Conception d'applications de traitement d'Images par Raisonnement à partir de Cas : définition, utilisation et gestion de cas

Valérie Ficet-Cauchard, Marinette Revenu, Christine Porquet

▶ **To cite this version:**

Valérie Ficet-Cauchard, Marinette Revenu, Christine Porquet. Conception d'applications de traitement d'Images par Raisonnement à partir de Cas : définition, utilisation et gestion de cas. RàPC' 99 Plate Forme AFIA, 1999, Palaiseau, France. pp.7-16. hal-00869813

HAL Id: hal-00869813
https://hal.science/hal-00869813

Submitted on 20 Jan 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Conception d'applications de traitement d'images par raisonnement à partir de cas : définition, utilisation et gestion de cas

Valérie FICET-CAUCHARD, Marinette REVENU et Christine PORQUET

GREYC-ISMRA – 6 Bd du Maréchal Juin – F14050 Caen cedex, http://www.greyc.ismra.fr

Résumé

Le RàPC est encore très peu utilisé dans le domaine du Traitement d'Images (TI). S'il existe des systèmes de RàPC résolvant des problèmes de diagnostic à partir d'images segmentées, on ne trouve pas de systèmes de conception de programmes de TI basés sur le RàPC. Ce type de raisonnement est pourtant bien adapté à des domaines mal formalisés, tels que le TI, puisqu'il permet la représentation des exceptions, l'utilisation d'informations manquantes et la résolution d'un problème complexe par combinaison de solutions de problèmes plus simples.

Nous proposons un système d'aide à la conception d'applications de TI utilisant une approche basée sur le RàPC. Dans cet article, nous définissons la notion de « cas en TI » et proposons des types de critères et des fonctions génériques associées à la caractérisation d'un cas. Puis nous détaillons le fonctionnement des différentes phases de notre module de RàPC : sélection d'un cas, adaptation récursive et mémorisation de nouveaux cas. Ces différentes étapes sont illustrées sur un exemple concret d'application.

1. Introduction

L'application du Raisonnement à Partir de Cas (RàPC) à l'analyse d'images reste peu fréquente. Dans les systèmes existants, le module de RàPC se focalise sur la phase d'interprétation d'images ; la phase de traitement, réalisant l'extraction des objets présents, est effectuée en aval. Cette limitation à la phase d'interprétation résulte du fait que ces systèmes sont dédiés à un type d'image particulier sur lequel on effectue toujours le même traitement (à l'exception du réglage des paramètres). Parmi ces systèmes on peut citer SICT-WICT [11], dédié à l'interprétation d'images IRM du thorax, et le système de Perner [15], dédié à la détection de défauts dans des images de composants métalliques. Le problème est ici défini par la donnée d'une image déjà segmentée et la solution consiste en l'interprétation de cette image en termes du domaine de l'image. Notre objectif est de réaliser un système de RàPC appliqué au Traitement d'Images (TI) plus général que les systèmes existants : notre système doit également gérer la phase de traitement des images de façon à s'appliquer à tout type d'analyse ou d'image. Selon notre point de vue, le problème est défini par une image et par une requête, et la solution par un programme de TI. Le module de RàPC, présenté dans cet article, est intégré dans TMO, qui est un système dédié d'une part à l'acquisition interactive des connaissances et d'autre part à leur réutilisation. L'architecture de ce système est détaillée dans [8] et [9].

Dans la section 2, nous décrivons le cadre de nos travaux en rappelant nos objectifs par rapport au TI et en décrivant brièvement l'approche et les modèles choisis pour la représentation des applications. Les sections 3 et 4 sont consacrées au module de RàPC proprement dit. Nous commençons par décrire ce que nous entendons par « cas en TI » ainsi que les fonctions utilisées pour le calcul de la similarité (section 3). Ensuite, nous décrivons l'algorithme de recherche/adaptation en précisant comment se déroulent la sélection d'un cas, l'adaptation récursive de la solution sélectionnée et la mémorisation des nouveaux cas créés en illustrant chacune de ces phases sur un exemple concret (section 4).

