Modèles Mathématiques et Représentations Discrètes pour la Description des Images Couleurs

Luc Brun

12 juillet 1999

Résumé

Nous allons présenter dans ce document, un système permettant à l'utilisateur de guider le processus de segmentation. La prise en compte des interventions de l'utilisateur et les modifications de la partition en fonction de celles-ci sont réalisées à l'aide d'un modèle permettant de coder la géométrie et la topologie d'une partition. Ce modèle, combiné à plusieurs algorithmes de segmentation ainsi qu'a diverses fonctions d'éditions, permet à l'utilisateur de désigner interactivement les régions qui doivent être découpées ou fusionnées. Cette interaction entre l'utilisateur et les algorithmes de segmentation doit permettre d'obtenir de bons résultats sur une très grande variété d'images sans ajustement de paramètres.

1 Introduction

Segmenter une image consiste à partitionner celle-ci en un ensemble de groupes de pixels connexes et homogènes, appelé une **région** de l'image. L'homogénéité est généralement calculée à partir des valeurs des pixels. On peut, par exemple, regrouper tous les pixels dont la distance colorimétrique est inférieure à un seuil. Le postulat sous-jacent de cette étape est que des groupes de pixels homogènes appartiennent au même objet. On espère donc à partir des régions obtenir les **objets** ou des parties des objets présents dans l'image. La segmentation idéale est bien sûr celle qui crée une région pour chaque objet.

Cette segmentation "idéale" est un objectif très difficile et certainement impossible à atteindre. Par exemple, si l'on segmente une image représentant une chaise, doit-on avoir une seule région représentant la chaise ou plusieurs régions décomposant la chaise avec ses pieds, son dossier, etc. ? La réponse à une telle question varie en fonction de l'utilisateur et du domaine concerné. Faute d'une définition précise du terme d'objet, de nombreux chercheurs ont développé des algorithmes de segmentation restreints à des domaines précis. Par exemple, lors du développement d'un algorithme de segmentation d'images satellites on définira précisément le concept de route et l'algorithme sera construit de façon à regouper les pixels correspondant à cette définition. Afin d'élargir le domaine des algorithmes de segmentation, on les munit souvent d'un ensemble de paramètres permettant de modifier légèrement leur comportement et donc de les adapter à des types d'images proches. Toutefois, l'adaptibilité induite par ce type de procédé reste faible et les paramètres doivent souvent être adaptés à chaque image.

L'approche que nous allons présenter permet de s'affranchir de ces limitations par une approche interactive de la segmentation. Les méthodes de segmentation utilisées sont très générales et n'ont aucune connaissance a priori des objets présents dans l'image. Ces méthodes sont utilisées en conjonction avec une structure de données permettant de coder la partition de l'image en régions. Cette structure de données est sufisamment souple pour permettre à l'utilisateur de désigner interactivement les régions qui ne sont pas assez homogènes et doivent être découpées, ou les frontières séparant deux régions d'un même objet qui doivent être fusionnées. Les découpes et fusions successives dirigées par l'utilisateur doivent conduire à la partition souhaitée par celui-ci.

Le reste du document se décompose comme suit : nous allons dans la section 2 décrire le modèle permettant de coder la partition. Nous illustrerons, dans la section 3, les possibilitées de ce modèle à l'aide d'un exemple de segmentation interactive.

2 La structure de données

Les structures de données utilisées en segmentation doivent gérer deux problèmes : l'aspect géométrique qui décrit la forme des régions et l'aspect topologique qui décrit les relations de voisinage ou d'inclusion des régions. Les modèles habituellement utilisés en segmentation [15, 16, 3] sont essentiellement dédiés à l'un ou l'autre de ces problèmes. Le modèle présenté dans cette section est basé sur un niveau géométrique et un niveau topologique coopérant pour donner une représentation unifiée de l'image segmentée.

Classiquement, une région peut être indifféremment définie par l'ensemble de ses pixels ou par sa frontière. Un important défaut des frontières définies par une séquence de pixels provient de leur inconsistence topologique [3]. Ce problème peut être résolu en utilisant **les frontières inter-pixels** (Figure 1).

