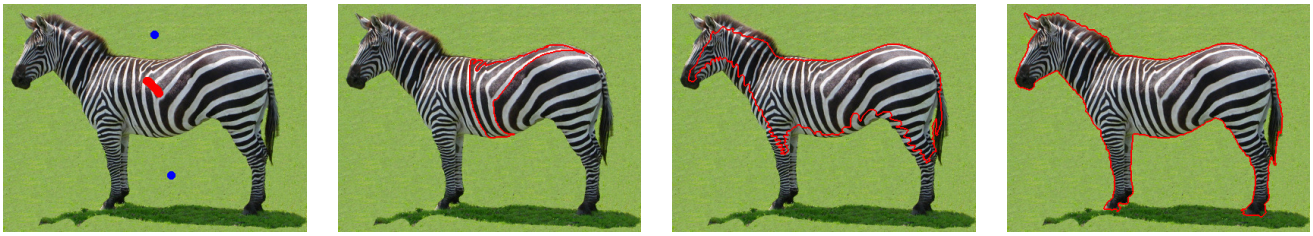
Diffusion de labels multivaluée pour la segmentation semi-supervisée

Pierre BUYSSENS, Olivier LÉZORAY

GREYC CNRS (UMR 6072), UNICAEN, ENSICAEN, Équipe Image
6, Bd. Maréchal Juin, 14000 Caen, FRANCE
{pierre.buyssens,olivier.lezoray}@unicaen.fr

Résumé – Les méthodes de diffusion ont prouvé leur efficacité pour des tâches comme la segmentation semi-supervisée. L'intégration de patches dans le calcul de leur fonction de vitesse permet de traiter aisément des images texturées. Cependant le surcoût calculatoire reste trop important pour des tâches de bas niveau. Dans cet article, nous proposons une nouvelle fonction de potentiel multivaluée basée sur la couleur qui contourne ce défaut. Elle permet des segmentations semi-supervisées rapides sur des images naturelles et de texture.

Abstract – Diffusion methods has proven their efficiency for tasks such as semi-supervised segmentation. The introduction of patches as a part of their speed function allows to deal with textured images. However, the computational burden with such variants remains too important for low-level tasks. In this paper, we propose a multivalued color-based potential function that partly alleviates this flaw. It allows to efficiently perform semi-supervised segmentation of natural and textured images.



Germes initiaux | Résultat avec [3] | Résultat avec [2] | Méthode proposée

1 Introduction

Les méthodes basées sur la diffusion de labels ont largement été utilisées à des fins de segmentation [8, 1, 10, 3] (pour ne citer que ces travaux). Lorsqu'un utilisateur labélise manuellement des parties de l'image, on parle alors de segmentation semi-supervisée, approche pour laquelle la plupart des méthodes de diffusion peuvent être utilisées. En fait, elles diffèrent principalement de par le nombre d'interactions manuelles nécessaires pour obtenir une bonne segmentation.

La segmentation du zèbre (voir plus haut) montre que des germes très partiellement distribués (première image) sont clairement insuffisants avec une méthode de l'état-de-l'art comme [3] (deuxième image). En fait, [3] requiert beaucoup plus de germes pour bien segmenter le zèbre, et c'est aussi le cas pour la plupart des approches basées sur une diffusion : la qualité de la segmentation repose en grande partie sur la nature de l'image et sur la quantité d'interactions d'un utilisateur.

Dans ce papier, nous proposons un cadre multivalué pour la diffusion dont le but est d'obtenir de bonnes segmentations tout en réduisant la quantité d'interactions manuelles.

La méthode proposée repose sur la résolution d'une équation Eikonale qui calcule pour chaque pixel p de l'image I la distance $U(x)$ au plus proche pixel germe. L'image est ainsi partitionnée et donc segmentée.

Notre méthode [1] étend nos précédents travaux [2] afin d'encapsuler une composante multivaluée. L'équation considérée est :

$$\begin{cases} |\nabla U(x)| = P(u, L_t) & \forall p \in I \\ U(x) = \phi(x) & \forall x \in L_0 \end{cases} \quad (1)$$

où P est la fonction de potentiel (positive), ϕ une fonction d'initialisation, et L_t l'ensemble des pixels labélisés au temps t. L_0 représente ainsi l'ensemble des pixels initialement labélisés (les germes).