2. Cadre des travaux : le système TMO

L'objectif initial de nos travaux est de représenter et de structurer les connaissances de différents experts en TI pour leur permettre de les partager et de les réutiliser. Nous proposons pour cela un système interactif facilitant l'acquisition des connaissances auprès de l'expert au fur et à mesure de leur mise en évidence par la réalisation d'applications. Dans cette section, nous présentons tout d'abord l'approche choisie pour la mise au point des applications de TI. Nous décrivons ensuite le modèle utilisé pour représenter ces applications. Enfin, nous décrivons brièvement les fonctionnalités intégrées pour la création d'applications par acquisition interactive de connaissances et pour l'exécution d'applications.

2.1. Représentation des applications par plans hiérarchiques

L'approche que nous avons choisie pour la mise au point d'applications de TI est fondée sur « l'exploitation intelligente de bibliothèques d'opérateurs ». Un opérateur est un programme qui effectue une opération de base sur une/des image/s. Il prend en entrée la/les image/s à traiter ainsi que des paramètres et il fournit en sortie une/des images et/ou des résultats numériques ou symboliques. L'intérêt d'utiliser de telles bibliothèques est que la construction d'une application consiste à enchaîner et à paramétrer des opérateurs. L'utilisateur peut donc

s'abstraire du code informatique et réaliser une programmation au « niveau connaissance ».

Une première famille de systèmes facilitant l'utilisation de telles bibliothèques regroupe les environnements de programmation graphique. Ces systèmes [12] [17] proposent à l'utilisateur de sélectionner et d'enchaîner des opérateurs mais ils ne lui permettent pas d'expliquer ni de modéliser son raisonnement. Il est donc difficile de réutiliser les éléments de solution construits antérieurement pour d'autres applications.

Une deuxième famille de systèmes de TI utilisant des bibliothèques de programmes est celle des planificateurs automatiques tel que OCAPI [4] ou BORG [5], le générateur de plans mis au point dans notre laboratoire. Ces systèmes possèdent une représentation des connaissances qui autorise leur réutilisation. Cependant les connaissances qui y sont explicitées et représentées le sont plus dans un souci d'opérationnalisation que pour la compréhension de l'utilisateur, ce dernier intervient donc peu dans la résolution.

Dans le but de bénéficier des avantages dus à l'interactivité des environnements de programmation graphique, notre système doit permettre à l'utilisateur de sélectionner et d'enchaîner des opérateurs de TI. Mais il doit également lui donner la possibilité de modéliser et d'expliquer le raisonnement qui l'a amené à cet enchaînement, dans le but de réutiliser la stratégie mise en œuvre, comme le font les systèmes automatiques.

Cependant, une application réelle nécessite l'enchaînement de plusieurs dizaines d'opérateurs. Pour modéliser le raisonnement associé à cet enchaînement, nous proposons une représentation par arbre de tâches appelé « plan de TI ». Cet arbre correspond à une décomposition hiérarchique d'un problème en sous-problèmes, chaque problème ou sous-problème étant associé à une tâche de TI. Comme on peut le voir sur la figure 1, un plan représente non seulement l'enchaînement d'opérateurs de TI correspondant aux feuilles de l'arbre, mais également le raisonnement nécessaire à la création de cet enchaînement, correspondant à des tâches de TI schématisées par les rectangles gris.

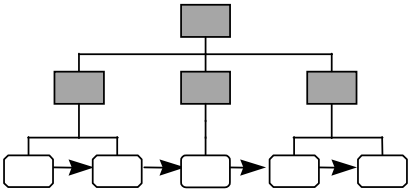

Fig. 1 : représentation d'un plan de TI

2.2. Le modèle TMO

Dans notre système, les plans de TI ainsi que les tâches de contrôle (pour la gestion des plans et du système) sont représentés uniformément à l'aide du modèle « tâche – méthode – outil » (TMO).

Une tâche représente un but ou un sous-but dans le système. Une méthode décrit un savoir-faire, elle spécifie comment une tâche peut être réalisée. Un outil est la réification d'un code informatique (opérateur de TI, fonction Lisp ou C) en termes conceptuels pour l'utilisateur avec un lien sur le code pour la mise en œuvre.

