Le génie pour l'industrie

MTI-805 : COMPRÉHENSION DE L'IMAGE

OPTIMISATION DE LA SEGMENTATION PAR TURBOPIXELS EN UTILISANT LA MORPHOLOGIE MATHÉMATIQUE

PROFESSEUR : LUC DUONG

PRODUIT PAR : SAM GHAZAL & M'HAND KEDJAR
Code ÉTS : GHAS16087706 & KEDM09088002

DATE : 2016-04-22

Sommaire

1. Introduction

La segmentation d'images numériques occupe une grande place dans le domaine du traitement d'images. En quelques termes, le but de segmenter une image numérique consiste en la division de l'image en sous-segments, par rassemblement de pixels entre eux selon des critères prédéfinis, formant ainsi ce qu'on nomme superpixels, afin de tenter d'identifier des parties ou objets significatifs dans l'image. Ceci est très utile en imagerie médicale, entre autres, lorsqu'il est nécessaire d'analyser des images à l'intérieur d'un processus de diagnostic.

La plupart des algorithmes populaires de segmentation font partie d'au moins une des catégories suivantes :

- La segmentation fondée sur les régions (region-based segmentation) : À partir d'une première partition de l'image, il y a division ou regroupement de régions :
    - Croissance de régions (region-growing),
    - Décomposition/fusion (split and merge).
- La segmentation fondée sur les contours (edge-based segmentation) : Considère la forte probabilité qu'il existe une transition détectable de niveau d'intensités des pixels de différentes régions. Ces discontinuités dans les niveaux d'intensités sont le plus souvent détectées à l'aide du calcul de dérivée de premier ordre (gradient) de fonction d'intensité 2D, f(x,y), de l'image analysée.
- La segmentation fondée sur la classification ou le seuillage des pixels en fonction de leur intensité (classification ou thresholding) : Regroupement des pixels en diverses classes selon le rapport d'informations qu'elles entretiennent avec l'ensemble de l'image.
- Une combinaison d'au moins deux des trois catégories ci-dessus.

La segmentation par superpixels, dont TurboPixels, fait partie, de façon restreinte, de la catégorie de segmentations basées sur les régions. Elle vise une optimisation en tentant une simplification maximale de l'image tout en minimisant la sous-segmentation. C'est

un algorithme relativement récent [1] et très performant. Les superpixels portent plus d'information que les pixels et s'alignent mieux avec les bords de l'image que les patches rectangulaires. Les superpixels peuvent grandement améliorer la vitesse de traitement car leur nombre dans une image varie de 50 à 2000, comparé aux centaines de milliers de pixels. Un exemple d'une segmentation par superpixels est illustré en figure 1.

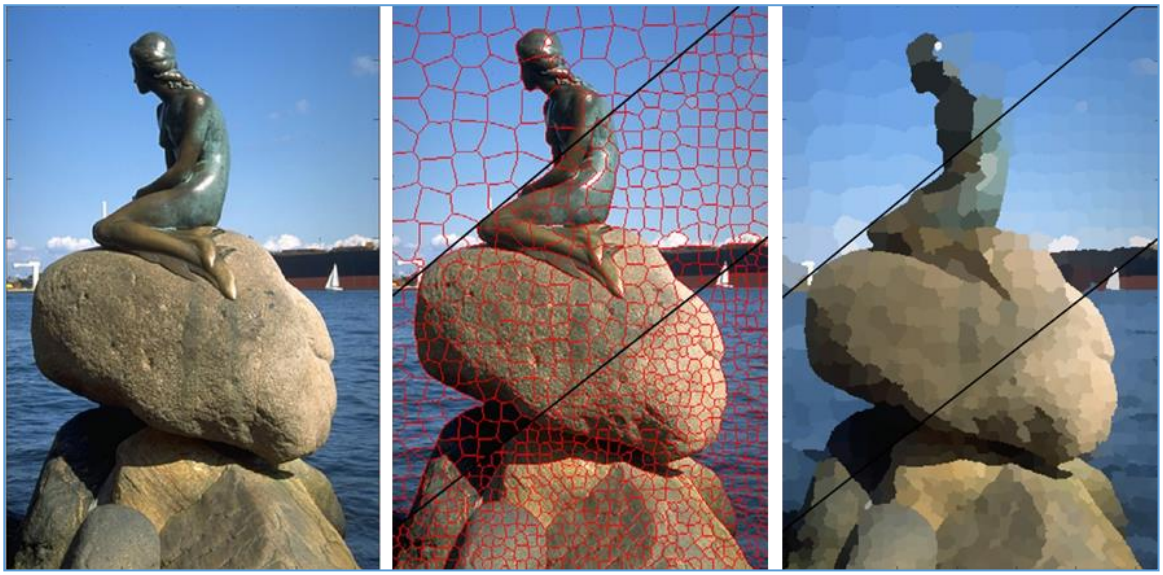

Figure 1: Exemple d'une segmentation par superpixels : la figure de gauche est l'image originale. Celle du milieu représente la segmentation par TurboPixels en utilisant N = 200 (en haut), N = 600 (milieu), et N = 1600 (bas) superpixels. La figure de droite représente la reconstruction de l'image où la couleur de chaque superpixel est remplacée par la moyenne des couleurs des pixels qui le composent.

Le terme superpixels a été introduit par Ren et Malik en 2003 [2], et ils sont depuis devenus extrêmement populaires dans plusieurs applications de la vision par ordinateur. Leurs avantages ont été analysés dans [3] et intégrés dans des applications comme la reconnaissance d'objets [4], la segmentation [5] et l'estimation de profondeur [6]. Les cotés négatifs dans l'utilisation de la segmentation par superpixels comme une étape de prétraitement sont l'effort de calcul pour générer les superpixels et, d'une façon plus importante, le risque de perte des bords de l'image en les plaçant à l'intérieur d'un superpixel.

1.1 Problématique

La segmentation d'image est l'un des problèmes les plus stimulants et aussi les plus complexes dans le domaine de la vision par ordinateur. Le but de la segmentation est le partitionnement de l'image en régions connectées, sans chevauchement, et en étant homogènes par rapport à certaines caractéristiques de l'image (couleur, texture, …). Dans l'image segmentée, les régions ont, contrairement aux pixels simples, de nombreuses caractéristiques intéressantes comme la forme, la texture, et ainsi de suite. Un être humain reconnait les objets de l'environnement en utilisant le système visuel et la segmentation des images en couleur.

La segmentation par superpixels est conçue pour produire une représentation plus simple et plus compacte pour une image, tout en gardant son sens sémantique intact. Dans cette optique, les principales propriétés recherchées dans un algorithme qui génère des superpixels sont :

- La faible complexité au niveau du temps de calcul.
- Le contrôle sur le choix du nombre de superpixels.
- L'adhérence des superpixels aux bords de l'image.
- L'obtention de superpixels compacts.
- Le faible nombre de paramètres nécessaires pour son implémentation.

