Un Atelier Logiciel pour la conception d'applications de segmentation d'images

Régis Clouard, Marinette Revenu, Abderrahim Elmoataz & Christine Porquet

Groupe de Recherche en Informatique, Image et Instrumentation de Caen, URA-CNRS 1526
GREYC - ISMRA, 6 bd Maréchal Juin F-14050 Caen Cedex France
E-mail : Regis.Clouard@greyc.ismra.fr

RÉSUMÉ

Nous présentons ici un atelier logiciel dédié à la conception interactive d'applications de segmentation d'images. Un utilisateur spécifie les objectifs à atteindre relativement à un contexte d'application, et le système compose alors un programme applicatif, par assemblage d'opérateurs paramétrés prédéfinis. Ce programme se presente sous la forme d'un graphe d'opérateurs hétérogène, capable d'adapter son comportement aux spécificités de chacune des images traitées. Pour obtenir un graphe d'opérateurs répondant aux spécifications d'un utilisateur, nous utilisons un modèle de planification d'opérateurs hiérarchique et incrémentale, faisant intervenir des connaissances à cinq niveaux d'abstraction.

ABSTRACT

In this paper, a software workbench dedicated to the interactive conception of image segmentation applications is presented. The user specifies objectives to be reached according to some application context and the system can then produce a run-time program, by putting together predefinied parametrized operators. This program can be seen as a heterogeneous graph of operators, that can adapt its behavior to the peculiarities of each image to be segmented. The building of graphs of operators answering the user's objectives is achieved by a hierarchical and incremental process involving five levels of abstraction.

1. INTRODUCTION

La conception de systèmes généraux pour la segmentation d'images est au coeur de nombreux travaux de recherche [1], [2] ou dans le cas du Traitement du Signal [3]. L'enjeu est bien évidemment de proposer des systèmes plus flexibles et plus fiables dont le comportement dépend étroitement de la tâche à réaliser et des images traitées. Les utilisateurs de tels systèmes n'ont plus alors à connaître les algorithmes et peuvent donc se focaliser sur les spécifications des objectifs à atteindre ainsi que sur la description du contexte de l'application.

Cette approche nécessite des architectures logicielles permettant, d'une part de faire cohabiter des connaissances procédurales et déclaratives, et d'autre part des modèles de résolution de problèmes intégrant l'utilisateur.

Nous présentons ici un atelier logiciel dédié à la conception d'applications de segmentation selon deux points de vue :

– de manière automatique lorsque le domaine de l'image à traiter est suffisamment connu,

– de manière interactive pour le développement incrémental d'applications, lorsque l'on aborde un nouveau domaine.

La cohabition de ces deux aspects dans l'atelier est motivée par le souci de traiter de vraies applications, tout en s'affranchissant le plus possible des techniques de segmentation pour se focaliser sur l'explicitation des connaissances de segmentation d'images. Ce n'est qu'en s'attaquant à des problèmes concrets que l'on peut préciser la nature des vrais problèmes et qu'une réelle participation des experts du domaine d'application peut être envisagée.

Ce travail se situe dans le cadre plus général du pilotage de code informatique en vue de la réutilisation de programmes. Avec des tels environnements, les utilisateurs n'ont plus à connaître les algorithmes de segmentation, et ils peuvent de ce fait se concentrer sur la définition des objectifs à atteindre ainsi que sur la description du contexte de leur application. Néanmoins, il faut encore posséder de solides compétences en segmentation, ne serait-ce que pour définir les bonnes tâches et les bons indices visuels [4].

Dans cette communication, nous commençons par présenter le principe de conception d'applications adopté, qui est basé sur la production d'un graphe d'opérateurs répondant à une requête formulée par un utilisateur. Nous détaillons ensuite cette notion de graphes d'opérateurs, puis décrivons la façon de les construire par un processus de planification hiérarchique et incrémental.

2. PRINCIPE DE CONCEPTION D'APPLICATIONS

D'une façon générale, concevoir une application de segmentation d'images, c'est partir des spécifications d'une requête exprimée par des utilisateurs en termes de tâches à accomplir sur une classe d'images, pour aboutir à la production d'un programme informatique adapté. Le programme ainsi défini est destiné à produire, par transformations successives des images données en entrée, de nouvelles images qui mettent en évidence l'information recherchée.