Contrairement à la fonction de potentiel classique basée sur le gradient de l'image, où P est complètement connue au début de la diffusion, la fonction de potentiel proposée est dynamique et est calculée selon des statistiques des régions qui croissent lors de la diffusion.

Afin de traiter efficacement des objects multivalués, nous modélisons les régions avec des modèles de mélange de Gaussiennes (GMM pour *Gaussian Mixture Model*) de sorte que chaque région peut être composée de plusieurs caractéristiques différentes. Sur l'image de zèbre par exemple, la région rouge

1. Code disponible à l'adresse : http://sites.google.com/site/pierrebuyssens/code/multivalued-diffusion

est principalement composée d'une composante blanche et d'une composante noire. La région (front) peut alors croître librement selon ces deux composantes et englobe finalement tout le zèbre (image de droite). À noter que les GMM ont été utilisés dans un cadre d'optimisation basé sur les coupes de graphes (Graph-Cut) [7]. Notre approche présente cependant une complexité globalement inférieure.

La section 2 détaille la méthode proposée, et des résultats comparatifs sur des images naturelles et des images de texture sont fournis à la section 3.

2 Diffusion multivaluée proposée

Dans la suite, un pixel p consiste en un couple de coordonnées (x_p, y_p) et d'un vecteur de caractéristiques associés $\mathbf{X}_p \in \mathbb{R}^d$ ($d = 3$ pour des images couleurs). Une région R_i est entourée par un front Γ_i. Le modèle de mélange de Gaussiennes (GMM) associé à R_i est composé de K Gaussiennes et est noté $\Pi_i = \sum_{k=1}^{K} \pi_k G_k^i$ où G_k^i est la $k^{\text{ième}}$ Gaussienne du mélange et π_k son coefficient de mélange. Dans la suite, nous supposons que les coefficients de mélange pour un GMM donné somment à 1 (i.e. $\sum_{k=1}^{K} \pi_k = 1$). L'algorithme est, dans la suite, détaillé pour un GMM en particulier, donc l'indice i est occulté pour des questions de clarté. Une Gaussienne donnée G_k consiste en un vecteur moyen $\boldsymbol{\mu}_k$, une matrice de covariance $\boldsymbol{\Sigma}_k$ de dimension $d \times d$, et d'un nombre d'échantillons n_k associés.

2.1 Fonction de potentiel basée sur un mélange de Gaussiennes

Dans cet article, nous proposons une fonction de potentiel dynamique qui dépend des statistiques des régions en train de croître. Soit un mélange Π composé de K Gaussiennes associé à une région R

$$\Pi = \sum_{k=1}^{K} \pi_k G_k \tag{2}$$

avec

$$G_k(\mathbf{X}) = \frac{1}{(2\pi)^{\frac{d}{2}} |\boldsymbol{\Sigma}_\mathbf{k}|^{\frac{1}{2}}} e^{\left(-\frac{1}{2}(\boldsymbol{X}-\boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}^{-1} (\mathbf{X}-\boldsymbol{\mu}_k)\right)} \tag{3}$$

alors le potentiel $P(p|R)$ d'un pixel $p \in \Gamma$ est calculé selon

$$P(p|R) = \frac{1}{\gamma(p|R)} \tag{4}$$

où

$$\gamma(p|R) = \text{argmax}_k(\pi_k G_k(\mathbf{X}_p)) \tag{5}$$

Ici, la fonction γ peut être vue comme la fonction de vitesse du front Γ pour un pixel p. L'équation 5 n'associe à p (dont le vecteur de caractéristiques est $\boldsymbol{X}_p$) qu'une Gaussienne (assignation dure [2]). À cette étape, on peut voir qu'une Gaussienne qui est sous représentée au sein du mélange (i.e. $\pi_k \ll 1$) va être sujette à de faible vitesse $\pi_k G_k(\cdot)$. La diffusion d'une telle Gaussienne n'est ainsi pas favorisée.

2. Un assignation probabiliste (doux) peut sembler préférable. Cependant les bénéfices sont négligeables en comparaison du surcoût de calcul.

2.2 Détails de l'algorithme

Initialisation : Soit un ensemble de pixels initialement labélisés, l'algorithme commence par modéliser chaque sous ensemble de même label à l'aide d'un modèle de mélange de Gaussiennes. Chaque GMM est initialement composé d'au plus K composantes ($K = 10$ dans nos expériences). Pour un GMM donné, chaque vecteur moyen μ_k est estimé par l'algorithme des k-means, et une seule Gaussienne est associé à chaque pixel labélisé. Les matrices de covariance Σ_k sont alors calculées selon cet assignement. Finalement, les coefficients du mélange sont calculés comme la probabilité pour un pixel d'appartenir à une Gaussienne donnée $\pi_k = n_k / \sum_{t=1}^{K} n_t$.