Dans le but d'évaluer les performances d'un algorithme donné, nous devons définir une base de métriques. Dans le cas des superpixels, les deux mesures les plus souvent utilisées sont l'erreur de sous-segmentation et le rappel de contours. Ces deux mesures qui dépendent grandement de la segmentation Ground Truth ne sont pas définies d'une manière consistante à travers la littérature, et ne permettent pas à elles seules d'évaluer les performances d'un algorithme donné. Ainsi, la définition de métriques additionnelles et indépendantes du Ground Truth est nécessaire pour une évaluation plus complète des performances d'une segmentation par Superpixels.

1.2 Objectif

L'objectif de ce travail est d'utiliser des étapes de prétraitement morphologiques pour améliorer la segmentation par TurboPixels. L'évaluation sera portée en utilisant quatre métriques ; deux qui sont dépendantes du Ground Truth à savoir le rappel de contours et l'erreur de sous-segmentation, et les deux autres qui sont indépendantes du Ground Truth : la compacité et la variation expliquée.

Ce rapport est organisé comme suit : Après cette petite introduction, la section 2 traitera de la revue de littérature des algorithmes basés sur la segmentation par superpixels. Dans la section 3, nous allons décrire un peu plus en détail l'algorithme des TurboPixels et montrer comment la segmentation par superpixels est générée. Nous allons ensuite détailler les différentes métriques utilisées pour l'évaluation des performances des superpixels, et détailler la base de données d'images utilisée. Dans la section 4, nous expliquerons l'utilisation de la morphologie mathématique en tant qu'étape de prétraitement pour l'amélioration de la segmentation. Les résultats obtenus sont présentés et analysés dans la section 5, avec des exemples qualitatifs de la segmentation et de la reconstruction d'image, et quantitatifs au niveau des métriques de performance. Nous terminerons par une conclusion et des perspectives à venir.

2. Revue de la littérature

De nombreux auteurs ont étudié les superpixels depuis que le terme a été établi dans [2]. En particulier différents algorithmes de segmentation par superpixels ont été proposés, par exemple Ncuts [7], Local Variation [8], Watershed [9], MeanShift [10], TurboPixels [1] (l'objet de ce travail) et plus récemment SLIC [11], SEEDS [12].

L'algorithme des TurboPixels et celui basé sur Ncuts arrivent à produire des superpixels compacts. D'autres comme MeanShift, Watershed n'ont pas de contrainte de compacité. Il en résulte une sous-segmentation assez importante.

L'algorithme de segmentation par Watershed se base sur l'expansion de régions et évite la sur-segmentation par l'utilisation de certains marqueurs. Watershed est reconnu parmi les algorithmes de segmentation performants et rapides, que ce soit pour détection de contour ou segmentation basée sur les régions. Dans l'article [13], les auteurs présentent une façon d'effectuer la segmentation par Watershed, suite à une série d'opérations morphologiques, et démontrent une amélioration dans certaines des mesures de performance de segmentation, due à l'utilisation de la morphologie mathématique.

Nous proposerons également l'utilisation de certaines opérations morphologiques dans le but d'explorer leurs effets sur la segmentation par TurboPixels.

3. Mesures de performance des superpixels

3.1 Algorithme des TurboPixels

Les TurboPixels est un algorithme pour générer des superpixels développé par Levinshtein et al. [1]. Inspiré par la méthode des contours actifs, après avoir placé les centres des superpixels sur une grille régulière, les superpixels se forment sur la base d'un contour qui évolue. Le contour est implémenté en une méthode des surfaces de niveau (level set) de la fonction :

$$\Psi: \mathbb{R}^2 \times [0, \tau) \rightarrow \mathbb{R}^2 \quad (1)$$

Qui est ensuite évolué suivant la formule :

$$\Psi_\tau = -S \| \nabla \Psi \| \quad (2)$$

Où : $\nabla \Psi$ désigne le gradient de Ψ, et Ψ_τ la dérivée temporelle. Ici, la vitesse S décrit l'évolution future du contour. En pratique, Ψ serait la distance euclidienne signée et l'évolution est effectuée en utilisant une discrétisation du premier ordre. Le contour à l'itération (n+1) est donné par :

$$\Psi^{n+1} = \Psi^n - S_I S_B \| \nabla \Psi^n \| \triangle t \quad (3)$$

Chaque application de cette équation correspond à une étape dans le temps.

La vitesse S est divisée en deux composantes : S_I qui dépend du contenu de l'image, et S_B qui assure que les superpixels ne se chevauchent pas. D'une manière itérative, les superpixels évoluent en calculant S_I et S_B, et en appliquant l'équation (3). La procédure est résumée dans l'algorithme 1 :

```
fonction turbopixels(Image,Nombre de superpixels)
      Placer les centres des superpixels sur une grille régulière
      Initialiser  Ψ^0
      Répéter
            Calculer S_I et S_B
            Évoluer le contour en calculant Ψ^(n+1)
            Mettre à jour les pixels assignés
            n := n+1
      Jusqu'à ce que tous les pixels sont assignés
      Déduire la segmentation par superpixels S de Ψ
      Retourner S
```

Algorithme 1: Implémentation des TurboPixels

3.2 Métriques des Superpixels

3.2.1 Dépendantes du Ground Truth :

Pour l'évaluation de la qualité de la segmentation, nous nous focaliserons sur quatre métriques d'erreurs : Le rappel des contours, l'erreur de sous-segmentation, la compacité et la variation expliquée. Les deux premières sont dépendantes de la segmentation Ground Truth, alors que les deux dernières ne le sont pas.

3.2.2 Rappel des contours

Le rappel des contours est la fraction des contours du Ground Truth qui sont à une distance r d'au moins un bord d'un superpixel [14]. On utilise r = 0.0075*DI, avec DI : la diagonale de l'image. Ceci est pour rester consistant avec les mesures de performances des autres études faites avec la base de données de Berkeley.

Soit G la segmentation Ground Truth d'une image et S sa segmentation par superpixels.

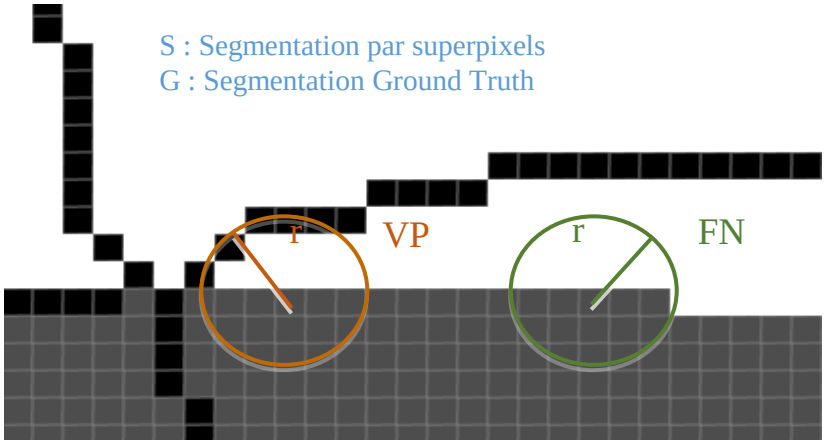

Figure 2: Rappel des contours

.

