Croissance de régions basée sur l'équation Eikonale pour la génération de superpixels

Pierre Buyssens, Isabelle Gardin, Su Ruan

Croissance de régions basée sur l'équation Eikonale pour la génération de superpixels

Pierre BUYSSENS[1], Isabelle GARDIN[1,2], Su RUAN[1]

[1]LITIS EA 4108 - QuantIF, Université de Rouen, 22 boulevard Gambetta, 76183 Rouen Cedex, France

[2]Département de médecine nucléaire, centre Henri-Becquerel, 1 rue d'Amiens, 76038 Rouen, France

pierre.buyssens@gmail.com

Résumé – Nous proposons une nouvelle méthode pour la génération de superpixels d'une image. Celle-ci repose sur une croissance de régions régie par la résolution d'une équation Eikonale adaptée. La méthode présente des performances supérieures à l'état de l'art en terme d'adhérence aux contours, tout en étant nettement plus rapide.

Abstract – We propose a new method to generate efficiently superpixels of an image. It lies on a region growing driven by the solution of an adapted Eikonal equation. The proposed method outperforms state-of-the-art methods on boundary adherence while being faster.

1 Introduction

La simplification d'une image en superpixels consiste à grouper des pixels de l'image en régions perceptuellement homogènes. En capturant certaines redondances dans l'images, les algorithmes générant des superpixels fournissent un moyen simple d'extraire des caractéristiques locales de l'image, et réduisent considérablement les complexités des méthodes mises en œuvre ensuite. Les superpixels sont de plus en plus utilisés comme élément clé dans de nombreux domaines de recherche tels la segmentation [4, 10], l'estimation de profondeur [11] ou encore la localisation d'objets [3].

Les principales propriétés attendues d'une méthode générant des superpixels sont:

- l'adhérence des superpixels aux contours des objets,
- la faible complexité,
- le contrôle du nombre de superpixels, et
- la simplicité d'utilisation (*i.e.* faible nombre de paramètres).

Une dernière propriété optionnelle est la nécéssité parfois d'obtenir des superpixels compacts.

La méthode ERGC (pour *Eikonal-based Region Growing Clustering*) répond à ces spécificités [1]. Elle consiste à réaliser une croissance de régions à partir de germes initiaux. Cette croissance est régie par une version modifiée de l'équation Eikonale prenant en compte les caractéristiques des régions pour le calcul de la solution :

$$\|\nabla U(\boldsymbol{x})\| = F(\boldsymbol{x}, R), \boldsymbol{x} \in \mathcal{I} \quad (1)$$

avec $\mathcal{I}$ le domaine image, R une région de l'image, et $U(\boldsymbol{x})$ la distance géodésique de $\boldsymbol{x}$ au germe le plus proche.

2 Méthode proposée

Par défaut le seul paramètre de ERGC est le nombre de germes initiaux. Ceux–ci sont placés sur une grille régulière, puis chaque germe est déplacé au pixel de gradient minimal dans un voisinage 3×3. Cette initialisation (similaire à [1, 5]) évite de placer un germe sur un contour. La couleur moyenne d'une région (futur superpixel) est ensuite initialisée avec la couleur du germe et de ses voisins en 4-connexité afin d'initialiser de manière plus robuste les superpixels.

Afin de résoudre efficacement l'équation 1, nous utilisons l'algorithme du Fast Marching [6] avec $F_c(\boldsymbol{x}, R_i) = \|\boldsymbol{C}_{\boldsymbol{x}} - \boldsymbol{C}_{R_i}\|_2^2$ où $\boldsymbol{C}_{\boldsymbol{x}}$ et $\boldsymbol{C}_{R_i}$ sont respectivement la couleur du pixel $\boldsymbol{x}$ et la couleur moyenne de la région R_i. Lors du déroulement de l'algorithme, lorsqu'un pixel est fixé à une région, sa couleur moyenne est mise à jour :

$$\boldsymbol{C}_{R_i} \leftarrow \frac{(\boldsymbol{C}_{R_i} \times Card(R_i) + \boldsymbol{C}_{\boldsymbol{x}})}{(Card(R_i) + 1)}$$

