TRAITEMENT D'IMAGES

Rapport Final de Travaux Pratiques

Réalisé par : **DEKPE Afi Elolo Gisèle**

Introduction

Ce document reporte notre travail pratique réalisé dans le cadre du cours de traitement d'images. Le programme issu de ce travail permet d'effectuer une chaîne complète de traitement sur les images fournies pour ce TP. Les étapes de cette chaîne sont :

- la phase de pré-segmentation : pour la préparation des images pour la phase segmentation;
- la phase de segmentation: pour l'identification des objets par des masques;
- la phase de post-segmentation : pour l'amélioration des régions détectées lors de la phase de segmentation .

Dans les lignes suivantes, nous allons présenter le fonctionnement du programme que nous avons mis en place et décrire les étapes précités.

1 Fonctionnement du programme

Dès l'exécution, le programme prend en paramètre l'image à traiter. Pour faire le fonctionner, il faut se placer dans le dossier du projet et exécuter les commandes suivantes :

a) pour la compilation :

```
make
```

b) pour l'exécution, le nom de l'exécutable suivi de l'image à traiter. Exemple pour l'image "*objets1*" :

```
./exe_afielologisele_p23rsc-tp objets1.jpg
```

c) pour les résultats, le programme s'exécute et affiche tous les résultats concernant l'image chargée en de petites fenêtres. Les images résultantes sont enregistrées dans le dossier du projet.

2 Réalisation de la chaîne de traitement d'une image

2.1 Phase de pré-traitement

Pour cette phase de préparation, les opérations que nous avons faites sont :

- **la modification de la taille des images** car de notre point de vue, elles étaient un peu trop grandes. De plus la réduction de la taille de l'image permet de réduire le temps de calcul. Pour cela, nous avons choisi une taille de **350 sur 350**;
- **l'application du contraste** car les images semblent être sombres. Nous avons appliqué une fonction sous la forme (g(i,j)=alpha . f(i,j)+bêta) qui prend en paramètre une valeur **alpha = 1.3** pour contrôler le contraste et une autre **bêta = 40** pour contrôler la luminosité. Les figures ci-dessous montrent les différences :

Figure 1: Objets1, Image originale

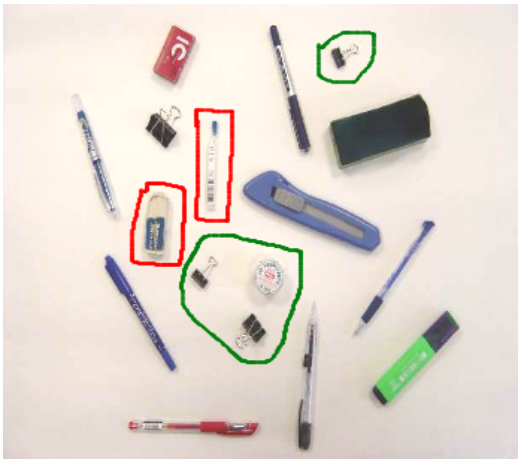

Figure 2: Objets1, Image contrastée

Figure 3: Objets2, Image originale

Figure 4: Objets2, Image contrastée

Figure 5: Objets3, Image originale

Figure 6: Objets3, Image contrastée

Figure 7: Objets4, Image originale

Figure 8: Objets4, Image contrastée

- **l'application du filtre Médian** ***(le meilleur des filtres)*** pour atténuer les bruits et lisser les images. Bien que les images soient claires, nous pensons qu'une application du filtre Médian apporterait une netteté de plus à l'image à segmenter. Avec la valeur de la ***longueur maximale du noyau*** $= 2$, cette opération n'a pas donné un grand changement à l'apparence des images mais nous pensons que c'est important de se rassurer de la netteté de l'image à segmenter;

Les images sont maintenant claires et nettes, nous pouvons passer à la phase de segmentation.