Le calcul du rappel des contours est fait comme suit :

- **Vrais Positifs (VP)** : Le nombre de pixels du contour de G pour lesquels il existe un pixel de contour de S à une distance r
- **Faux Négatifs (FN)** : Le nombre de pixels du contour de G pour lesquels il n'existe pas un pixel de contour de S à une distance r
- **Rappel des contours R(S,G)** :

$$R(S,G) = \frac{VP(S,G)}{VP(S,G) + FN(S,G)} \quad (4)$$

Comme le but à atteindre est d'avoir des superpixels qui adhèrent bien aux contours de l'image, une grande valeur de R(S,G) est recherchée.

3.2.3 Erreur de sous-segmentation

L'erreur de sous-segmentation mesure la fraction de pixels qui débordent de la segmentation causée par les superpixels par rapport à une segmentation manuelle. Un segment Ground Truth G divise un superpixel P en une partie P^+ et une partie P^-. Ceci est illustré à la figure 3.

Il existe différentes implémentations pour le calcul de cette métrique, dans [1], pour chaque segment S, une sommation est faite pour toutes les P^- qui chevauchent ce segment :

$$\sum_{S \in G} \frac{\sum_{P:P \bigcap S \neq \emptyset} |P^-|}{|S|} \tag{5}$$

Par exemple, dans la figure 3, ça serait : $\frac{|A^-| + |B^-| + |C^-|}{|S|}$

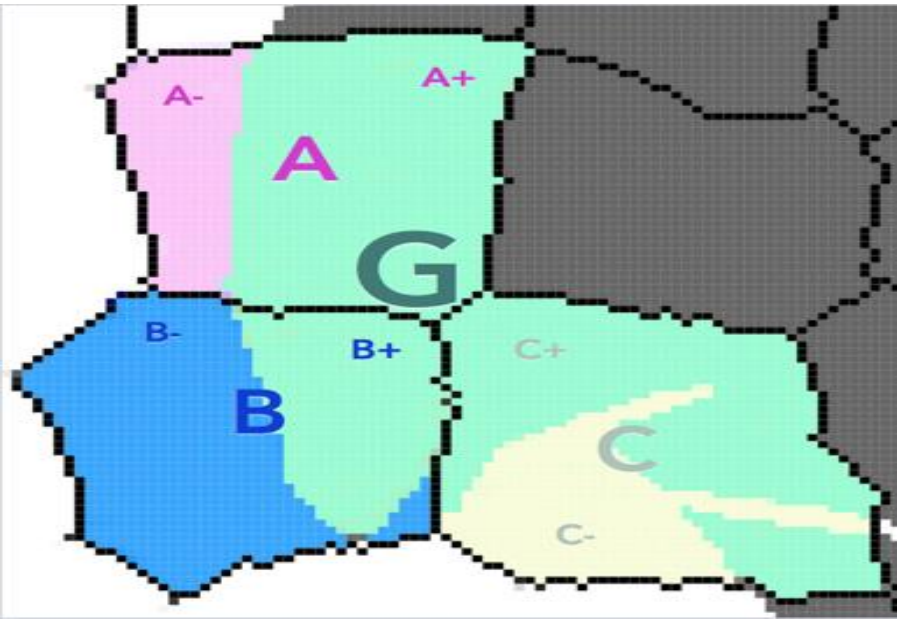

Figure 3: Sous-segmentation

Cependant, il y a un sérieux désavantage pour les larges superpixels qui ont seulement un petit chevauchement avec le segment Ground Truth. Ainsi, les auteurs de l'article [15] utilisent un modèle similaire, mais seulement les superpixels qui ont un chevauchement d'au moins de 5% avec le segment G sont considérés. Pour éliminer ce paramètre libre, [16] propose une nouvelle formulation pour le calcul de l'erreur de sous-segmentation, en prenant le minimum entre P^+ et P^-. Avec N le nombre de pixels dans l'image, ceci donne :

$$UE = \frac{1}{N}\left[\sum_{S \in G}\left(\sum_{P:P \bigcap S \neq \emptyset} min(P^+, P^-)\right)\right] \tag{6}$$

Par exemple, dans la figure 3, ça serait : $\frac{|A^-| + |B^+| + |C^-|}{|S|}$

3.2.4 Indépendantes du Ground Truth :

- **Compacité** :

La mesure de compacité des superpixels est basée sur le quotient iso-périmétrique Q(S) [17]. Pour un superpixel S_j, le quotient périmétrique met en relation l'aire $A(S_j)$ du superpixel avec l'aire d'un cercle avec le même périmètre $U(S_j)$:

$$Q(S) = \frac{4\pi A(S_j)}{U(S_j)^2} \tag{7}$$

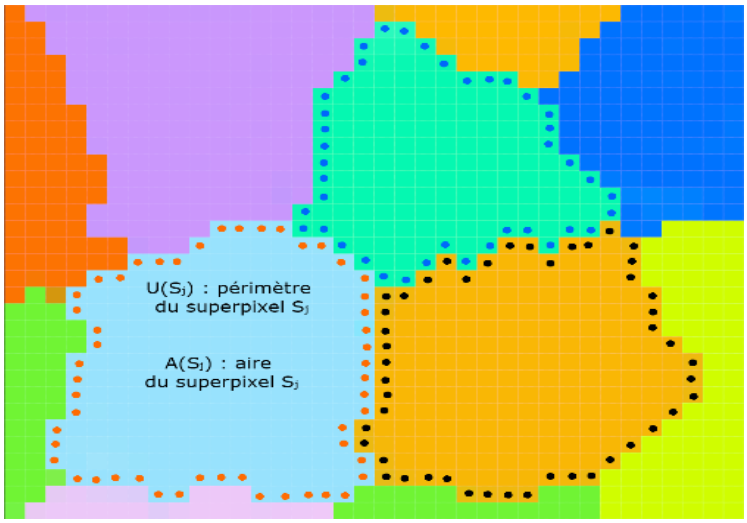

Figure 4: Compacité

Comme le cercle représente la forme la plus compacte, le quotient périmétrique mesure la compacité du superpixel, qui atteint 1 si et seulement si le superpixel a la forme d'un cercle. La mesure de compacité considère le quotient périmétrique de tous les superpixels pondérés par leur aire :

$$CO(S) = \frac{1}{N} \sum_{S_j \in S} Q(S).|S_j| \tag{8}$$

Ainsi, pour calculer cette métrique, les superpixels doivent représenter des composants connectés. Ceci n'est pas toujours le cas dans certains algorithmes de superpixels comme MeanShift ou WaterShed.

- **Variation expliquée** :

Une autre mesure qui ne dépend pas de la segmentation Ground Truth est la variation expliquée. La variation expliquée quantifie dans quelle mesure la variation de couleur dans l'image est capturée par la segmentation par superpixels [18] :

$$EV(S) = \frac{\sum_{S_j \in S} (I(S_j) - \mu)^2}{\sum_{n=1}^{N} (I(x_n) - \mu)^2} \quad (9)$$

Où μ est la moyenne de la couleur de l'image, $I(S_j)$: la moyenne du pixel S_j, $I(x_n)$: la valeur du pixel x_n. L'information contenue dans une image est définie principalement par une variation de couleur ou d'intensité. Ainsi, la variation expliquée mesure la fraction des informations capturées par la segmentation par superpixels.