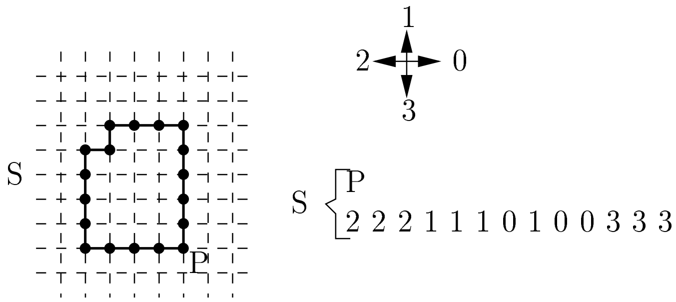


FIG. 1 – *Région définie par des chemins inter-pixels.*

Dans ce cas, la frontière d'une région est définie comme un chemin fermé 4-connexe de points demi-entiers [7] décrivant les coins des pixels (Figure 1). Cette approche introduite par Brice et Fennema [2] peut être associée à plusieurs topologies discrètes [10, 11, 8]. On peut notamment montrer qu'un chemin fermé de points demi-entiers vérifie le Théorème de Jordan. De plus, tout chemin fermé peut être décomposé en un ensemble de **segments** définis comme les chemins de longueur maximum séparant deux régions. Les extrémités des segments sont appelés des **noeuds** [7] (Figure 2).

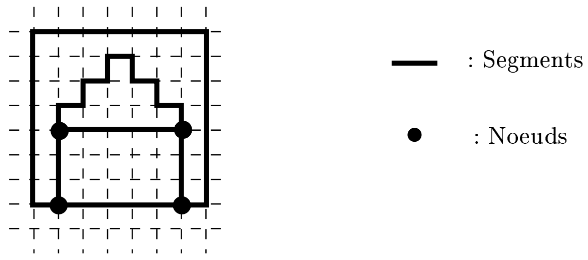


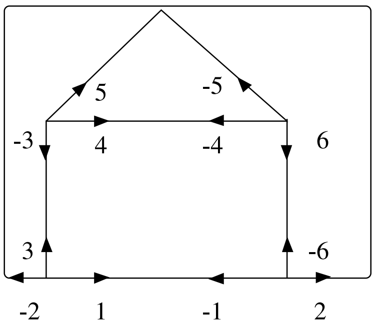
FIG. 2 – *Partition d'une image codée à l'aide de segments et de noeuds*

La représentation inter-pixels permet un calcul rapide des paramètres des frontières des régions où des intersections de régions. Toutefois, cette représentation ne fournie qu'une définition implicite des régions et l'extraction de paramètres de celle-ci peut être coûteux. Par exemple, calculer l'ensemble des régions contenues dans une région R nécessite de parcourir l'ensemble des pixels de R. Il est donc nécessaire d'associer au niveau géométrique un modèle permettant de coder de façon efficace les relations d'adjacence ou d'inclusion entre régions. Les structures de graphes telle que le Graphe d'adjacence de Régions [9] sont souvent utilisées pour coder de telles informations. Toutefois, coder simultanément les adjacences entre régions et l'ensemble des segments des régions impose l'utilisation conjointe d'un graphe des frontières et d'un graphe des régions [12, 13, 5].

Ce double codage peut être évité en utilisant les cartes combinatoires [6]. Une carte combinatoire est un couple de permutations (σ,α) définies sur un ensemble $\boldsymbol{\mathcal{D}}$ d'étiquettes appelées **brins**. La permutation α est une involution sans point fixe. Chaque brin représente une demi-arête de la carte planaire. Chaque cycle de α $(b,\alpha(b))$ représente une arête de la carte. Chaque cycle de σ est la séquence de brins rencontrés en tournant dans le sens positif (inverse de celui des aiguilles d'une montre) autour d'un sommet (Figure 3). Les cycles de σ représentent donc les sommets de la carte planaire. Une solution bien adaptée au codage des brins consiste à représenter ceux-ci par des entiers positifs et négatifs. On définit alors la permutation α par :

$$\forall b \in \boldsymbol{\mathcal{D}} \ \alpha(b) = -b$$

Ces conventions ont été utilisées dans l'exemple de la Figure 3.



σ

$$\forall b \in \{-6,\ldots,-1,1,\ldots,6\} \quad \alpha(b) = -b$$
$$\sigma = (3,-2,1)(-6,-1,2)(-5,-4,6)(5,-3,4)$$

$$\varphi = (-4,5)(4,6,-1,3)(1,2)(-5,-3,-2,-6)$$
$$\varphi(-4) = \sigma \circ \alpha(-4) = \sigma(4) = 5$$

FIG. 3 – *Carte combinatoire associée à une carte planaire.*

Le cycle d'un brin b par une permutation π est noté $\pi^*(b)$. Par exemple, le cycle $\sigma^*(-4)$ représenté sur la Figure 3, est égal à $(-4,6,-5)$. On a donc $\sigma^*(-4) = \sigma^*(6) = \sigma^*(-5)$.