Dans l'approche que nous proposons, la construction du programme se fait de manière dynamique, par assemblage d'opérateurs prédéfinis. Les opérateurs sont des briques de base, codés une fois pour toutes et regroupés dans une bibliothèque. Le programme se présente alors sous la forme d'un graphe d'opérateurs. Ce sont les spécifications des intentions et du contexte de l'application décrites dans les requêtes qui interviennent à tous les niveaux pour orienter la sélection, la paramétrisation et l'enchaînement des opérateurs de la bibliothèque et construire les graphes correspondants.

Dans ce cadre, notre atelier logiciel peut être vu comme un outil d'expérimentation favorisant le dialogue et la coopération entre les différents intervenants dans l'analyse d'applications de segmentation : les expert du domaine d'application (biologistes, géographes) et les experts en segmentation d'images. La conception d'une application se fait au travers d'une démarche en trois étapes :

1. L'étape ***d'initialisation*** consiste à acquérir une vision réaliste des difficultés des traitements à effectuer et à identifier les indices perceptifs. Les utilisateurs formulent une première version de la requête à appliquer sur une image représentative de la classe étudiée. Le système propose alors une première version du graphe d'opérateurs, et produit les premières images résultantes.

2. Au vu de ces premiers résultats, les utilisateurs sont alors capables de préciser leurs exigences, au cours de l'étape ***d'amélioration***. La requête est peu à peu affinée par analyse des différents résultats produits sur une même image.

3. Une fois que l'on dispose de résultats satisfaisants sur l'image de départ, l'étape de ***validation*** consiste à soumettre ce même graphe à toute une série d'images de la classe, dans le but de tester sa pertinence. Au besoin, la requête est alors encore affinée de manière à prendre en compte de nouvelles particularités non encore examinées.

A l'issue de cette dernière étape, on peut espérer que le graphe d'opérateurs obtenu représente effectivement l'application étudiée. A partir de ce graphe, un générateur de code C++ permet de créer le programme exécutable équivalent.

3. NOTION DE GRAPHE D'OPERATEURS

Un graphe définit une chaîne de transformations d'images. Les noeuds caractérisent des opérateurs de Traitement d'Images exécutables et les liens correspondent aux flux de données. Dans un tel graphe, les opérateurs sont enchaînés de façon à ce que les images de sortie de l'un servent d'images d'entrée à ceux qui lui succèdent. L'exécution du graphe résulte alors de l'exécution de chacun de ses opérateurs dans l'ordre défini par les liens. Les opérateurs considérés ici sont des fonctions de transformation d'images de la forme :

$$v=opérateur(paramètre^*, [masque], source^*, destination^*).$$

La notion d'image est prise ici au sens large et représente aussi bien des images de pixels que des cartes de régions, des graphes de régions, des histogrammes. Les paramètres permettent d'adapter le comportement de l'opérateur aux spécificités des objectifs et des images. Le code de retour v fournit une valeur ou un vecteur de valeurs numériques (*valeurs de seuils, nombre de régions*). Il est important de noter qu'un graphe est générique de l'application pour laquelle il a été développé, c'est-à-dire, qu'il doit être capable de traiter avec la même efficacité toutes les images de la classe. Il faut pour cela qu'il possède des capacités d'auto-configuration, lui permettant d'adapter son comportement aux spécificités de chacune des images. Matsuyama [1] identifie trois mécanismes qui permettent de concevoir des graphes plus robustes, plus souples, capables de s'adapter à la variabilité et l'hétérogénéité des images. Ces mécanismes sont en fait des **macro-opérateurs** qui permettent l'exécution des opérateurs selon différents schémas :

– $mask(op, M, Ie^*, Is^*)$ permet de n'appliquer l'opérateur op que sur les pixels des images d'entrée Ie non masqués par la carte de régions M, pour construire les images de sortie Is.

– $combine(op_1,...,op_n,C)$ permet de combiner le résultat de plusieurs opérateurs, $op_1,...,op_n$, selon une relation C (logique, spatiale, etc.).

– $optimize(op,p,v_1,v_n,f_e,I)$ permet de rechercher la valeur du paramètre p, en procédant par optimisation de la fonction d'évaluation f_e sur les différents résultats obtenus à partir des valeurs prises dans l'intervalle $[v_1,v_n]$, et sur une image de référence I.