Cette formalisation montre clairement que la carte de priorités utilisée pour la résolution de l'équation Eikonale n'est pas fixe, et évolue selon la croissance des régions. Cette modification est la principale différence par rapport aux méthodes plus classiques utilisant par exemple la carte de gradient comme image de potentiel. Cette croissance de régions favorise la création de superpixels homogènes et adhérant bien aux contours. Cette homogénéité de couleur peut être observée à la figure 1. Les superpixels dans les coins de l'image épousent le gradient de couleur bleue présent dans le ciel, et sont ainsi concentriques.

[1]Code disponible publiquement à l'adresse
https://sites.google.com/site/pierrebuyssens/ergc

Similairement à [1], un terme de contrainte spatiale (F_s) peut être ajouté au membre de droite de l'équation 1 afin d'obtenir des superpixels compacts. La normalisation des termes couleur et spatial n'étant pas simple, un facteur de compacité m permet de pondérer ces deux quantités. Le second terme de l'équation 1 devient ainsi:

$$F(\boldsymbol{x}, R_i) = \sqrt{F_c(\boldsymbol{x}, R_i)^2 + F_s(\boldsymbol{x}, \boldsymbol{s}_i)^2}$$

avec

$$F_s(\boldsymbol{x}, \boldsymbol{s}_i) = \frac{\|\boldsymbol{x} - \boldsymbol{s}_i\|_2^2}{S} \times m$$

où S est le pas d'échantillonage des germes initiaux, et $\boldsymbol{s}_i$ est le germe initial de la région R_i. La figure 1 montre l'effet du paramètre m sur la compacité des superpixels.

Pour le traitement des images couleurs, les images sont traitées dans l'espace CIELAB. $\boldsymbol{C}$ se réduit alors au vecteur $[l, a, b]^T$.

3 Résultats et comparaison à l'état de l'art

Nous évaluons les performances de ERGC en la comparant à 4 méthodes basées sur des graphes : GS04 [2], NC05 [7] et les deux variantes de la méthode de [9] GCa10 et GCb10. Nous comparons également 3 méthodes basées sur l'intensité ou le gradient de l'image : TP09 [5], QS08 [8] et la récente méthode SLIC [1].

La base de données Berkeley utilisée pour les tests comprend 300 images et approximativement 10 images manuellement segmentées pour chaque image.

Le tableau 2 résume ces résultats comparatifs et présente les temps d'exécution sous forme de coefficients multiplicateur avec les temps de SLIC comme référence.

3.1 Adhérence aux contours

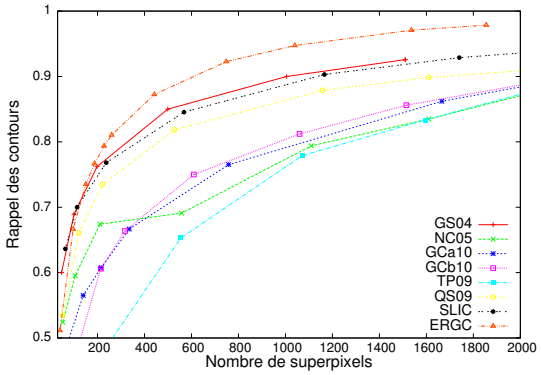
Le rappel des contours (*Boundary Recall*) mesure la fraction de vrais contours se retrouvant dans l'image segmentée par l'algorithme. Pour l'expérience, un contour segmenté est considéré comme bon si il se trouve à une distance inférieure à 2 pixels d'un vrai contour (comme dans [1, 9, 5]).