3.2.5 Base de données :

Berkeley Segmentation Data Set and Benchmarks 500 (BSDS500) [19] : Cette nouvelle base de données est une extension de l'originale BSDS300, où pour les 300 images qui sont utilisées pour l'apprentissage et la validation, sont ajoutées 200 nouvelles images pour le test. Pour chaque image, une segmentation manuelle (Ground Truth) a été réalisée par au moins 5 personnes différentes. La figure 5 illustre une petit exemple d'une segmentation manuelle réalisée sur quelques images.

Figure 5: Exemples pris de la base de données BSDS500: pour chaque image correspond une segmentation Ground Truth

4. Morphologie mathématique en tant que prétraitement

La morphologie (branche non-linéaire du traitement du signal) est un outil mathématique qui fut engendré par la théorie des ensembles et la géométrie intégrale, utilisé dans le domaine de traitement de signaux et d'images. Les filtres et opérateurs morphologiques sont des transformations non-linéaires et causent des modifications dans les caractéristiques géométriques de l'image.

4.1 Élément structurant

L'élément structurant en morphologie est utilisé par les opérateurs morphologiques pour modifier l'image initiale. C'est un sous-ensemble d'image avec lequel l'image est balayée. Il y en a de différentes formes, entre autres, diamant, carré, croix. La figure 6 illustre son utilité.

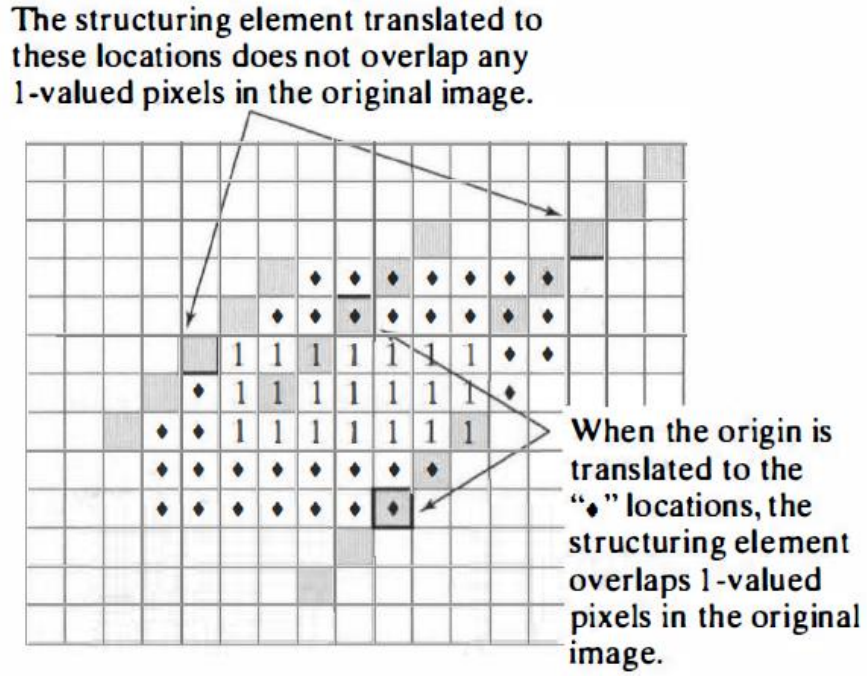

Figure 6 : Balayage de l'image par l'élément structurant

4.2 Dilatation

Dilatation de l'image ***f(x,y)*** par l'élément structurant ***b*** en niveaux de gris :

$\boldsymbol{D_f}$: Domaine de l'image

$\boldsymbol{D_b}$: Domaine de l'élément structurant ***b***

Voici l'opération mathématique de dilatation qui sert à connecter les caractéristiques de l'image :

$$(f \oplus b)(s,t) = \max\{f(s-x,t-y) - \frac{b(x,y)}{(s-x)}, (t-y) \in D_f; (s,y) \in D_b\} \qquad (10)$$

Cette opération a pour effet d'élargir les éléments de l'image.

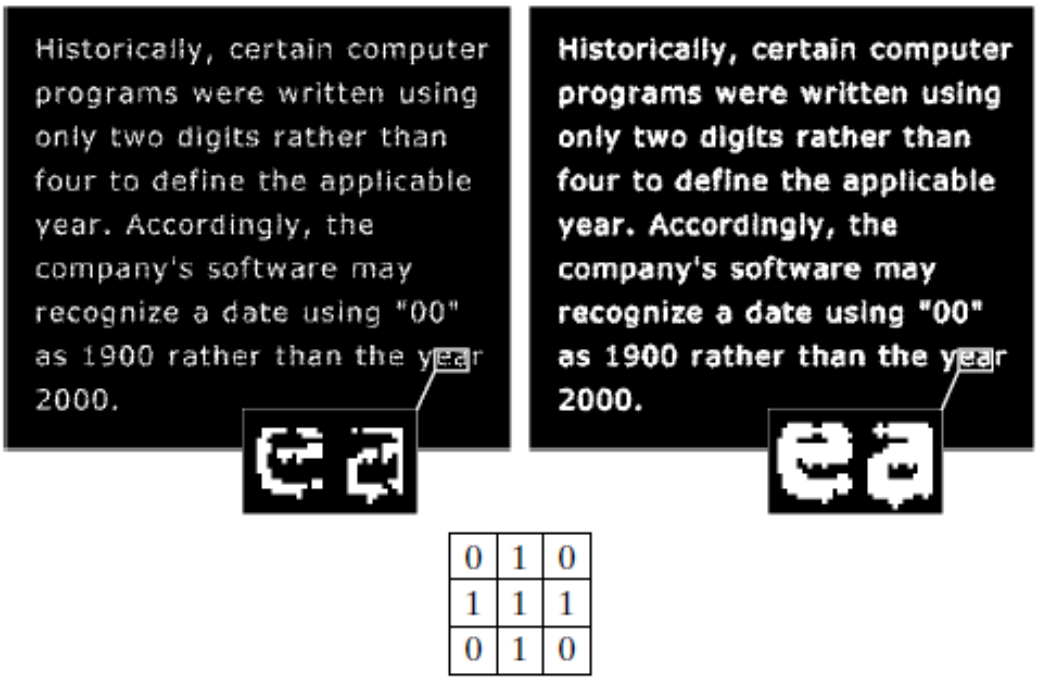

Figure 7 : Dilatation avec élément structurant en forme de croix

4.3 Érosion

Érosion de l'image ***f(x,y)*** par l'élément structurant ***b*** en niveaux de gris. Voici l'opération mathématique d'érosion qui sert à déconnecter les caractéristiques de l'image et éliminer les plus petites :

$$(f \ominus b)(s,t) = \max\{f(s-x,t+y) - \frac{b(x,y)}{(s+x)}, (t+y) \in D_f; (s,y) \in D_b\} \qquad (11)$$

Après avoir effectué l'opération de dilatation, on applique l'érosion (série d'opérations nommée « opening ») pour nettoyer (éroder) les bords des éléments de l'image ayant été dilatés. Cette opération permet de rendre les bordures des éléments de l'image plus évidentes. C'est ce qui pourrait contribuer à une segmentation plus efficace et performante.