2.2 Phase de Segmentation

Pour ce faire, nous avons :

- **converti en niveau de gris** pour commencer. Ci-dessous les images :

Figure 9: Objets1, Image en niveau de gris

Figure 10: Objets2, Image en niveau de gris

Figure 11: Objets3, Image en niveau de gris

Figure 12: Objets4, Image en niveau de gris

- **appliqué l'algorithme d'OTSU** sur les images obtenues en niveau de gris, on obtient alors les images segmentées. La fonction *« threshold »* de opencv permettant d'appliquer l'**algorithme d'OTSU** est celle que nous avons utilisée. Nous avons utilisé l'argument *THRESH_BINARY_INV* pour avoir une binarisation inversée avec un fond noir. La segmentation permet de détecter tous ou la plupart des objets présents dans nos images (cela dépend du pré-traitement fait préalablement) et de les représenter par des masques. Le seuillage appliqué dans notre traitement

a permis de transformer une image en niveau de gris en une image dont les valeurs de pixels sont **1 ou 0**.

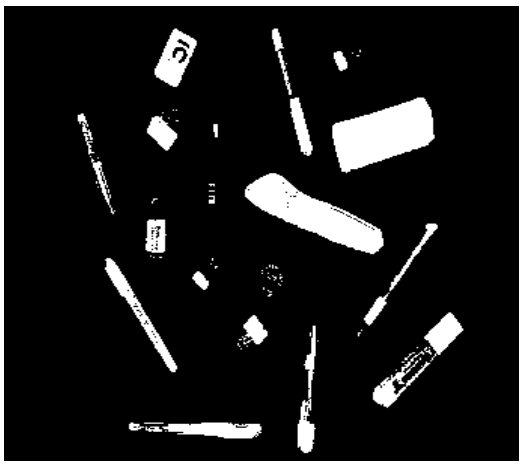

Figure 13: Objets1, Image segmentée

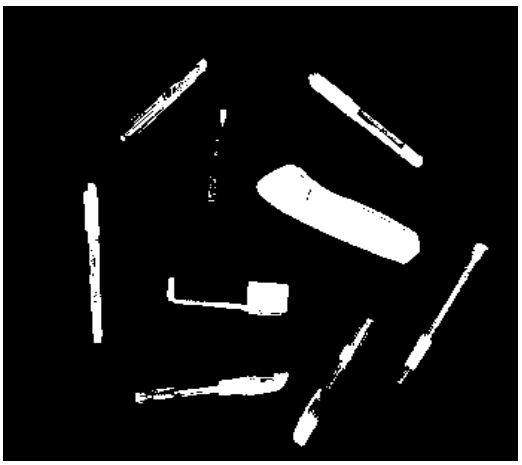

Figure 14: Objets2, Image segmentée