L'erreur de sous-segmentation (*undersementation*) mesure la fraction de pixels "débordant de la segmentation"causée par les superpixels chevauchant la segmentation manuelle. Étant donnée une segmentation manuelle $g_1, \ldots, g_k$ et une segmentation superpixellique $R_1, \ldots, R_N$, l'erreur de sous-segmentation pour un segment g_i est calculée via:

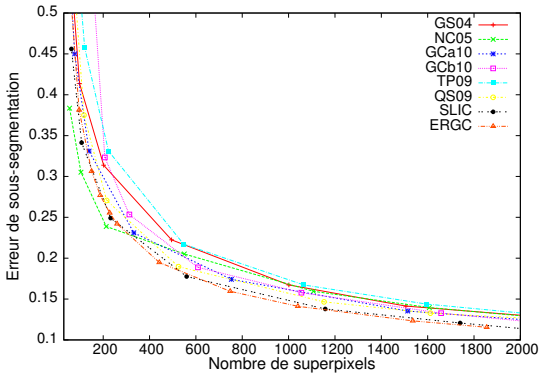
$$\frac{\left[\sum_{\{R_j | R_j \cap g_i > B\}} Area(R_j)\right] - Area(g_i)}{Area(g_i)}$$

où B est fixé à 5% de $Area(R_j)$ (calcul identique à celui de [1]). La moyenne pour tous les segments et toutes les images est ensuite calculée.

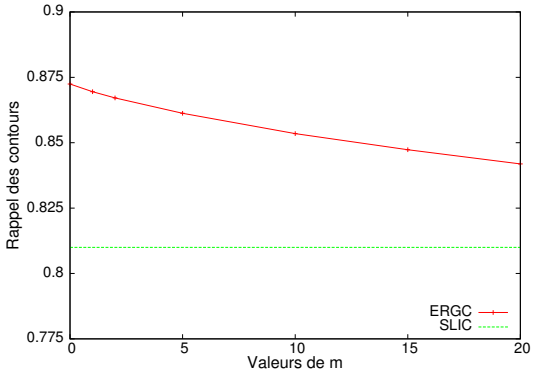
Les figures 2(a) et 2(b) montrent le rappel des contours et l'erreur de sous-segmentation pour les différents algorithmes



(a) Rappel des contours



(b) Erreur de sous-segmentation



(c) Rappel des contours selon m

Figure 2: Comparaison quantitative du rappel des contours (a) et de l'erreur de sous-segmentation (b) selon le nombre de superpixels. (c) Moyenne du rappel des contours pour 500 superpixels (environ) selon différentes valeurs de m. Le facteur de compacité de SLIC reste fixe à sa valeur par défaut.

Figure 1: Image *Aigle* avec 150 (coins supérieur gauche) et 600 (coins inférieur droit) superpixels (Nombre de superpixels pour l'image entière). De gauche à droite, variation du facteur de compacité : $m = 0$, $m = 1$, et $m = 20$.