Les faces d'une carte combinatoire correspondent aux cycles de la permutation $\varphi = \sigma \circ \alpha$ (Figure 3). Afin de pouvoir désigner les faces, nous associons une étiquette à chaque brin. Nous introduisons donc une fonction d'étiquettage de face, λ, constante sur chaque cycle de φ. Une telle fonction vérifie la condition suivante :

$$\forall b,b' \in \boldsymbol{\mathcal{D}} \ \ \lambda(b) = \lambda(b') \Leftrightarrow \varphi^*(b) = \varphi^*(b')$$

Afin d'accéder rapidement au cycle d'une face à partir de son étiquette, nous définissons également une fonction d'étiquettage inverse λ^{-1}, définie par :

$$\forall f \in \boldsymbol{\mathcal{F}} \quad b = \lambda^{-1}(f) \Rightarrow \forall b' \in \varphi^*(b) \quad \lambda(b') = f$$

où $\boldsymbol{\mathcal{F}}$ est égale à l'image de $\boldsymbol{\mathcal{D}}$ par λ et correspond à l'ensemble des étiquettes de faces.

La fonction λ permet d'associer à chaque brin b la face $\lambda(b)$ qui le contient et, inversement, la fonction λ^{-1} permet de retrouver un brin $\lambda^{-1}(f)$ de chaque face f. Les autres brins du cycle peuvent être retrouvés en calculant le cycle $\varphi^*(\lambda^{-1}(f))$.

Si nous utilisons une orientation positive pour la permutation σ, tous les cycles de φ, excepté un, sont orientés dans le sens des aiguilles d'une montre [6]. Les faces orientées dans le sens des aiguilles d'une montre sont appelées des faces finies. Inversement, la face orientée dans le sens positif est appelée la face infinie et est notée f_φ^∞. Cette face correspond à l'ensemble du plan moins les faces finies. Nous utilisons les faces infinies pour gérer les relations d'inclusion qui ne sont pas définies dans le formalisme des cartes planaires. Ces relations d'inclusion sont définies à l'aide de deux fonctions **father** and **child** qui associent, respectivement, à chaque face infinie la face finie qui la contient et à chaque face finie la liste des faces infinies contenues dans celle-ci.

Le lien entre l'ensemble des segments et des noeuds définissant le niveau géométrique de notre modèle et les cartes planaires codant le niveau topologique est effectué en associant :

- les noeuds et les sommets,
- les segments et les arêtes,
- les régions et les faces.

Partant d'une partition d'une image, on construit sa représentation géométrique en créant p noeuds et q segments. On crée alors un sommet pour chaque noeud et une arête pour chaque segment. Une arête relie deux sommets si et seulement si les noeuds correspondants sont les extrémités du segment correspondant à l'arête. On peut montrer [7] que les graphes ainsi constitués sont bien des cartes planaires, donc des cartes combinatoires.

3 Les algorithmes de segmentation

Nous avons vu dans la section 2 que notre modèle permettait de coder une partition 4-connexe. Cette partition peut être définie explicitement par un tableau de labels ou implicitement par une fonction indiquant si deux pixels adjacents appartiennent à la même région. La mise à jour de notre modèle à partir d'une définition implicite de la partition permet d'utiliser dans une même session différents algorithmes de segmentation. Il est de plus possible de mettre à jour une partition définie par une première segmentation [3]. L'utilisateur peut donc appliquer différentes méthodes de segmentation sur différentes partie de l'image et donc adapter le traitement au type de région à traiter.