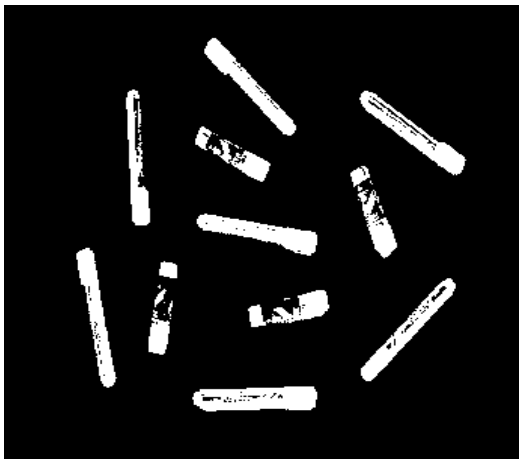

Figure 15: Objets3, Image segmentée

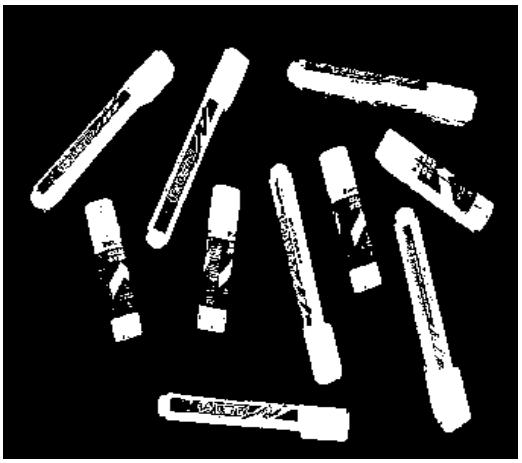

Figure 16: Objets4, Image segmentée

Analyse : les masques que nous avons obtenus montrent que la méthode d'OTSU s'est bien réalisée sur nos objets spécialement pour les **figures 13, 15, 16**. La **figure 13** montre les moindre détails sur les petits objets; contrairement à la **figure 14** qui montre juste les gros objets. Cela prouve qu'à chaque image correspond son traitement. Les zones d'ombres des objets sont intégrées aux régions. Les parties des objets de l'image dont les couleurs sont proches du fond sont considéré comme le fond.
Les images étant segmentées, nous pouvons passer à la phase de post-segmentation.

2.3 Phase de post-traitement

Pour finir, nous allons corriger les images. Nous avons :

- appliqué l'opérateur morphologique de **dilatation sur l'image** une fois, en passant par la formule de l'élément structurant et en utilisant la fonction de Opencv *dilate*.
- ensuite, appliqué l'opérateur morphologique d'**érosion** une fois, en passant par la formule de l'élément structurant et en utilisant la fonction *erode*.