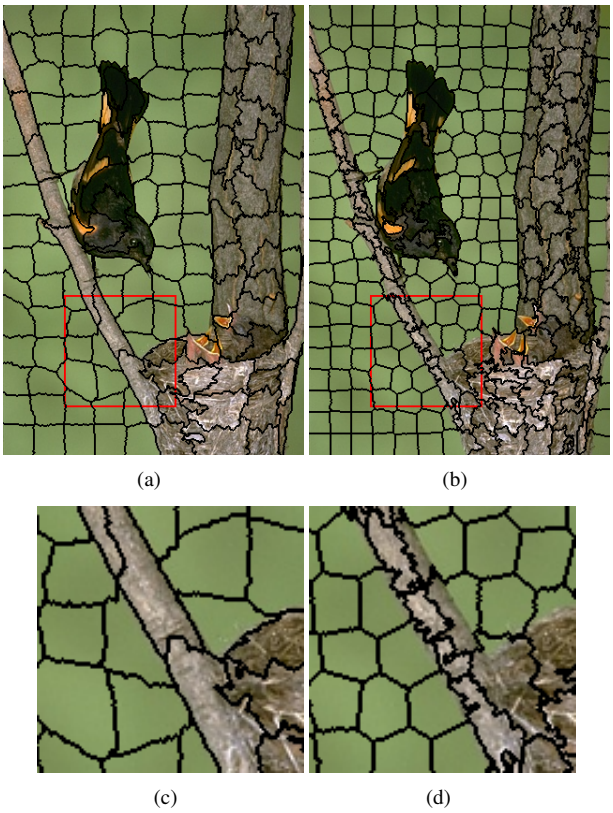


Figure 3: (a) Résultat sur l'image *oiseau* avec 150 ERGC superpixels et (b) environ 300 SLIC superpixels. Bas: détail pour ERGC (c) et SLIC (d). Malgré presque deux fois plus de superpixels, SLIC échoue à bien adhérer aux contours de la branche.

selon le nombre de superpixels de l'image. Ces résultats ont été obtenus sans contrainte spatiale ($m = 0$). L'ajout de telles contraintes n'a pas d'influence sur l'erreur de sous-segmentation, mais décroit légèrement le rappel des contours (Figure 2(c)). Celui-ci reste cependant supérieur à celui obtenu par SLIC. Au dessus de 400 ERGC superpixels, environ deux fois plus de SLIC superpixels sont nécéssaires pour obtenir un rappel de contours équivalent. La figure 3 montre un tel résultat: certains contours de la branche ne sont pas bien adhérés par SLIC malgrè leur nombre relativement élevé.

La méthode ERGC présente les meilleurs résultats de sous-segmentation et de rappel des contours. ERGC a également été testée en niveaux de gris uniquement (colonne ERGC$_g$ du tableau 2). La méthode présente alors un taux de sous segmentation plus élevé, mais est plus rapide.

Table 1: Comparaison des temps d'exécution (en secondes) des algorithmes SLIC et ERGC pour différentes tailles d'image. L'indice $_g$ indique que la méthode repose sur des images en niveaux de gris.

Taille des images	SLIC	ERGC	ERGC$_g$
320×240	0.4	0.14	0.07
481×321	0.84	0.28	0.16
2048×1536	16	7.8	4.7
4096×3072	63.6	35.8	21.2

3.2 Complexité et vitesse d'exécution

Les superpixels sont souvent utilisés comme prétraitement. Il est ainsi essentiel de les générer en un temps raisonable. La complexité de ERGC repose essentiellement sur la complexité de l'algorithme du fast-marching, qui est de l'ordre de $\mathcal{O}(n \log(n))$ avec une structure de tas appropriée pour le tri des pixels selon leur distance géodésique. Malgré cette complexité théorique, ERGC est très rapide en pratique, et est quasi-linéaire.

Les temps d'exécution sont reportés au tableau 2 sous la forme de coefficients multiplicateurs avec les temps de SLIC en référence. Bien que SLIC ait une complexité théorique en $\mathcal{O}(n)$, l'algorithme requiert plusieurs itérations pour converger ce qui est lent en pratique. ERGC est ainsi deux à trois fois plus rapide que SLIC pour des images de la base Berkeley.

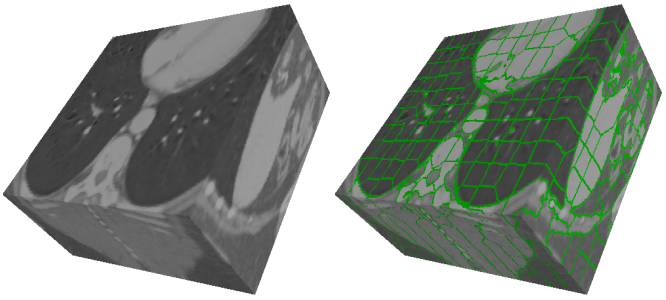
Le tableau 1 donne à titre indicatif les temps d'exécution de SLIC et de ERGC pour différentes tailles d'image. Ceux-ci ont été obtenu sur un ordinateur de bureau équipé d'un processeur Intel double cœur cadencé à $1.30GHz$ et de 4 GB RAM.

