais cela n'a pas été suffisant.

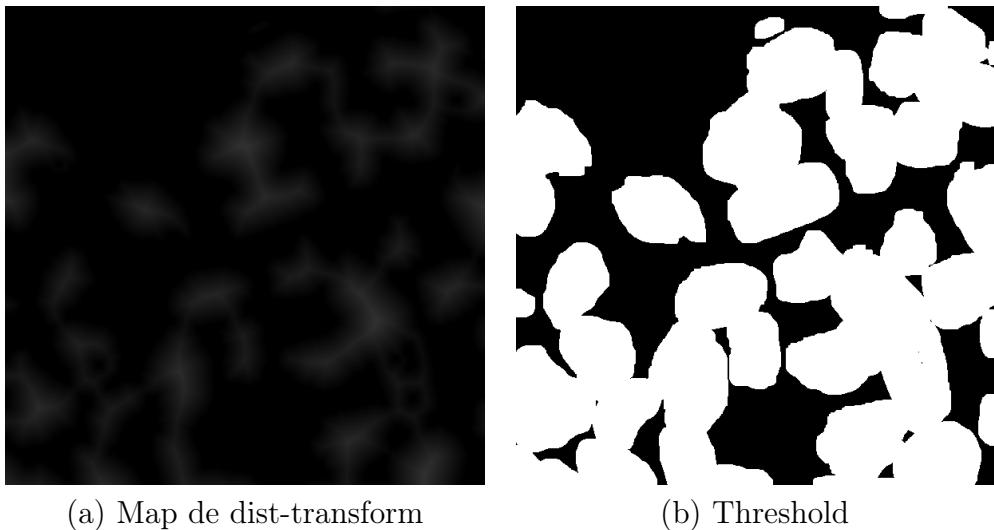


FIGURE 4 – Operation de calcul de distance des pixels environnant

## 5 SLIC

Nous nous sommes basés sur votre article sur le même sujet pour cette partie [1]. La première étape a consisté à segmenter l'image en régions d'intérêt en utilisant l'algorithme SLIC. Une fois l'arrière-plan appliqué pour supprimer les segmentations non pertinentes, nous obtenons la figure 7. Nous remarquons que la figure est divisée en plusieurs groupes de pixels (superpixels) que nous devrons combiner.

### 5.1 Feature Extraction

L'article propose différentes métriques à utiliser, et le code explique comment les implémenter. Les métriques tirées des images prétraitées sont les suivantes : la moyenne (mean), l'écart type (std), l'asymétrie (skew), le kurtosis (kurt), ainsi que les valeurs maximales de l'histogramme. Nous avons également examiné le degré de luminosité et effectué une classification préliminaire par pixels. Cette classification provient de l'image floutée que nous avons soumise à l'algorithme KMeans pour effectuer un premier regroupement, comme illustré dans la figure 10. Nous avons ensuite retenu la classe la plus représentative. Toutes les caractéristiques étaient normalisées. Pour remédier au problème potentiel selon lequel KMeans ne donnerait pas nécessairement des identifiants en fonction de la distance entre les classes, nous avons essayé d'utiliser l'ACP pour le regroupement, mais les résultats n'ont pas été concluants.

Nous avons également essayé d'examiner la direction générale du gradient pour aider l'algorithme à différencier les pierres en fonction de leurs formes, mais cela ne s'est pas avéré utile.

### 5.2 Clustering

Je ne crois pas avoir vu que l'article explique comment effectuer la segmentation finale de l'image, c'est-à-dire comment combiner les superpixels. Par conséquent, nous avons décidé d'effectuer un clustering des superpixels par apprentissage non supervisé. Nous