À noter que les Gaussiennes non pertinentes (celles qui ne représentent aucun pixel) sont supprimées automatiquement. Un GMM initial peut ainsi être composé de moins de K composantes.

Diffusion : Le processus de diffusion est effectué via l'algorithme du Fast Marching [9]. Le potentiel de chaque pixel $p \in \Gamma_i, i \in \{1, \ldots, N\}$ est calculé via l'équation 4. Lors de la diffusion, à chaque fois qu'un pixel p est labélisé, la Gaussienne attenante G (celle qui a maximisé $\pi_k G_k$, voir l'équation 5) est mise à jour avec $\mathbf{X}_p$. Afin d'éviter le calcul du vecteur moyen $\boldsymbol{\mu}$ et de la matrice de covariance $\boldsymbol{\Sigma}$ de G de zéro, ce qui peut être *très* coûteux, G est simplement mise à jour. L'algorithme, non trivial, permettant la mise à jour d'une Gaussienne G avec un vecteur de caractéristiques $\mathbf{X}$ est détaillé à l'algorithme 1. À noter que les coefficients de mélange pour toutes les Gaussiennes du GMM sont également mis à jour étant donné qu'ils somment à 1 ($\sum_{k=1}^{K} \pi_k = 1$). L'algorithme termine lorsque tous les pixels ont été labélisés.

Algorithme 1 Mise à jour d'une Gaussienne G par un vecteur de caractéristiques $\mathbf{X}$

Entrées: $\mathbf{X}$, G
Sortie: G mis à jour
$\boldsymbol{\Sigma} \leftarrow \boldsymbol{\Sigma} \times (n-1)$
$n \leftarrow n+1$
$\boldsymbol{\Delta} \leftarrow \mathbf{X} - \boldsymbol{\mu}$
{Mise à jour de la moyenne $\boldsymbol{\mu}$}
$\boldsymbol{\mu} \leftarrow \boldsymbol{\mu} + \boldsymbol{\Delta}/n$
{Mise à jour de la matrice de covariance $\boldsymbol{\Sigma}$}
$\boldsymbol{\Lambda} \leftarrow \mathbf{X} - \boldsymbol{\mu}$
$\boldsymbol{\Sigma}_{\mathbf{i,i}} \leftarrow \boldsymbol{\Sigma}_{\mathbf{i,i}} + \boldsymbol{\Delta}_\mathbf{i} \times \boldsymbol{\Lambda}_\mathbf{i}$
$\boldsymbol{\Sigma}_{\mathbf{i,j}} \leftarrow \boldsymbol{\Sigma}_{\mathbf{i,j}} + \boldsymbol{\Lambda}_\mathbf{i} \times \boldsymbol{\Lambda}_\mathbf{j}$ if $i \neq j$
$\boldsymbol{\Sigma} \leftarrow \boldsymbol{\Sigma}/(n-1)$
{Mise à jour des coefficients de mélange $\pi_k, \forall\, G_k$}
$\pi_k \leftarrow n_k / \sum_{t=1}^{K} n_t$

2.3 Complexité

La complexité de la diffusion repose essentiellement sur la complexité de l'algorithme du Fast Marching et du nombre de Gaussiennes. Avec une structure de tas appropriée pour trier les

pixels selon leur distance, la complexité du Fast Marching dépend du nombre total de pixels D et est de l'ordre de $\mathcal{O}(D\log(D))$. De plus, le calcul du potentiel pour chaque pixel p implique K estimations de $\pi_k G_k(\mathbf{X}_p)$ (équation 5). La complexité globale de l'algorithme est donc $\mathcal{O}(K\,D\log(D))$.

Cette complexité brute doit cependant être nuancée et peut être largement réduite. En effet le calcul de $\gamma(p|R)$ (équation 5) peut facilement être parallélisé. De plus, en utilisant des structures de données différentes (avec un surcoût en mémoire), des implémentations du Fast Marching ayant une complexité en $\mathcal{O}(D)$ ont été proposées [6, 11]. La complexité de notre algorithme peut ainsi être réduite théoriquement en $\mathcal{O}(D)$. Cette complexité linéaire est un réel avantage supplémentaire pour l'utilisation de notre approche. Sans optimisation particulière (voir code), la segmentation du zèbre est effectuée en $2s$.