4 Conclusion

Dans ces travaux, nous présentons une nouvelle méthode pour la génération efficace de superpixels. Fondée sur une croissance de régions régie par la solution d'une équation Eikonale, la méthode ERGC surpasse les techniques de l'état de l'art que ce soit en terme d'adhérence aux contours ou en vitesse d'exécution. ERGC peut de plus être facilement adaptée à des

Table 2: Comparatif quantitatif de différentes méthodes pour l'adhérence aux contours et la vitesse d'exécution. L'indice $_g$ indique que la méthode repose sur des images en niveaux de gris. Résultats obtenus avec approximativement 500 superpixels

	Basé graphes				Basé intensité/gradient				
	GS04 [2]	NC05 [7]	GCa10$_g$ [9]	GCb10$_g$ [9]	TP09 [5]	QS08 [8]	SLIC12 [1]	ERGC	ERGC$_g$
Adherence aux contours									
Erreur de sous-segmentation	0.23	0.22	0.22	0.22	0.24	0.20	**0.19**	**0.19**	0.21
Rappel des contours	0.84	0.68	0.69	0.70	0.61	0.79	0.82	**0.88**	**0.88**
Vitesse de segmentation									
320×240 image	$\times 3$	$\times 494$	$\times 14$	$\times 11$	$\times 22$	$\times 13$	$\times 1$	$\times 0.33$	$\times 0.19$
2048×1536 image	$\times 6$	N/A	$\times 21$	$\times 15$	$\times 53$	$\times 12$	$\times 1$	$\times 0.49$	$\times 0.31$
Contrôle du nombre de superpixels	Non	Oui	Oui	Oui	Oui	Non	Oui	Oui	Oui
Contrôle de la compacité	Non	Non	Non	Non	Non	Non	Oui	Oui	Oui
Extension aux supervoxels	Non	Non	Oui	Oui	Non	Non	Oui	Oui	Oui



(a) (b)

Figure 4: Approximativement 1000 supervoxels pour un volume CT. Pour des raisons de clarté, seul l'intérieur du corps est montré.

données 3D pour la création de supervoxels (Figure 4), et est disponible publiquement.

5 Remerciements

Ce travail a été réalisé dans le cadre du projet SIRTDose financé par l'appel à projet "Physique Cancer" 2012 (INSERM - Plan Cancer). Les auteurs remercient les auteurs de [1] pour les nombreux échanges sur les superpixels.

References

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

[2] P.F. Felzenszwalb and D.P. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004.

[3] Brian Fulkerson, Andrea Vedaldi, and Stefano Soatto. Class segmentation and object localization with superpixel neighborhoods. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 670–677. IEEE, 2009.

[4] Stephen Gould, Jim Rodgers, David Cohen, Gal Elidan, and Daphne Koller. Multi-class segmentation with relative location prior. *International Journal of Computer Vision*, 80(3):300–316, 2008.

[5] A. Levinshtein, A. Stere, K.N. Kutulakos, D.J. Fleet, S.J. Dickinson, and K. Siddiqi. Turbopixels: Fast superpixels using geometric flows. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(12):2290–2297, 2009.

[6] J.A. Sethian. *Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science*, volume 3. Cambridge university press, 1999.

[7] J. Shi and J. Malik. Normalized cuts and image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):888–905, 2000.

[8] A. Vedaldi and S. Soatto. Quick shift and kernel methods for mode seeking. *Computer Vision–ECCV 2008*, pages 705–718, 2008.

[9] O. Veksler, Y. Boykov, and P. Mehrani. Superpixels and supervoxels in an energy optimization framework. *Computer Vision–ECCV 2010*, pages 211–224, 2010.

[10] Yi Yang, Sam Hallman, Deva Ramanan, and Charless Fowlkes. Layered object detection for multi-class segmentation. In *Computer vision and pattern recognition (CVPR), 2010 IEEE conference on*, pages 3113–3120. IEEE, 2010.

[11] 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.