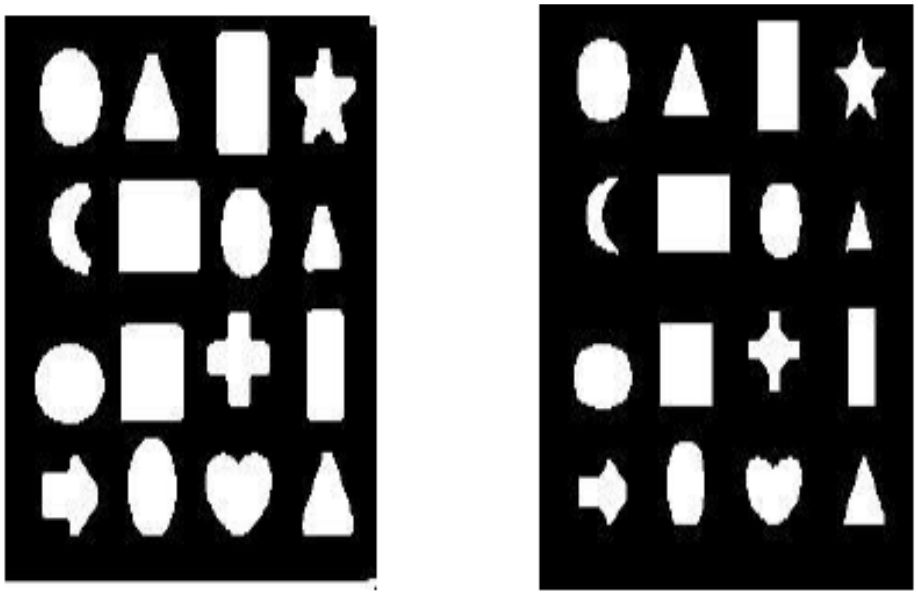

Figure 8 : Image dilatée Image érodée

5. Résultats

5.1 Algorithme des TurboPixels

Pour tester l'aspect qualitatif de l'algorithme des TurboPixels, nous avons pris quelques images test de la base de données de Berkeley et nous avons généré trois segmentations en variant le nombre de superpixels de 100, 400 et 1200. Les résultats obtenus sont affichés dans la figure 9.

Figure 9 : Exemple visuel de la segmentation par TurboPixels : N = 1200 (en haut à gauche), N = 400(au milieu) et N = 100 (en bas à droite)

Le temps d'exécution pour générer les superpixels est de 6 secondes pour N = 100, 5 secondes pour N = 200 et 4 secondes pour N = 1200. La taille de l'image est 481*321 pixels. L'algorithme est implémenté avec Matlab 2015a, sur un iMac 27-inch, Late 2013, 3.2 GHz Intel Core i5, 8 GB 1600 MHz DDR3, NVIDIA GeForce GT 755M 1024 MB.

Nous remarquons que la qualité de segmentation est excellente. Les superpixels obtenus sont compacts, de forme régulière et adhèrent parfaitement aux contours de l'image**.**

5.2 Reconstruction de l'image

Dans la figure 10 est présenté un exemple de reconstruction de l'image à partir de la segmentation par TurboPixels. À gauche l'image originale et à droite l'image reconstruite en utilisant 2000 superpixels où la couleur de chaque superpixel est remplacée par la moyenne de la couleur de tous les pixels qui le composent.

Figure 10 : Reconstruction de l'image

Visuellement, la qualité de construction est excellente. Nous sommes passés d'une image avec 481*321 = 154401 pixels à une représentation en utilisant uniquement 2000 segments. Le gain en terme de place mémoire est important, et le temps de calcul de certains algorithmes en traitement d'image serait grandement amélioré en travaillant sur les superpixels au lieu des pixels représentant l'image de départ

5.3 Résultats – Métriques

Dans cette section, nous allons présenter les résultats quantitatifs en utilisant les métriques de performance détaillées dans les sections précédentes. La figure 11 montre les résultats obtenues pour quelques images test de la base de données de Berkeley (BSDS500). Pour chaque image, sont présentés les rappels des contours, l'erreur de sous-segmentation, la compacité, la variation expliquée ainsi que le temps d'exécution de l'algorithme en secondes. Nous remarquons que globalement les performances s'améliorent quand le nombre de superpixels augmente, ce qui est un résultat plus au moins attendu. Par contre, le temps de calcul est inversement proportionnel au nombre de superpixels. Ceci est contre intuitif mais s'explique par la façon dont les TurboPixels

croissent en se formant sur la base d'un contour qui évolue. De plus, la première étape de l'algorithme étant de placer des germes sur une grille régulière, l'évolution de la segmentation se fait en parallèle et plus nous avons de germes, plus nous pouvons couvrir une surface plus grande.