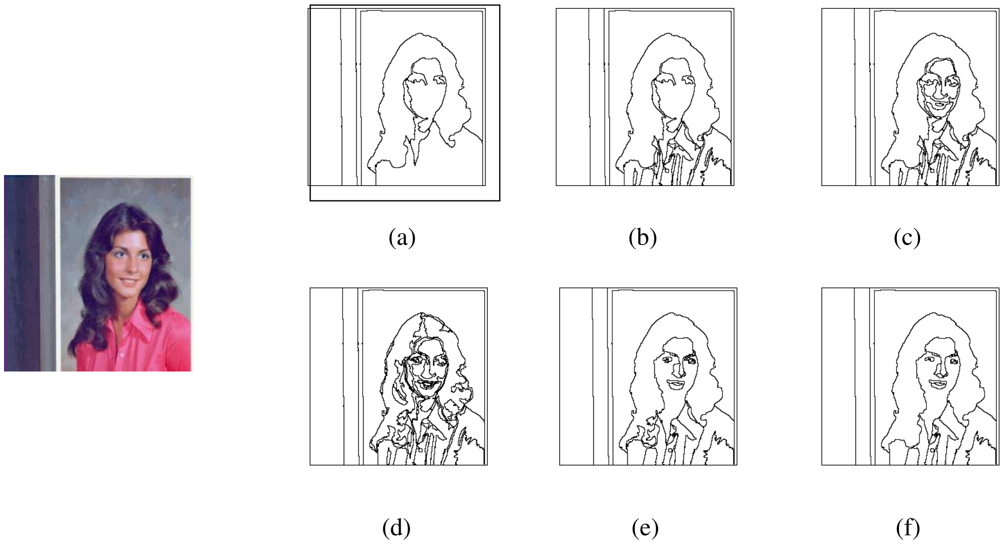


FIG. 4 – *Un exemple de session interactive*

La session interactive présentée sur la Figure 4 a été obtenue par des méthodes de segmentation utilisant des algorithmes de quantification [19, 1, 20]. Ces algorithmes permettent d'extraire les principales couleurs d'une région. L'ensemble des couleurs ainsi calculé est appellé l'ensemble des couleurs représentatives. Cet ensemble nous fournie un partitionnement implicite de la région : en effet, si $\{c_1, \dots, c_K\}$ désigne l'ensemble des couleurs représentatives, le prédicat $\mathcal{P}$ indiquant si deux pixels adjacents P_1 et P_2 appartiennent à la même région est donné par :

$$\mathcal{P}(P_1,P_2) \Longleftrightarrow ArgMin_{j\in\{1,\ldots,K\}} d(I(P_1),c_j) = ArgMin_{j\in\{1,\ldots,K\}} d(I(P_2),c_j) \quad (1)$$

où $I(P_1)$ et $I(P_2)$ désignent la couleur des pixels P_1 et P_2 et $ArgMin_{j\in\{1,\ldots,K\}} d(c,c_j)$ représente l'indice de la couleur représentative la plus proche de la couleur c.

Le prédicat $\mathcal{P}$ donné par l'équation 1 peut être calculé par des méthodes d'inversion de tables de couleurs [17, 14] ou en utilisant des structures de données générées par les algorithmes de quantification. L'ensemble des régions ainsi formé est alors soumis à un algorithme de fusion de régions [4] qui supprime les régions les moins significatives.

La Figure 4(a) montre le résultat de la première étape de segmentation appliquée sur toute l'image. L'utilisateur obtient la partition représentée sur la Figure 4(b) en sélectionnant le chemisier et en appliquant à nouveau une segmentation sur cette région. De même, la partition de l'Image 4(c) est obtenue en sélectionnant le visage et en y appliquant une nouvelle segmentation. Cette dernière étape permet d'extraire du visage les zones de fort contraste tel que les yeux ou les sourcils. Une seconde étape de segmentation est donc nécessaire pour extraire certaines parties peu contrastées comme le nez. La partition obtenue est représentée sur l'Image 4(d). Plusieurs suppressions de segments (voir l'Image 4(e)) permettent à l'utilisateur de supprimer les régions du visage correspondant à des zones d'ombre qui ne sont pas intéressantes d'un point de vue sémantique. Finalement, une dernière segmentation suivie d'une suppression permet de corriger la partie gauche du nez qui avait été partiellement mal segmentée. Notons que les régions représentant le fond de l'image et le rideau situé sur la gauche ont été correctement segmentées dès les premières étapes de l'algorithme. L'utilisateur n'a donc pas considéré ces régions. De même, si certaines régions de l'image présentent peu d'intérêt pour l'utilisateur, celui-ci peut les ignorer après leur extraction. Ce dernier point peu améliorer considérablement les temps de calculs lorsque l'on travaille sur des images de taille importante.

4 Conclusion

Nous avons présenté dans ce document, un modèle permettant de coder simultanément la géométrie et la topologie d'une partition. Ce modèle permet de définir ou de modifier une partition par différentes méthodes de segmentation. Il permet de plus une interaction entre l'utilisateur et les différents algorithmes de segmentation. Celui-ci a en effet la possibilité de désigner interactivement les régions à raffiner ou fussionner. Les interventions de l'utilisateur permettent de guider le processus de segmentation afin d'obtenir la partition désirée.