Figure 17: Objets1, Image dilatée

Figure 18: Objets1, Image érodée

Figure 19: Objets2, Image dilatée

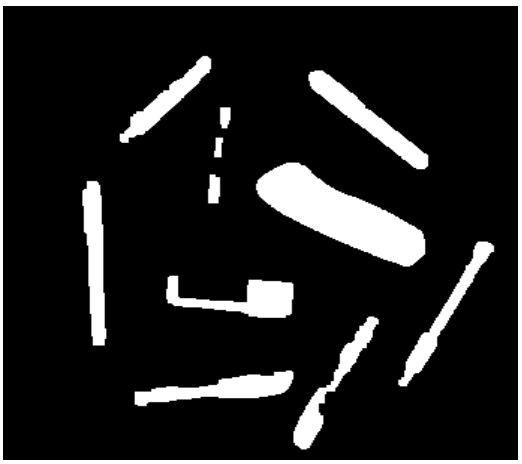

Figure 20: Objets2, Image érodée

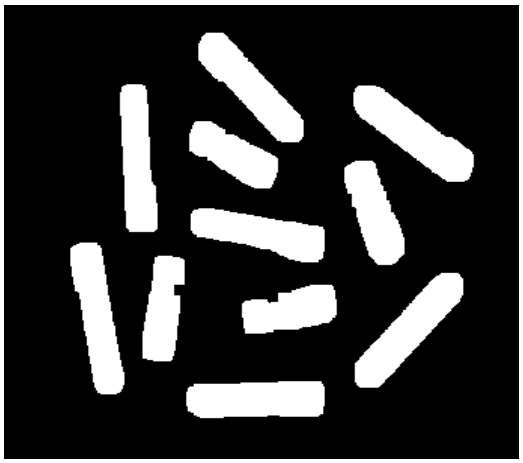

Figure 21: Objets3, Image dilatée

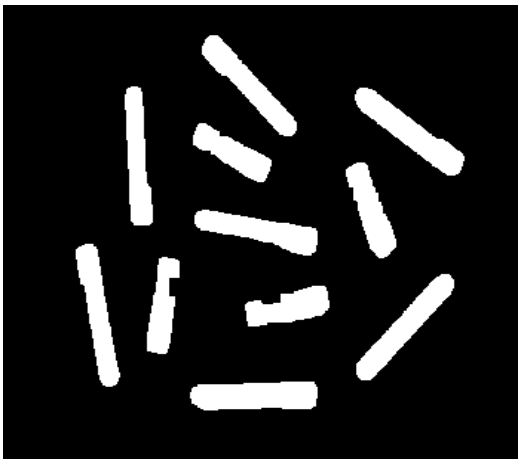

Figure 22: Objets3, Image érodée

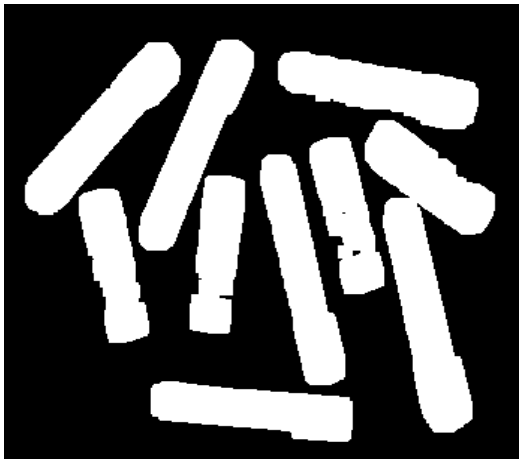

Figure 23: Objets4, Image dilatée

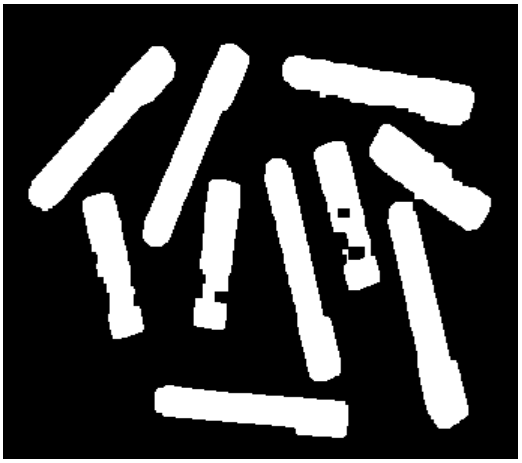

Figure 24: Objets4, Image érodée

Analyse : Nous avons appliqué ces opérateurs morphologiques (de type rectangulaire : $MORPH\_RECT$). Ainsi la dilatation a été faite pour retrouver l'unicité des objets qui se sont divisés en plusieurs parties lors de la segmentation. Et ensuite l'erosion pour retrouver la forme initiale des objets.

- recherché les contours pour pouvoir réaliser l'étiquetage. Ceci s'est réalisé grâce à la fonction **findContours** appliquée sur l'image érodée avec d'autres paramètres (voir code).
- ensuite étiqueté les régions détectées avec la fonction **drawContours**.
- au final, **coloré ces régions** avec des couleurs générées aléatoirement.

Le résultat final donne ceci :