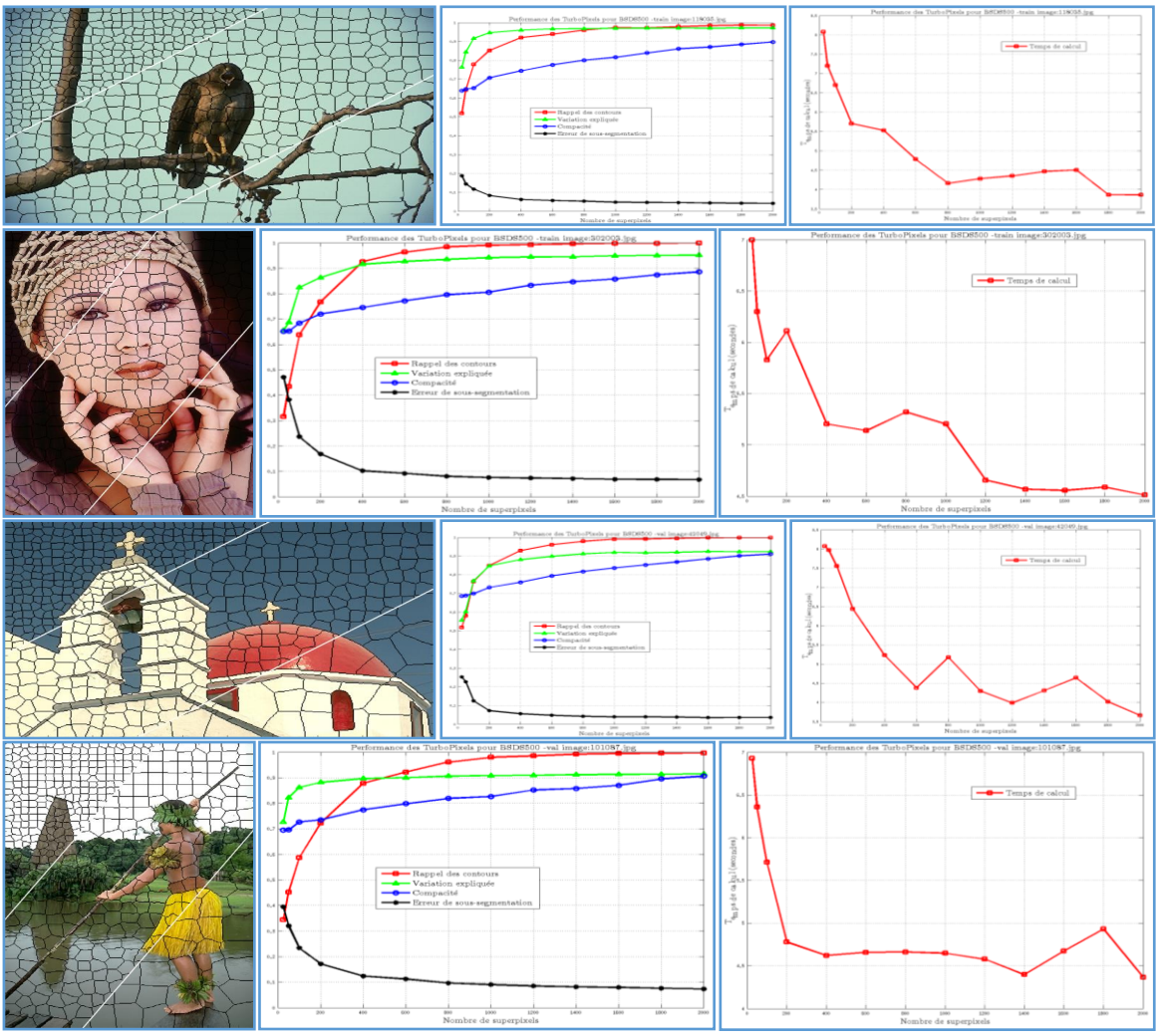

Figure 11 : Mesures de performance des TurboPixels : Rappel des contours, variation expliquée, compacité, erreur de sous-segmentation, et temps de calcul : N = 25 —> 2000

Nous constatons aussi que les performances au niveau des deux métriques qui dépendent de la segmentation manuelle (Ground Truth), à savoir l'erreur de sous-segmentation UE et le rappel des contours R (S, G), sont excellentes avec des taux de > 99% pour R (S, G) et <10% pour UE. Ces valeurs montrent que l'algorithme des TurboPixels arrive à produire des superpixels qui adhèrent bien aux contours de l'image et fournit une segmentation très proche de celle faite manuellement. Au niveau de la compacité, l'algorithme arrive à produire des superpixels assez compactes et de forme régulière. Cependant les performances n'atteignent pas les niveaux de R(S,G) ou UE. Même remarque pour les résultats de la variation expliquée qui quantifie dans quelle mesure la

variation de couleur dans l'image est capturée par la segmentation par superpixels. C'est dans cette dernière métrique que nous avons remarqué que le résultat dépend fortement de la distribution des couleurs dans l'image.

Nous avons aussi analysé le temps de calcul en fonction de la taille de l'image. La figure 12 présente les résultats obtenus au niveau du temps d'exécution de l'algorithme en variant la taille de l'image de 180*120 pixels à 4810*3210 pixels, et ceci pour un nombre de superpixels allant de 50 à 1800. La figure de gauche montre le temps en minutes en fonction de la taille de l'image normalisée par rapport à N_0 = 481*321. La figure de droite représente un zoom pour les images de taille allant de 0.14* N_0 à 1*N_0.

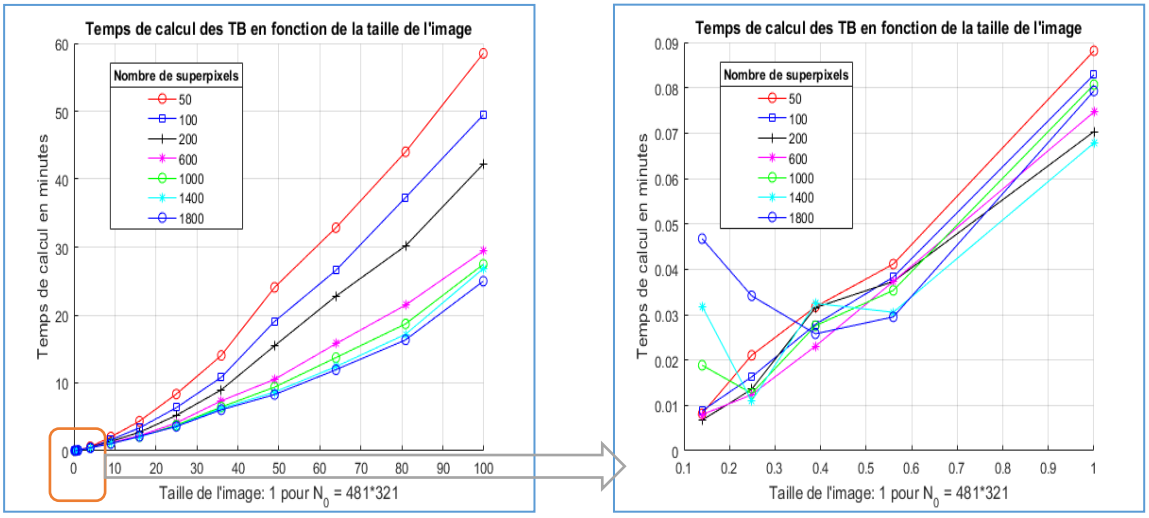

Figure 12 : Mesures de performance des TurboPixels : Temps de calcul en fonction de la taille de l'image

Nous constatons que le temps d'exécution de l'algorithme pour des images de taille inférieure à 10*N_0 est relativement linéaire en fonction de la taille de l'image. Par contre, pour des images de plus grande taille, le temps de calcul tend vers une courbe exponentielle. Cet effet devient plus marqué à mesure que le nombre de superpixels diminue.

5.4 Résultats – Morphologie par niveau de gris

Dans cette section, nous allons présenter les résultats des métriques de performance après avoir appliqué des étapes de prétraitement morphologiques. Nous allons commencer par illustrer l'effet de l'application de la morphologie sur l'aspect qualitatif des TurboPixels. La figure 13 montre les contours des superpixels obtenus en appliquant l'algorithme des TurboPixels. Chaque figure montre les contours des superpixels obtenus combinés à l'image originale. Pour bien apercevoir les détails, seule une zone restreinte de l'image est illustrée. La figure de gauche est la segmentation en appliquant l'algorithme de base.

La figure du milieu est la segmentation en ajoutant l'étape de prétraitement par morphologie. Et celle de droite, la différence entre les deux contours en valeur absolue.