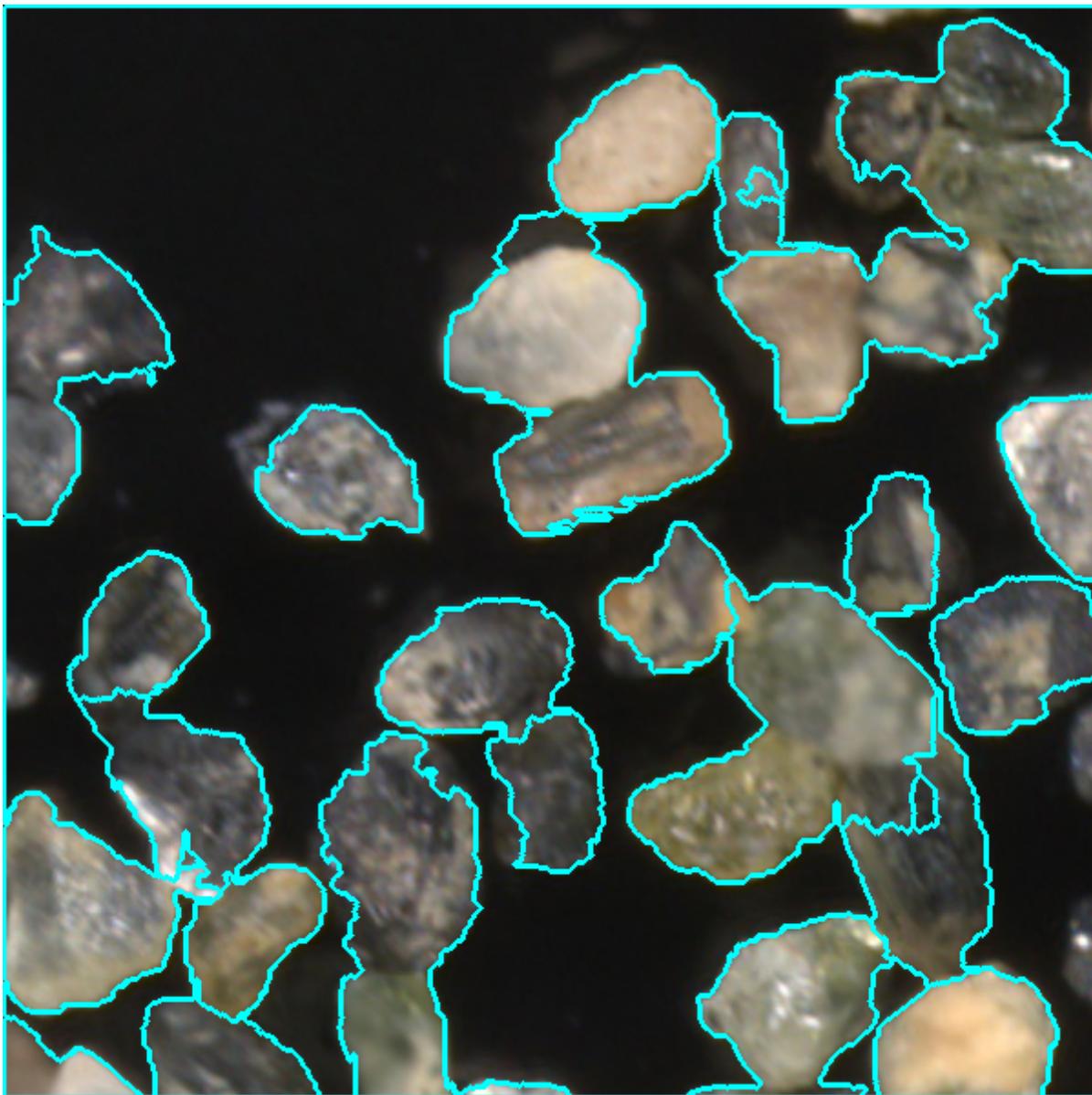


FIGURE 5 – Watershed sur image LAB

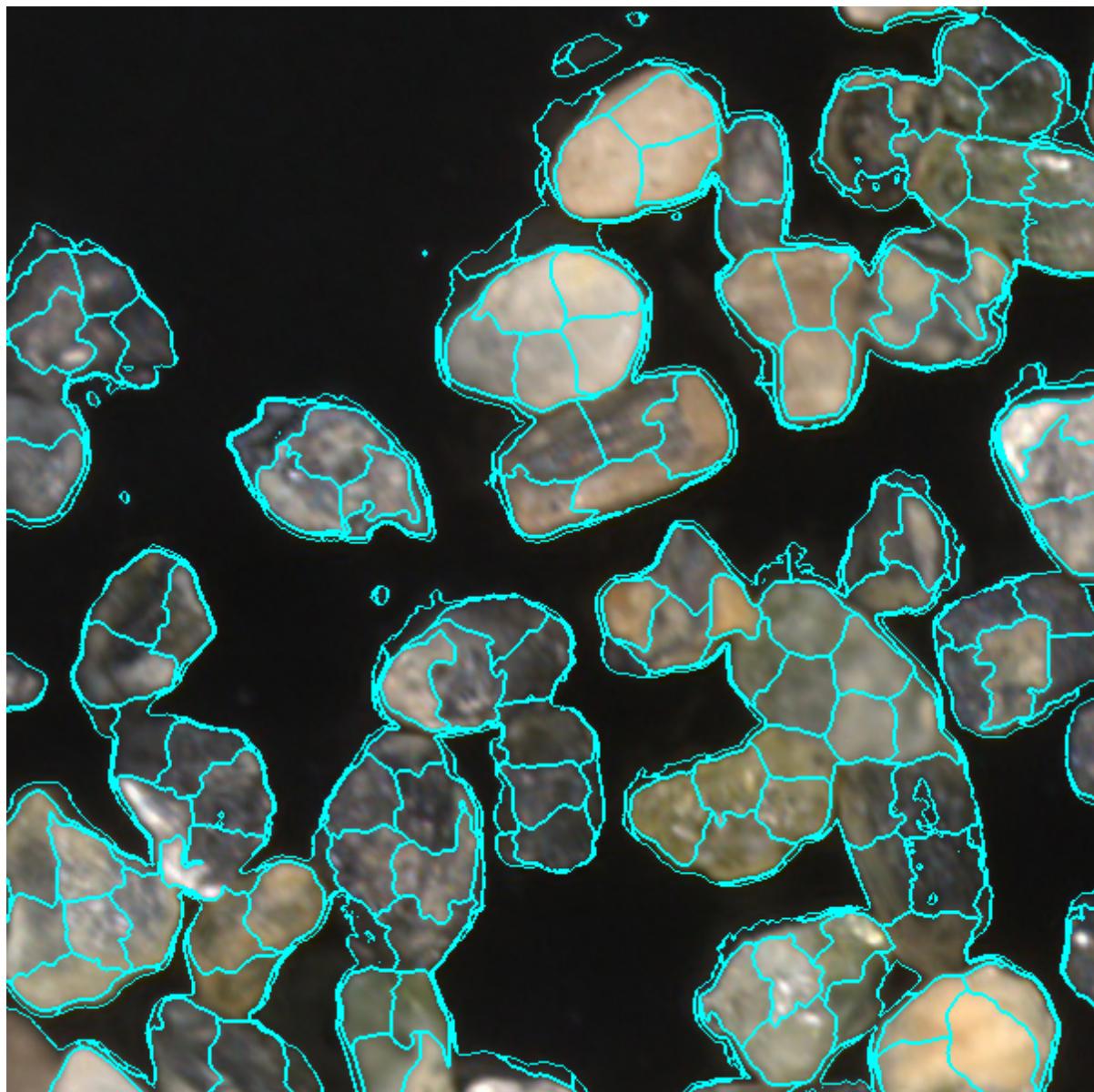


FIGURE 6 – SLIC segmentation

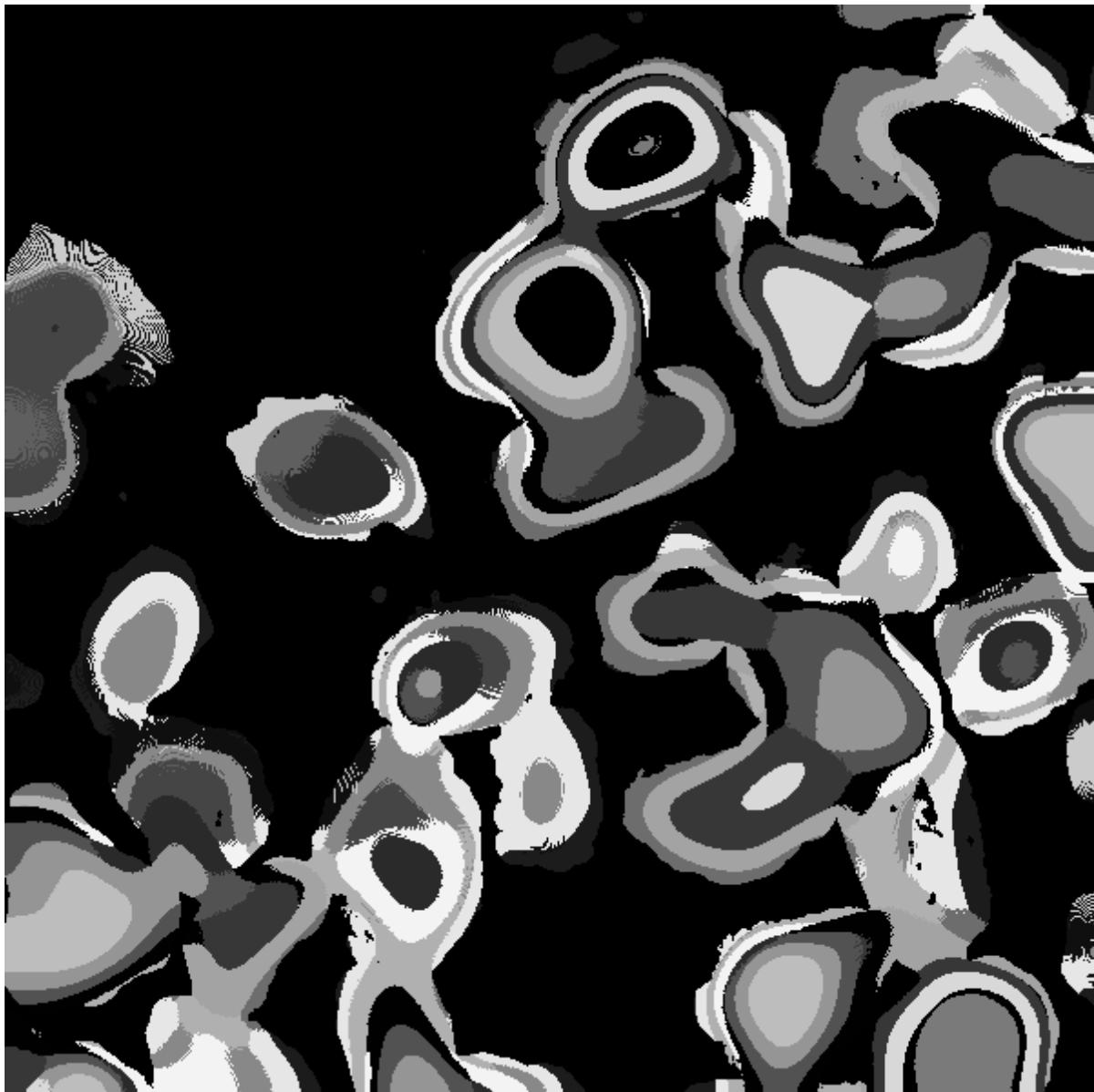


FIGURE 7 – Pixel clustering

avons testé KMeans, BisectingKMeans et AffinityPropagation, et avons constaté que AffinityPropagation donnait les meilleurs résultats en termes de clustering. Cependant, le problème était que, en fonction des caractéristiques et du nombre de classes, certains grains se regroupaient trop facilement, tandis que d'autres restaient séparés en une multitude de fragments. Nous avons essayé d'appliquer le clustering de manière itérative, mais cela avait le même effet de réduire le nombre de classes à créer.

Nous avons également essayé de combiner les superpixels manuellement afin de mieux exploiter les données spatiales. Pour ce faire, nous effectuons des dilatations des masques des superpixels que nous superposons pour vérifier s'ils étaient