La conception d'un algorithme de segmentation est légèrement différente selon qu'il sagit de concevoir une méthode automatique ou interactive. Par exemple, les algorithmes de quantification sont peu adaptés à la génération automatique de partitions. En revanche, le fait qu'ils renvoient des résultats d'autant plus précis qu'ils sont appliqués sur des ensembles restreints est très utile dans le cadre de la segmentation interactive. En effet, appliqués sur des régions de plus en plus petites, les algorithmes de quantification donneront des résultats de plus en plus précis sans ajustement de paramètres. Inversement, des méthodes comme les watersheds [18] dont le résultat ne dépend pas de la taille de l'image sont assez peu adaptées à une utilisation interactive. L'explicitation des contraintes spécifiques des algorithmes de segmentation interactifs et l'adaptation d'algorithmes existant à ces contraintes est une voie de recherche intéressante.

De plus, notre système permettant d'appliquer plusieurs algorithmes de segmentation lors d'une même session, la conception d'algorithmes capable d'analyser une région et d'appliquer sur celle-ci l'algorithme de découpe le plus adapté est une voie de recherche difficile mais certainement prometteuse.

Références

[1] J. P. Braquelaire and L. Brun. Comparison and optimization of methods of color image quantization. *IEEE Transactions on Image Processing*, 6(7):1048–1052, july 1997.

[2] R. Brice and C.L. Fennema. Scene analysis using regions. *Artificial intelligence*, 1:205–226, 1970.

[3] L. Brun. *Segmentation d'images couleur à base Topologique.* PhD thesis, Université Bordeaux I, 351 cours de la Libération 33405 Talence, December 1996.

[4] L. Brun and J. P. Domenger. A new split and merge algorithm with topological maps and inter-pixel boundaries. In *The fifth International Conference in Central Europe on Computer Graphics and Visualization*, february 1997.

[5] Luc Brun and Walter Kropatsch. Dual contraction of combinatorial maps. In *GbR99.* IAPR TC15, 1999.

[6] R. Cori. *Un code pour les graphes planaires et ses applications.* PhD thesis, Université Paris VII, 1975.

[7] J. P. Domenger. *Conception et implémentation du noyau graphique d'un environnement $2D\frac{1}{2}$ d'édition d'images discrètes.* PhD thesis, Labri Université Bordeaux I, 351 cours de la libération 33405 Talence, avril 1992.

[8] C. Fiorio. *Approche interpixel en analyse d'images : une topologie et des algorithmes de segmentation.* Thèse de doctorat, Université Montpellier II, 24 novembre 1995.

[9] Claudia Fuchs and Wolfgang Förstner. Polymorphic Grouping for Image Segmentation. In Eric Grimsen, editor, *5th ICCV 95*, pages 175–182, 1995.

[10] E. Khalimsky, R. Kopperman, and P.R. Meyer. Boundaries in digital planes. *Journal of applied Math. and Stochastic Analysis*, 3:27–55, 1990.

[11] V.A. Kovalevsky. Finite topology as applied to image analysis. *Computer Vision, Graphics, and Image Processing*, 46:141–161, 1989.

[12] Walter G. Kropatsch, Mark Burge, Souheil Ben Yacoub, and Nazha Selmaoui. Dual graph contraction with LEDA. *Computing, Supplementum: Graph Based Representations in Pattern Recognition*, No. 12:pp. 101–110, 1998.

[13] Walter G. Kropatsch, Mark Burge, and Heinz L. Idl. Dual Graph Contraction for Run Graphs. In Aleš Leonardis and Franc (eds.) Solina, editors, *Computer Vision - CVWW'98, Proceedings of the Computer Vision Winter Workshop*, pages pp. 75–86, Ljubljana, 1998. IEEE Slovenia Section.

[14] Brun Luc and Claude Secroun. A fast algorithm for inverse color map computation. *Computer Graphics Forum*, 17(4):263–271, December 1998.

[15] A. Rutovitz and J. L. Pfaltz. Data structures for operations on digital images. *Pictorial Pattern Recognition*, pages 105–133, 1968.

[16] S. L. Tanimoto. Image data structures. In S. L. Tanimoto and A. Klinger, editors, *Structured Computer Vision*, pages 31–56. Academic Press, New York, 1980.

[17] S. W. Thomas. Efficient Inverse Color Map Computation. In J. Arvo, editor, *Graphics Gems II*, pages 116–125, 528–535. Academic Press Professional, Cambridge, MA, 1991.

[18] L. Vincent and P. Soille. Watersheds in digital spaces : and efficient algorithm based on immersion simulations. *IEEE Trans. PAMI*, 13(6):583–598, 1991.

[19] Xiaoling Wu. Color quantization by dynamic programing and principal analysis. *ACM Transaction on Graphics*, 11(4):348–372, 1992.

[20] Zhigang Xiang. Color image quantization by minimizing the maximum intercluster distance. *ACM Transactions on Graphics*, 16(3):260–276, July 1997.