3 Résultats

3.1 Segmentation semi-supervisée multi labels

Dans cette section nous comparons notre approche avec des méthodes de l'état de l'art : une approche de diffusion basée sur le gradient [4], l'algorithme *Eikonal Region Growing Clustering* [3] [2], et la méthode dite *Power Watershed* [4] [3]. La première [4] résout une équation Eikonale avec une fonction de potentiel basée sur le gradient, tandis que la méthode *Power Watershed* [3] utilise une variante de la ligne de partage des eaux sur graphe. Finalement, nos précédents travaux [2] sont basés sur la résolution d'une équation Eikonale avec une fonction de potentiel qui ne dépend que de la couleur moyenne des régions en formation.

Les résultats comparatifs sur l'image *église* (Figure 1, première ligne) montrent de nombreuses erreurs de labélisation pour les méthodes de l'état de l'art, spécialement sur le devant du bâtiment ou sur son toit. Sur l'image *paysage* (Figure 1, deuxième ligne), des erreurs similaires peuvent être observées, particulièrement sur les rochers et l'herbe. Finalement, [4, 2] montrent de nombreuses erreurs de segmentation sur l'image *Oscar* (Figure 1, troisième ligne) tandis que la méthode *Power Watershed* [3] s'en sort mieux.

Malgré des labels initiaux distribués grossièrement et/ou très partiellement, notre approche basée sur des mélanges de Gaussiennes permet d'obtenir des segmentations de qualité équivalentes voire meilleures que celles obtenues à l'aide de méthodes de l'état de l'art.

3.2 Ajout d'informations de texture

La formulation proposée (voir la section 2) ne se limite pas à des vecteurs couleurs. Dans cette section, nous montrons des résultats supplémentaires sur des images de texture en ajoutant de l'information de texture à notre modèle. Introduits par [5], les tenseurs de structure sont une extension naturelle au gradient pour les images multivaluées. Ils encodent efficacement à la fois les variations locales de couleur ainsi que leur orientation. Pour des images RVB $2D$, les tenseurs de structure se réduisent à des matrices 2×2 :

$$\begin{aligned}\mathbf{S} &= \sum_{c\in\{R,G,B\}} \overrightarrow{\nabla I^c}.\overrightarrow{\nabla I^c}^T \\ &= \lambda_1.\mathbf{u}.\mathbf{u}^T + \lambda_2.\mathbf{v}.\mathbf{v}^T \quad \text{avec} \quad \lambda_1 > \lambda_2\end{aligned}$$

avec $\lambda_{\{1,2\}}$ les valeurs propres de $\mathbf{S}$ et $\mathbf{u}$, $\mathbf{v}$ les vecteurs propres associés respectivement à λ_1 et λ_2. Le vecteur propre associé à la plus petite valeur propre est orienté selon le contour, tandis que celui associé à la plus grande valeur propre est orienté selon l'orthogonale.

Dans la suite, nous utilisons la plus grande valeur propre et son vecteur propre associé comme composantes additionnelles à notre modèle. Le vecteur de caractéristiques associé à chaque pixel consiste ainsi en un vecteur de dimension 6 codant à la fois la couleur du pixel et sa texture locale.

À l'exception de ces vecteurs de dimension plus importante, le reste de l'algorithme reste inchangé. La figure 2 compare des résultats de segmentation des méthodes de la littérature avec la méthode que nous proposons. Malgré des labels initiaux distribués de manière grossière et/ou partielle, l'approche proposée discrimine différentes orientations de la même texture (colonne de gauche), des textures similaires avec différentes luminosité (colonne du milieu), et des textures plus complexes (colonne