voisins. Ensuite, nous les avons combinés en examinant les distances entre les superpixels et en les fusionnant si la distance était suffisamment faible. Les résultats étaient plus satisfaisants.

### 5.3 Conclusion

Nous avons testé de nombreuses combinaisons de caractéristiques et de prétraitements, et nous avons conclu qu'une segmentation basée sur une image XYZ avec des caractéristiques extraites des prétraitements XYZ, HSV et équilibrage d'histogrammes était suffisante. Les résultats ne sont pas les plus satisfaisants, mais ils sont meilleurs que les méthodes que nous avions essayées auparavant. La segmentation finale est présentée dans la figure 8.

## 6 Conclusion

En conclusion, l'objectif de ce TP était de détecter et caractériser des minéraux dans des images complexes. Nous avons exploré diverses techniques de segmentation et d'extraction de caractéristiques pour atteindre cet objectif. Au cours de cette tâche, nous avons rencontré divers défis, notamment la similarité entre les grains et la difficulté à représenter les séparations entre les grains qui proviennent de leur structure.

Nous avons également essayé d'autres méthodes comme Cuts et Spectral Clustering, mais une fois de plus, les résultats n'ont pas été satisfaisants. La méthode finale utilise SLIC avec la moyenne, l'écart type, l'asymétrie, le kurtosis, le degré de luminance et le maximum des histogrammes comme métriques, ainsi que les identifiants extraits du clustering par pixel. Le tout est ensuite regroupé en fonction de la proximité et de la distance entre les métriques de chaque superpixel.

Dans l'ensemble, ce TP nous a permis d'explorer de nombreuses techniques de traitement d'images, de segmentation et de clustering. Il souligne également l'importance de l'ajustement minutieux des paramètres en fonction des caractéristiques des images et des besoins spécifiques de la tâche. Bien que des améliorations puissent encore être apportées.