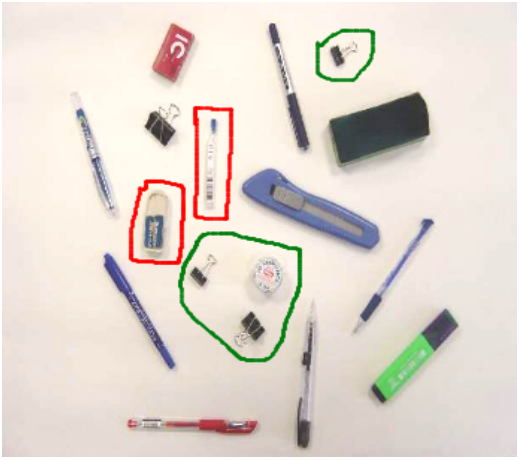

Figure 25: Objets1, Image contrastée

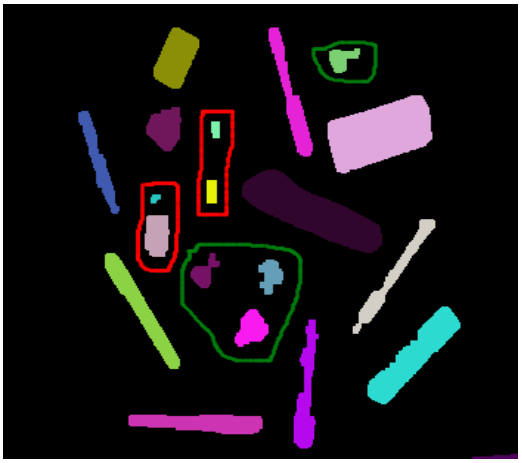

Figure 26: Objets1, Post-segmentation

Figure 27: Objets2, Image contrastée

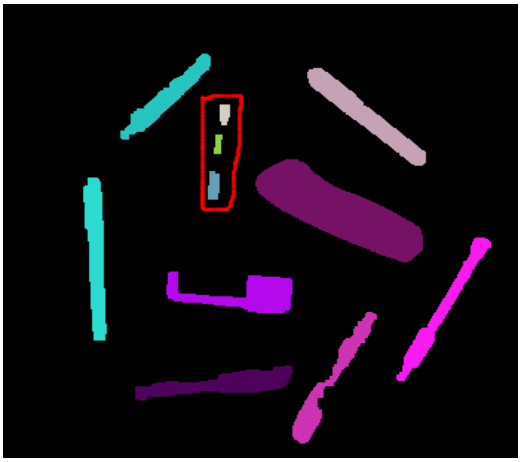

Figure 28: Objets2, Post-segmentation

Figure 29: Objets3, Image contrastée

Figure 30: Objets3, Post-segmentation

Figure 31: Objets4, Image contrastée

Figure 32: Objets4, Post-segmentation

Analyse : Des résultats obtenus, on peut dire que la chaîne de traitement diffère selon l'image.

Notons que les objets encadrés en rouge sur les **figures 25** et **27** correspondent à ceux encadrés en rouge sur les **figures 26** et **28**. Ce qui s'est passé à ce niveau est que ces objets possèdent des parties dont la couleur se rapproche du fond de l'image. En essayant d'augmenter les paramètres de la dilatation, cela affecte les autres objets de l'image qui se collent et forment un seul objet à la fin. Ce qui rendra probablement faux notre traitement.

Nous pensons que notre traitement a été efficace car on arrive à compter aussi les moindres petits objets **(encadrés en vert)**, **comme sur la figure 25** détecté sur la **figure 26**. Ceci grâce à une bonne réalisation de pré-traitement.

Conclusion

Ce rapport détaille les résultats de notre programme de chaîne complète de traitement d'images. Au niveau de la pré-segmentation, nous avons réduit la taille des images, appliqué un contraste puis lissé avec un filtre Median. Ensuite nous avons utilisé l'algorithme d'OTSU pour la phase de segmentation. Finalement nous avons appliqué une fermeture et/ou ouverture au niveau de la post-segmentation.

Finalement, les résultats obtenus sont à 90% satisfaisants. La limite est que notre programme ne peut pas traiter toutes les images de la même façon. La perspective de solution serait de reprendre notre programme, mais avec des paramètres différents surtout au

niveau de la dilatation et de l'érosion spécialement pour les images qui ne répondent pas au traitement actuelle.
Ce TP nous a permis de faire une révision générale sur les notions de traitements d'images appris en cours; au travers des recherches. Notons que tout ceci a été rendu possible grâce à une forte documentation sur **Opencv**.

Références

- Cours de Traitements d'images de Mme NGUYEN Thi Oanh
- https://docs.opencv.org/3.4/d3/dc1/tutorial_basic_linear_transform.html
- https://docs.opencv.org/2.4/doc/tutorials/imgproc/threshold/threshold.html
- https://docs.opencv.org/3.4/d7/d1c/tutorial_js_watershed.html