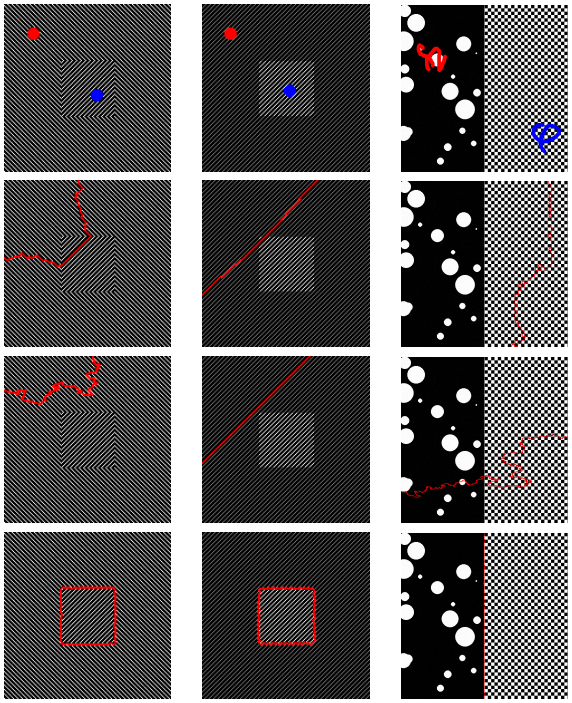


FIGURE 2 – Segmentations semi-supervisées d'images de texture. De haut en bas : labels initiaux, segmentations avec [2], avec [3], et avec notre approche.

3. https://sites.google.com/site/pierrebuyssens/code/ergc

4. http://powerwatershed.sourceforge.net/

Labels initiaux | Segmentations avec [4] | Segmentations avec [2] | Segmentations avec [3] | Méthode proposée

FIGURE 1 – Comparaison des segmentations sur les images *église* (première ligne), *paysage* (deuxième ligne), et *Oscar* (troisième ligne). De gauche à droite : labels initiaux, résultats de la segmentation avec la diffusion basée sur le gradient de l'image [4], avec une fonction de potentiel basée sur la moyenne couleur des régions [2], avec la méthode des *Power Watershed* [3], et avec notre approche basée sur une fonction de potentiel impliquant des mélanges de Gaussiennes.

de droite). Toutes les autres méthodes échouent à segmenter ces images.

4 Conclusion

Nous avons présenté une méthode de diffusion multivaluée pour la segmentation semi-supervisée d'images. Basée sur une diffusion à partir d'un ensemble de germes, l'approche proposée permet de segmenter des objets complexes modélisés à l'aide de modèles de mélange de Gaussiennes. Nous avons également démontré la flexibilité de notre approche en ajoutant de simples informations de texture de sorte que notre approche puisse efficacement discriminer différentes textures.

Références

[1] C. Alvino, G. Unal, G. Slabaugh, B. Peny, and T. Fang. Efficient segmentation based on eikonal and diffusion equations. *Int. J. of Comp. Math.*, 84(9) :1309–1324, 2007.

[2] P. Buyssens, I. Gardin, S. Ruan, and A. Elmoataz. Eikonal-based region growing for efficient clustering. *Image and Vision Computing*, 2014.

[3] C. Couprie, L. Grady, L. Najman, and H. Talbot. Power watershed : A unifying graph-based optimization framework. *PAMI*, 33(7) :1384–1399, 2011.

[4] X. Desquesnes, A. Elmoataz, and O. Lézoray. Eikonal equation adaptation on weighted graphs : fast geometric diffusion process for local and non-local image and data processing. *JMIV*, 46(2) :238–257, 2013.

[5] S. Di Zenzo. A note on the gradient of a multi-image. *Computer vision, graphics, and image processing*, 33(1) :116–125, 1986.

[6] S. Kim. An $\mathcal{O}(n)$ level set method for eikonal equations. *SIAM journal on scientific computing*, 22(6) :2178–2193, 2001.

[7] C. Rother, V. Kolmogorov, and A. Blake. Grabcut : Interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics (TOG)*, 23(3) :309–314, 2004.

[8] M. Rousson, T. Brox, and R. Deriche. Active unsupervised texture segmentation on a diffusion based feature space. In *CVPR*, volume 2, pages II–699. IEEE, 2003.

[9] J. A. Sethian. A fast marching level set method for monotonically advancing fronts. *Proceedings of the National Academy of Sciences*, 93(4) :1591–1595, 1996.

[10] V.-T. Ta, O. Lezoray, A. Elmoataz, and S. Schupp. Graph-based tools for microscopic cellular image segmentation. *Pattern Recognition Special Issue on Digital Image Processing and Pattern Recognition Techniques for the Detection of Cancer*, 42(6) :1113–1125, 2009.

[11] L. Yatziv, A. Bartesaghi, and G. Sapiro. $\mathcal{O}(n)$ implementation of the fast marching algorithm. *Journal of computational physics*, 212(2) :393–399, 2006.