Parmi ces améliorations, nous envisageons de revisiter l'algorithme Watershed et de retravailler la définition des points d'origine de l'écoulement, peut-être en améliorant l'extraction des bords. Nous pourrions également revoir notre méthode de clustering basée sur la proximité. Enfin, il serait nécessaire de mieux prendre en compte la forme des grains, peut-être en séparant les métriques de couleur et de forme.

Les commits montrent comment les différentes parties ont été implémentées : [git  
hub.com/BenoitFaure/TP\\_8INF804](https://github.com/BenoitFaure/TP_8INF804)

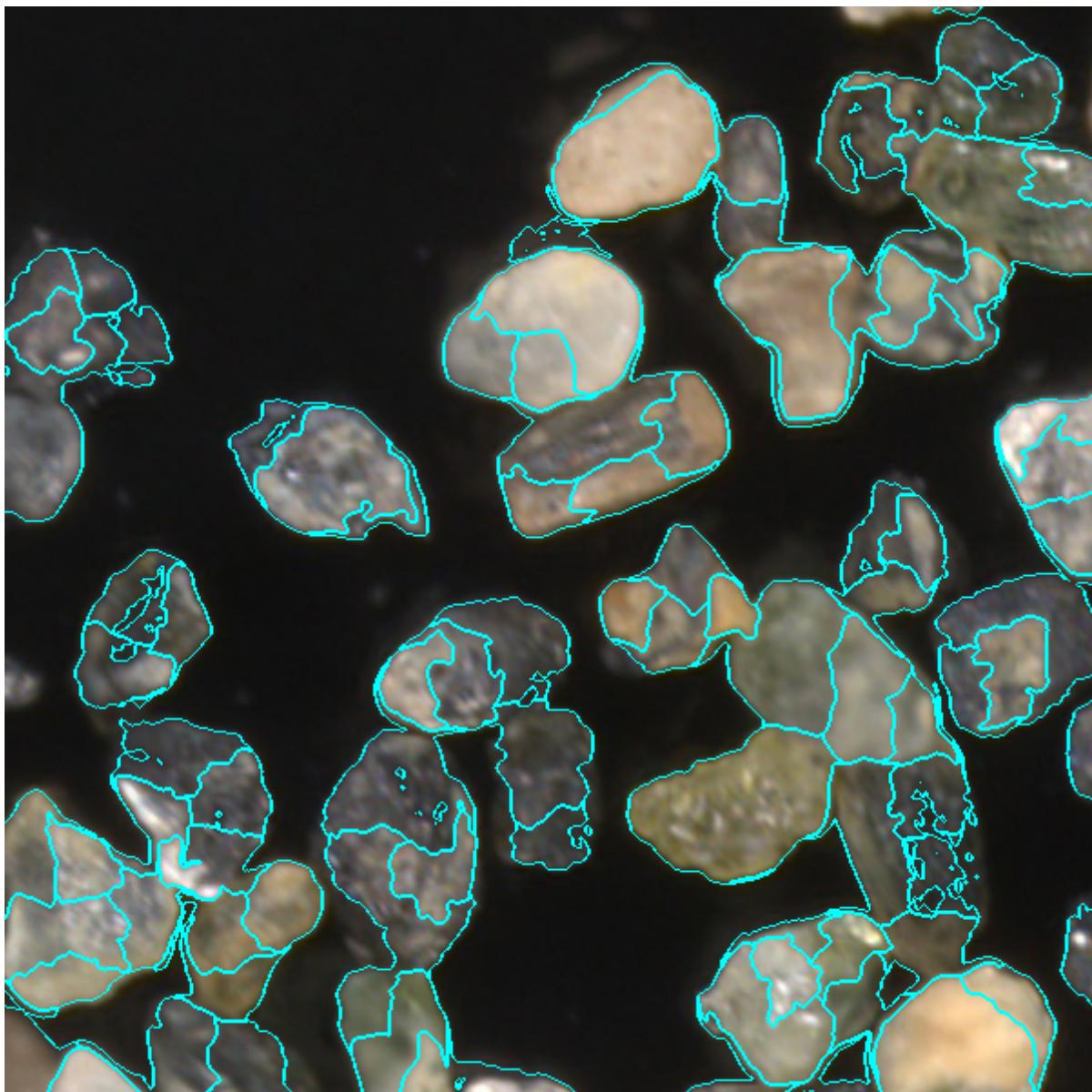


FIGURE 8 – SLIC superpixels and clustering by neighborhood and feature distance

## 7 Références

- [1] Julien Maitre, Kévin Bouchard, and L Paul Bédard. Mineral grains recognition using computer vision and machine learning. *Computers & Geosciences*, 130 :84–93, 2019.