Nous avons repris et développé ces trois mécanismes. Le masquage est une partie intégrante de nos opérateurs. Ils prennent tous une carte de régions optionnelle comme masque, ce qui permet à chaque opérateur de s'appliquer aussi bien sur l'image entière, que sur une partie masquée par la carte de régions. La recherche dynamique des valeurs de paramètres est étendue à des exécutions de type *while* (prendre le dernier résultat satisfaisant la fonction f_e), *until* (prendre le premier résultat satisfaisant la fonction f_e), et *for* (exécuter n fois consécutives l'opérateur).

Mask et *combine* permettent d'implémenter des stratégies d'analyse descendante, parce qu'ils offrent des mécanismes de focalisation sur des parties d'images. *Optimize, while, until, for* offrent la possibilité de calculer dynamiquement les valeurs de paramètres sur la base de mesures faites directement sur les images. Grâce à ces mécanismes, les graphes sont des compositions hétérogènes d'opérateurs (Figure 1). Les flux de données entre opérateurs sont des images, des valeurs numériques utili-

sées comme base pour le calcul de valeurs de paramètres, des cartes de régions représentant des masques des données, et des images servant d'images de référence pour les fonctions d'évaluation.

On notera que les opérateurs ne sont pas seulement ordonnés selon l'ordre d'échange des images d'entrée et de sortie, mais aussi en tenant compte de la façon d'obtenir les valeurs de paramètres, les masques de régions et les images de référence.

4. PLANIFICATION DE GRAPHES D'OPERATEURS

Nous proposons une planification des graphes d'opérateurs selon un processus hiérarchique descendant, comprenant quatre étapes :

1. En fonction de la nature du problème à résoudre, il faut faire le choix d'une stratégie de segmentation adaptée (*descendante, ascendante, mixte*).

2. En fonction des particularités du problème, des types d'indices visuels et des contraintes à respecter, il faut faire le choix de techniques pour implanter chacun des éléments de la stratégie retenue, avant de faire le choix effectif des opérateurs et de leurs valeurs de paramètres.

3. Une fois que l'on dispose d'un graphe d'opérateurs, le contrôle de son exécution se fait en utilisant les mécanismes de masquage, de combinaison et d'optimisation de paramètres.

4. Mais ce contrôle n'est pas suffisant pour garantir la pertinence du graphe, il faut pouvoir en plus évaluer sa qualité globale, c'est-à-dire comparer les résultats obtenus avec ceux attendus, et éventuellement apporter des améliorations.

4.1. Description des requêtes

Avec une telle approche, la requête revêt un caractère primordial pour guider le processus de résolution. Elle comporte deux parties : la spécification des objectifs et la caractérisation du contexte de l'application qui permet de donner un sujet à l'image. Les utilisateurs précisent leurs intentions en choisissant parmi des listes prédéfinies puis en paramétrant les tâches à accomplir et les contraintes à respecter. Les tâches font globalement référence aux objectifs de la segmentation (*extraire, isoler, partitionner,...*) et les contraintes définissent les niveaux de détail à atteindre (*localisation des contours précise*), les critères à optimiser (*contours de type « marche »*), et les restrictions à respecter (*ne pas séparer les objets qui se touchent*).

Le contexte de l'application, quant à lui, doit être décrit par une liste d'attributs-valeurs comportant trois niveaux de description : ***physique*** (*les conditions d'acquisition, la nature du capteur*), ***perceptif*** (*le modèle de contours et de régions*) et ***sémantique*** (*les caractéristiques et relations des objets représentés*).

4.2. Modèle de la connaissance

La planification du graphe d'opérateurs se fait selon un processus hiérarchique de décomposition de tâches d'un niveau en un enchaînement de sous-tâches plus élémentaires au niveau inférieur. Cinq niveaux d'abstraction sont utilisés; nous distinguons par ordre décroissant les niveaux ***requête***, ***tâche***, ***fonctionnalité***, ***procédure*** et ***opérateur***.