Il existe deux types de méthode : les méthodes « complexes » (fig. 2a) qui décomposent une tâche en sous-tâches à l'aide d'un arbre « PUIS » et les méthodes « terminales » (fig. 2b) qui réalisent une tâche en faisant appel à un code informatique par l'intermédiaire d'un outil. Enfin, un problème de TI pouvant être résolu par plusieurs stratégies, une tâche peut être associée à plusieurs méthodes (fig. 2c) sous forme d'un arbre « ou », le choix entre ces méthodes se faisant au moment de l'exécution.

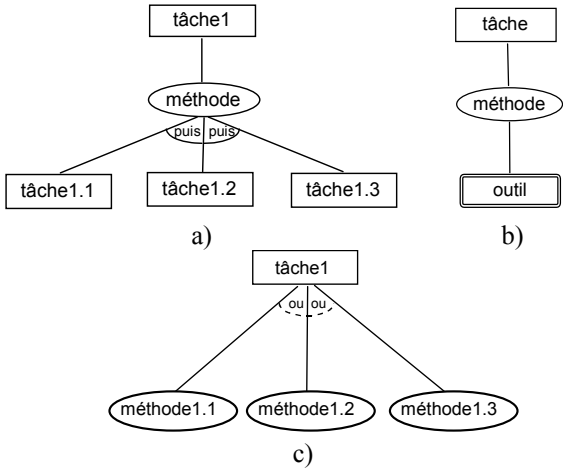

Fig. 2 : les différents enchaînements de tâche - méthode - outil

2.3. Fonctionnalités du système

Le système est muni d'une interface graphique dans laquelle plusieurs fonctionnalités ont été intégrées pour permettre la conception et l'exécution d'applications de TI de façon interactive. Elles comprennent en particulier la visualisation des applications sous forme d'arbres de tâches pour que l'utilisateur puisse étudier le raisonnement associé à un plan donné.

Pour créer un plan de TI, l'utilisateur définit ses tâches et ses outils en remplissant les champs correspondants dans des fenêtres adaptées, puis il les relie en définissant les méthodes et les flux de données entre tâches et sous-tâches ou entre tâches et outils. Il peut alors définir le mode d'obtention de la valeur de chaque paramètre des tâches ou des outils de trois manières : calculée dans une autre tâche ou dans un autre outil, fixée a priori ou demandée à l'utilisateur.

Pour exécuter une application, l'utilisateur sélectionne la tâche racine du plan correspondant ; celui-ci apparaît à l'écran sous la forme schématique d'un arbre de tâches. Le plan s'exécute alors interactivement en demandant à l'utilisateur de choisir une méthode quand il en existe plusieurs pour accomplir une même tâche et de lui fournir les valeurs des paramètres « utilisateur ». L'utilisateur peut ensuite accéder à toutes les informations sur les tâches et les outils exécutés et en particulier visualiser toutes les images intermédiaires pour étudier les points critiques.

La troisième fonctionnalité intégrée au système concerne la conception d'une application de TI par RàPC. Le module

la mettant en œuvre est décrit en détail dans les deux paragraphes suivants.

3. Représentation des cas et similarité

Un cas est globalement composé de deux parties : la description du problème et la description de la solution. La représentation d'une solution étant dépendante du modèle conceptuel choisi pour les modules du système précédemment décrits, nous commençons par présenter celle-ci.

Dans notre système, une solution est modélisée sous forme d'un arbre TMO, qui est repéré par sa tâche racine. Dans les systèmes de planification [16] [19] ou de conception [18], une solution est souvent construite en combinant des morceaux de plans issus de plusieurs cas. Pour autoriser ce type de conception, nous avons associé plusieurs cas à un même plan : le premier cas est associé à la tâche racine du plan, les autres sont associés à des sous-tâches de ce même plan qui sont représentatives d'une technique particulière. Dans l'exemple de la figure 3, des cas sont associés aux tâches T1, T2 et T4 qui définissent une certaine stratégie ; par contre, aucun cas n'est associé aux tâches T3, T5, T6, T7 et T8.