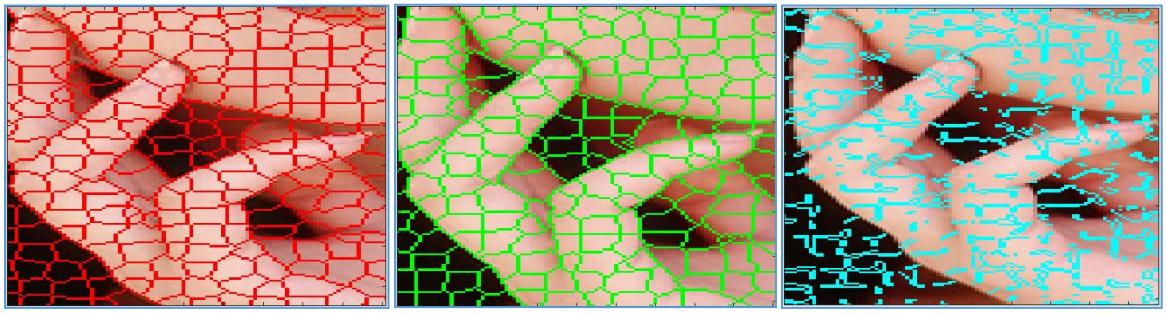

Figure 13 : Exemple qualitatif de segmentation avec étape de prétraitement. Comparaison avec l'algorithme de base

Nous remarquons que les deux contours (gauche et milieu) adhèrent bien aux contours de l'image originale. Au niveau de la différence sur l'image de droite, nous constatons que globalement, les superpixels obtenus dans les deux cas précédents adhèrent bien aux « vrais » contours de l'image.

La figure 14 présente les gains obtenus par rapport à l'algorithme de base pour les quatre métriques rappels des contours R(S,G), l'erreur de sous-segmentation UE, la compacité CO(S), et la variation expliquée EV(S). Il est utile de signaler que pour R(S,G), CO(S), EV(S), une valeur positive signifie un gain de performance, et le contraire pour UE.

Nous remarquons que l'introduction de cette étape de prétraitement morphologique permet d'obtenir de meilleurs résultats pour la segmentation par TurboPixels. Le gain le plus significatif est réalisé au niveau de la compacité, qui reste conséquent même si nous changeons le nombre de superpixels générés.

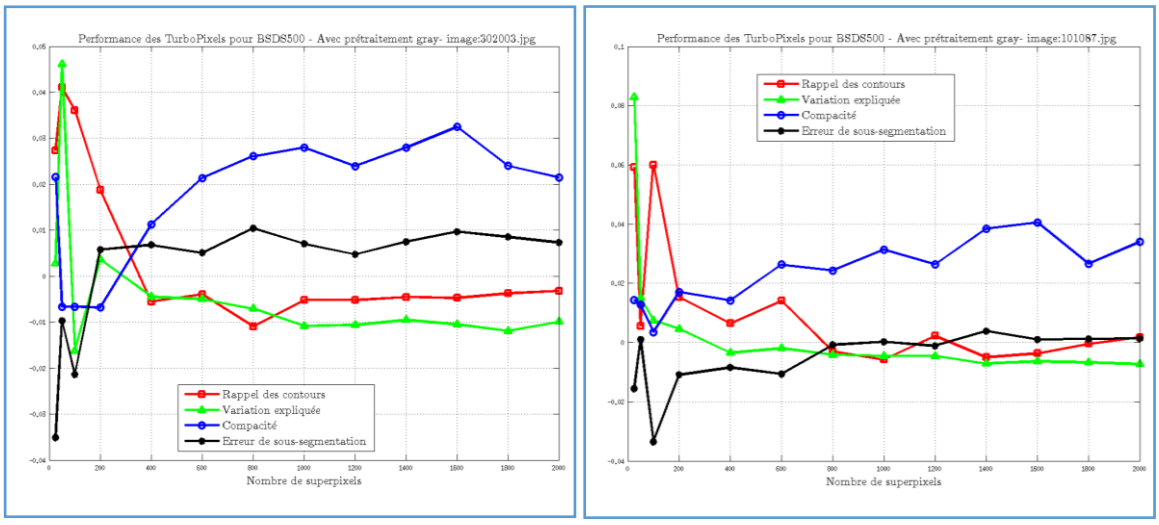

Figure 14 : Mesures de performance des TurboPixels avec prétraitement : Rappel des contours, variation expliquée, compacité, erreur de sous-segmentation, et temps de calcul : N = 25 —> 2000

Pour les autres métriques, nous constatons que le gain est plus important pour un nombre de superpixels inférieur à 200, et au-delà de ce nombre, les performances de l'algorithme se stabilisent mais d'une manière générale restent équivalentes par rapport à l'algorithme de base.

La figure 15 montre le résultat obtenu sur une moyenne de 65 images choisies aléatoirement à partir de la base de données de Berkeley. Sur la figure de gauche, nous observons des résultats similaires au paragraphe précédent, excepté au niveau de la variation expliquée où nous remarquons qu'elle se dégrade un peu si le nombre de superpixels augmente. La figure de droite montre la différence du temps d'exécution en secondes entre la méthode avec prétraitement morphologique et l'algorithme de base. Nous remarquons que l'introduction du prétraitement allonge l'exécution du programme d'environ 1 seconde pour les faibles valeurs du nombre de superpixels. Les temps d'exécution se rejoignent à mesure que le nombre de superpixels augmente.

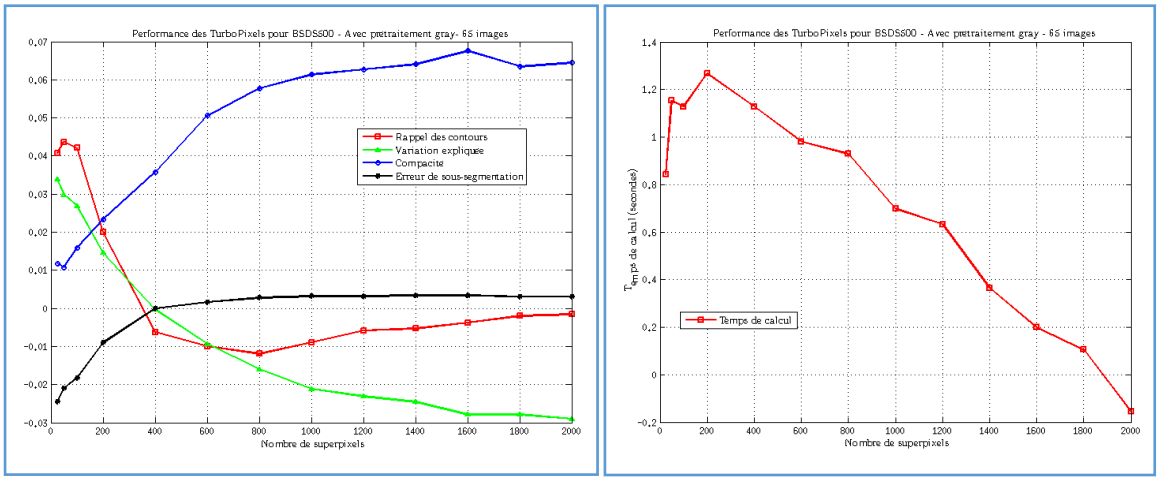

Figure 15 : Mesures de performance des TurboPixels avec prétraitement : Rappel des contours, variation expliquée, compacité, erreur de sous-segmentation, et temps de calcul : N = 25 - 2000 (Moyenne sur 65 images)

Ainsi, nous pouvons conclure que l'introduction d'une étape de prétraitement morphologiques permet bien d'améliorer les métriques de performance des TurboPixels sans en dégrader le temps de calcul.