L'évaluation des résultats suit, elle, le processus hiérarchique inverse. Les résultats de l'exécution des opérateurs sont remontés le long des liens de décomposition jusqu'aux tâches de plus haut niveau. A chaque étape de la remontée, les résultats de la décomposition sont évalués par des règles locales à la tâche. Si celle-ci est jugée non satisfaite, alors elle est corrigée en essayant une nouvelle décomposition s'il en existe.

Un plan de segmentation est représenté par un arbre ET, PUIS de tâches à cinq niveaux (Figure 2). Une tâche est caractérisée par un but à atteindre, des contraintes, des images d'entrée, des images de sortie, et par des règles d'évaluation si elles existent.

Les décompositions de tâches en sous-tâches peuvent être effectuées, soit par Sources de Connaissances (SC) selon le modèle du Blackboard [5], lorsque le système fonctionne en mode automatique, soit par l'utilisateur s'il est suffisamment compétent [6], lorsque le système est en mode manuel. A tout instant, l'utilisateur a la possibilité de passer d'un mode à l'autre.

Une SC décrit de manière déclarative, une façon unique d'effectuer une décomposition d'une tâche donnée en un enchaînement de sous-tâches au niveau inférieur. Il y a donc au-

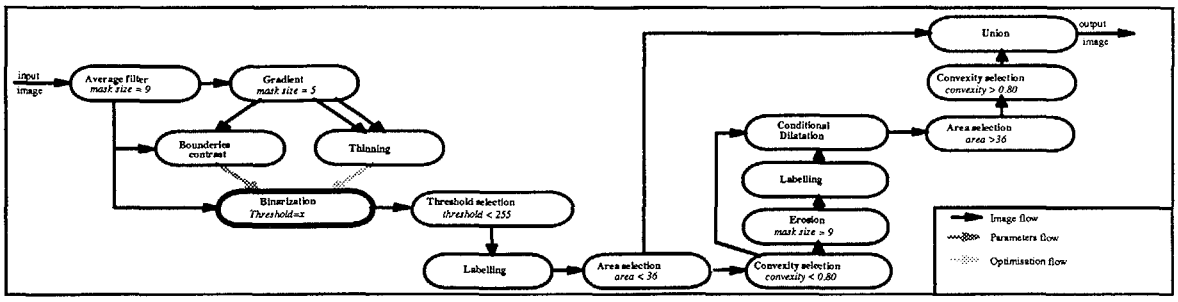

Figure 1 : Un exemple de graphe d'opérateurs pour une application décrite dans [7].

L'opérateur binarization *est exécuté en mode optimisation, par recherche du maximum de coïncidence entre les frontières des régions obtenues et les contours amincis obtenus par l'opérateur* thinning. *L'opérateur* erosion *est, lui, exécuté en mode itération. Le nombre d'itérations est déterminé par la valeur en nombre de pixels du rayon minimum des objets de la scène.*

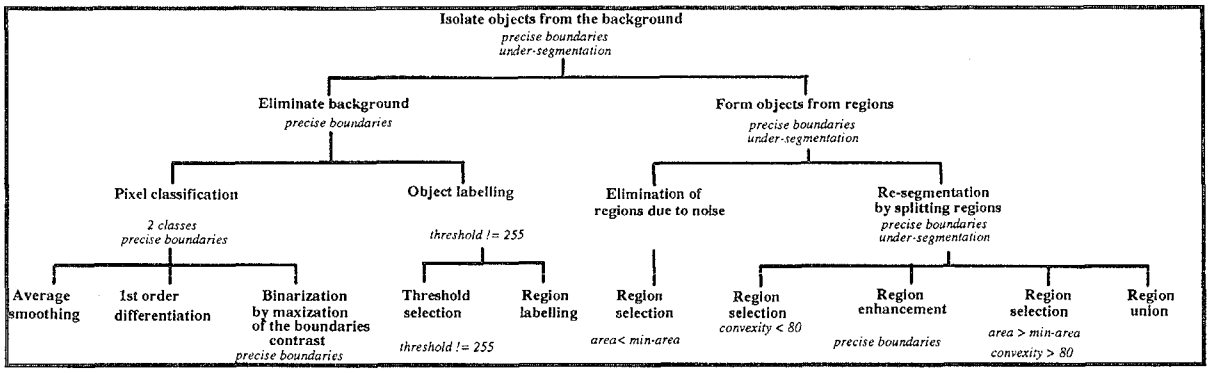

Figure 2 : Exemple des quatre premiers niveaux du plan qui a permis de produire le graphe d'opérateurs de la Figure 1