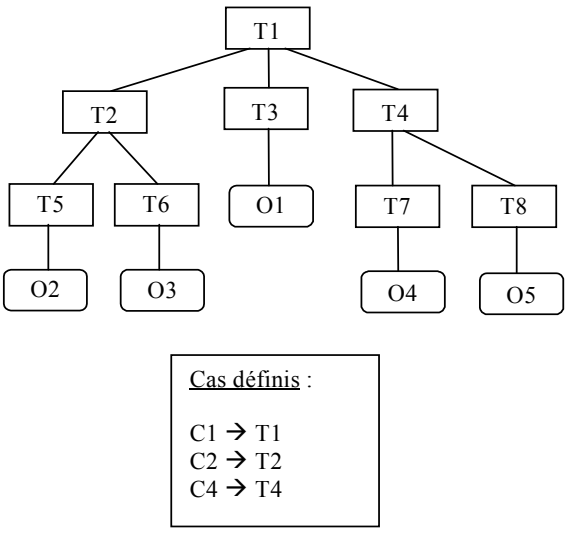

Fig. 3 : exemple d'association d'un ensemble de cas à un plan TMO

La description du problème s'effectue à l'aide d'un ensemble de critères discriminants, qui ont été déterminés à l'aide d'une étude approfondie du TI. Les résultats de cette étude sont présentés dans le section 3.1. Puis dans la section 3.2, nous décrivons les fonctions de similarité utilisées pour comparer deux cas à l'aide des critères retenus. Un exemple concret de cas est donné en section 4.

3.1. Critères de sélection d'un cas

La détermination des critères de similarité permettant de caractériser un problème de TI s'appuie d'une part sur une étude des systèmes de TI présentée en détail dans [10] et d'autre part sur une étude d'ouvrages et de thèses dédiés aux techniques de TI [6] [7].

Le premier problème concerne le choix d'un vocabulaire commun aux différents traiteurs d'images. A l'exception des actions de bas niveau (correspondant aux opérateurs de notre bibliothèque), il n'existe pas réellement de consensus sur les termes utilisés en TI. Ceci est dû, en particulier, à la difficulté de s'abstraire du domaine de provenance de l'image (beaucoup de traiteurs d'images travaillent sur un seul domaine d'application et utilisent donc les termes de ce domaine).

Nous proposons des critères issus d'une classification des termes fréquemment retenus pour décrire les actions et les données du TI. Nous avons distingué deux grandes catégories de critères : ceux liés à la description de la tâche et ceux liés à la description de l'image et de son contexte.

Critères liés à la description de la tâche

Ces critères sont des informations se rapportant à l'opération réalisée par la tâche et à sa position dans le plan par rapport aux autres tâches. Ils comprennent le type ou la phase de traitement, la définition même du problème et le niveau d'abstraction correspondant.

Le **type ou la phase de traitement** correspond globalement au type de problème résolu par la tâche. Suivant le niveau d'abstraction de la tâche concernée, on prendra en compte :

- soit le type de traitement : la tâche racine d'un plan complet définit un traitement de haut niveau qui appartient à un type de traitement (*restauration*, *détection*, *segmentation*, …),
- soit la phase de traitement : chaque sous-tâche d'un plan définit une partie du traitement qui correspond à une phase particulière de ce traitement (*prétraitement*, *détermination de germes*, *détermination des régions*, …).

Les différentes phases de traitement représentent un découpage vertical du plan (fig. 4) pour un type de problème, certaines phases pouvant être optionnelles (par exemple, la phase de prétraitement dans un problème de segmentation).

La **définition du problème** est composée d'un ensemble de mots clefs sélectionnés parmi trois listes prédéfinies : une liste de **verbes** décrivant les opérations effectuées par la tâche (*détecter*, *classifier*, *binariser*, *lisser*, …), une liste de **noms** correspondant, soit aux objets sur lesquels l'action est effectuée (*contours*, *régions*, *fond,* …), soit à la technique appliquée (*variance*, *croissance*, …) et une liste d'**adjectifs** qualifiant, soit les objets sur lesquels l'action est réalisée (*petit, local*, …), soit l'action elle-même (*partiel*, *fort*, …).