Conclusion

Dans ce projet de session, nous avons étudié l'utilisation des étapes de prétraitement morphologiques pour améliorer la segmentation par superpixels. Nous avons opté pour l'algorithme des TurboPixels car c'est une méthode assez robuste et qui donne de bons résultats au niveau de la segmentation tout en étant relativement rapide en termes de temps de calcul. Dans le but d'évaluer les performances de la segmentation par superpixels, nous avons implémenté quatre métriques dont deux quantifient la qualité de la segmentation obtenue par rapport à une segmentation manuelle (Ground Truth), et les deux autres mesurent plus la forme compacte ou non des superpixels ainsi que dans quelle mesure ils arrivent à bien expliquer la variation des couleurs présentes dans l'image. À l'analyse des résultats obtenus, nous avons remarqué que l'algorithme des TurboPixels présente d'excellents résultats au niveau de l'erreur de sous segmentation (< 10%) et le rappel des contours (> 99%). Les performances sont moins importantes pour les deux autres métriques mais elles atteignent quand même les 80%. Nous avons aussi montré que d'une manière générale, les performances des TurboPixels s'améliorent quand le nombre de superpixels augmente. Étonnamment, ceci s'est aussi révélé vrai pour le temps de calcul qui est inversement proportionnel au nombre de superpixels.

Nous avons illustré un exemple de reconstruction d'image à partir des superpixels générés, et nous remarquons que malgré le faible nombre de superpixels utilisés, la qualité de reconstruction est assez remarquable et permettra d'améliorer grandement la vitesse de traitement de certains algorithmes de traitement d'images. Ces résultats confirment que les TurboPixels est une méthode très robuste pour la segmentation. Au niveau du temps d'exécution de l'algorithme, il parait assez raisonnable pour les applications où l'aspect temps réel n'est pas très important. L'algorithme arrive à segmenter une image 481*321 en moins de 10 secondes la plupart du temps. Par contre, le temps de calcul augmente considérablement avec la taille de l'image avec des valeurs autour de 30 minutes pour une image de 4810*3210 pixels. Ceci peut être un handicap pour certaines applications où le temps d'exécution est primordial.

L'introduction d'une étape de prétraitement morphologique permet bien d'améliorer les performances de la segmentation par TurboPixels. La compacité est la métrique qui présente le gain de performance le plus significatif et qui est maintenu même si on change le nombre de superpixels générés. Pour l'erreur de sous-segmentation, le rappel de contours et la variation expliquée, nous avons remarqué que le gain est plus important pour un nombre de superpixels inférieur à 200, et au-delà de ce nombre, les performances de l'algorithme se stabilisent mais sont assez équivalentes à la méthode de base.

Comme perspectives pour les travaux futurs, nous pouvons ajouter d'autres facteurs de morphologie comme le gradient morphologique pour améliorer davantage la segmentation pour les autres métriques. Nous pouvons aussi envisager les opérations morphologiques dans l'ordre inverse, érosion suivie d'une dilatation (nommée closing) sur une série d'images et vérifier si cela pourrait apporter plus de raffinement sur certaines d'entre elles, et du fait même, assurer une meilleure segmentation. Pour

terminer, Nous pouvons tenter de valider les effets bénéfiques de la morphologie mathématique à la segmentation par TurboPixels en utilisant d'autres métriques que celles apparaissant dans ce projet, telles que « Boundary Based Error measure » et « Contour Mapping ».

Références bibliographiques :

[1]. A. Levinshtein, A. Stere, K. Kutulakos, D. Fleet, S. Dickinson, and K. Siddiqi, « Turbopixels: Fast superpixels using geometric ows, » TPAMI, 2009.

[2]. X. Ren and J. Malik. « Learning a classification model for segmentation. » In Computer Vision, International Conference on, pages 10–17, Nice, France, October 2003.

[3]. T. Malisiewicz and A. A. Efros, « Improving spatial support for objects via multiple segmentations, » in BMVC, 2007.

[4]. C. Pantofaru, C. Schmid, and M. Hebert, « Object recognition by integrating multiple image segmentations, » in ECCV, 2008.

[5]. P. Mehrani and O. Veksler, « Saliency segmentation based on learning and graph cut refinement, » in BMVC, 2010.

[6]. C Lawrence Zitnick and Sing Bing Kang. Stereo for image-based rendering using image over-segmentation. International Journal of Computer Vision, 75(1):49–65, 2007

[7]. S. Yu and J. Shi. "Multiclass spectral clustering. IEEE ICCV," vol. 1, pp. 313–319, 2003.

[8]. P. Felzenszwalb and D. Huttenlocher. "Efficient graph-based image segmentation." IJCV, 59(2):167–181, 2004.

[9]. Luc Vincent and Pierre Soille. "Watersheds in Digital Spaces: An Efficient Algorithm Based on Immersion Simulations." PAMI, 13(6):583– 598, 1991.

[10]. D. Comaniciu and P. Meer. "Mean shift: A robust approach toward feature space analysis." IEEE Trans. PAMI, 24(5):603–619, 2002.

[11]. R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk. Slic superpixels compared to state-of-theart superpixel methods. Transactions on Pattern Analysis and Machine Intelligence, 2012.

[12]. M. Van den Bergh, X. Boix, G. Roig, B. de Capitani, L. Van Gool, "SEEDS: Superpixels Extracted via Energy-Driven Sampling," Computer Vision – ECCV 2012 Volume 7578 of the series Lecture Notes in Computer Science pp 13-26

[13]. K. Parvati, B. S. Prakasa Rao, and M. Mariya Das, “Image Segmentation Using Gray-Scale Morphology and Marker-Controlled Watershed Transformation,” Discrete Dynamics in Nature and Society, vol. 2008, Article ID 384346, 8 pages, 2008.

[14]. P. Arbeláez, M. Maire, C. Fowlkes, J. Malik. « Contour detection and hierarchical image segmentation. » Transactions on Pattern Analysis and Machine Intelligence, volume 33, number 5, pages 898–916, 2011.

[15]. R. Achanta, A. Shaji, K.Smith, A. Lucchi, P. Fua, and S. Süsstrunk, « SLIC Superpixels, » Tech. Rep., 2010.

[16]. P. Neubert, P. Protzel. « Superpixel benchmark and comparison. » Forum Bildverarbeitung, 2012.

[17]. A. Schick, M. Fischer, R. Stiefelhagen. « Measuring and evaluating the compactness of superpixels. » Proceedings of the International Conference on Pattern Recognition, pages 930–934, 2012.

[18]. D. Tang, H. Fu, and X. Cao. « Topology preserved regular superpixel. » In Multimedia and Expo, International Conference on, pages 765–768, Melbourne, Australia, July 2012

[19]. “Contour Detection and Hierarchical Image Segmentation.” P. Arbelaez, M. Maire, C. Fowlkes and J. Malik. IEEE TPAMI, Vol. 33, No. 5, pp. 898-916, May 2011.