tant de SCs que de décompositions possibles. Pour chaque sous-tâche créée, il faut préciser le but à atteindre, les contraintes, l'endroit où récupérer les images d'entrée, et les éventuelles règles d'évaluation. Une SC a un format condition-action, dans lequel la condition est décomposée en deux, une partie déclencheur qui décrit le type de problème que la SC peut contribuer à résoudre, et une partie précondition qui définit les conditions de son application. La partie action agit sur le plan de segmentation en créant, supprimant ou modifiant des tâches dans le plan.

Les SCs de planification sont de la forme :

```
déclencheur: ∃ une tâche T avec le but B
condition: la tâche T n'est pas encore résolue
action: Si <le contexte possède la valeur y>
     ou <T possède la contrainte z>
  alors
     <créer la décomposition de la tâche T
     en un enchaînement donné de sous-tâches
     ayant chacune un but, des contraintes,
     des règles d'évaluation et un chemin pour
     récupérer les images d'entrée>
  sinon
     <créer la même décomposition mais avec
     des valeurs de contraintes et des règles
     d'évaluation différentes>
```

Lorsqu'il y a plusieurs SCs pour décomposer une même tâche, le choix de la « meilleure » est effectué soit par le module de contrôle [5], soit par l'utilisateur lui-même selon le mode de résolution en cours. On trouvera dans [5] le détail du modèle de contrôle retenu, qui s'inspire fortement de celui de BB1 [5].

5. CONCLUSION

Notre travail s'inscrit dans le cadre du pôle « Traitement et Analyse d'Images » de Caen. Ce pôle couvre un large domaine d'applications : le biomédicale, les sciences des matériaux, la géologie, etc. Il est un champ d'investigation privilégié pour étudier et affiner les concepts proposés et faire coopérer différents types d'experts.

Notre Atelier a été validé sur différentes applications, principalement d'origine biomédicale : analyse d'images de cytologie et d'histologie de l'oesophage, pour lesquelles différents plans de segmentation ont été construits. Le plan de la Figure 2 est un exemple de tels plans, qui fait intervenir 17 SCs de décomposition et 15 opérateurs. Sur cette même application, trois plans différents ont été expérimentés et évalués.

La méthodologie proposée est basée sur l'expérience acquise sur le terrain et tente d'apporter une réponse aussi satisfaisante que possible à la problématique : « comment concevoir de nouvelles applications de façon rapide et efficace, et répondant au mieux aux exigences des utilisateurs ? ».

REFERENCES

[1] Matsuyama, T. (1989). Expert systems for Image Processing : Knowledge-based Composition of Image Analysis Processes. *CVGIP*, 48: 22-49.

[2] Clément, V. & M. Thonnat (1993). A knowledge-based approach to integration of image processing procedures. *CVGIP image understanding*. 27: 166-184.

[3] Shekhar, C., S. Moisan & M. Thonnat (1994). Towards an intelligent problem solving environment for signal processing. *Mathematics and computers in Simulation*. 36: 347-359.

[4] Dalle, P., P. Dejean & J.M. Inglebert (1993). Méthodologie de conception d'applications de Traitement d'Images, *4e Journées ORASIS*, Mulhouse, 92-95.

[5] Clouard, R., C. Porquet & M. Revenu (1993). Resolution of image processing problems by dynamic planning within the framework of the Blackboard model, *SPIE Intelligent Robots and Computer Vision XII*, Vol 2056, Boston, USA, 419-429.

[6] Clouard, R., M. Revenu, A. Elmoataz & C. Porquet (1995), A Software Workbench for Knowledge Acquisition and Integration in Image Processing, *International Workshop on the Design of Cooperative Systems*, Juan-Les-Pins, 298-312.

[7] Clouard, R., A. Elmoataz, C. Porquet & M. Revenu (1995). BORG : a Knowledge-based system for the automation of Image Segmentation Task. *In Proc. Fifth Internat. conf. on Image Processing and its applications*, Edinburgh, UK, Juillet.

[8] Hayes-Roth, B., (1985) *A blackboard architecture for control*, Artificial Intelligence, 26: 251-321.