Comme on peut le voir sur les exemples proposés, le vocabulaire appartenant à ces trois listes de mots clefs est indépendant de tout domaine de provenance de l'image.

Enfin les **niveaux d'abstraction**, dont l'ensemble correspond à un découpage horizontal du plan (fig. 5), sont basés sur ceux définis dans le planificateur automatique BORG [5].

- le *niveau intentionnel* : les tâches de ce niveau répondent à la question « quoi faire ? » et portent sur les objectifs du TI.

- le *niveau fonctionnel* : les tâches de ce niveau répondent à la question « comment faire ? » et font référence à une technique du TI dans laquelle on fait abstraction des contraintes techniques liées à la réalisation.
- le *niveau opérationnel* : les tâches de ce niveau répondent à la question « avec quoi ? » et représentent un savoir-faire technique sur le TI qui peut s'implanter sous forme d'algorithme.

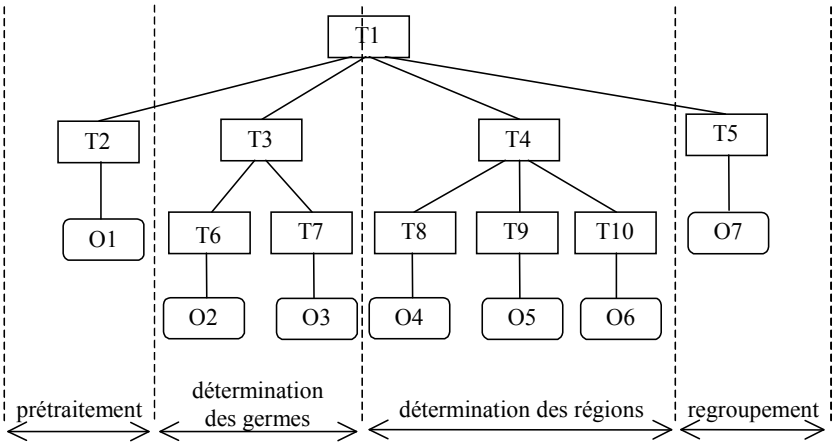

Fig. 4 : exemple de découpage vertical d'un plan pour un problème de segmentation

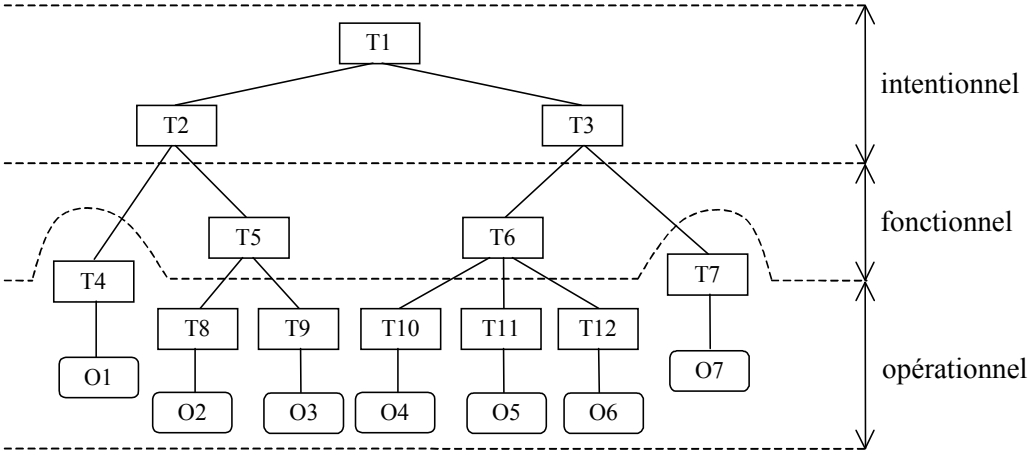

Fig. 5 : exemple de découpage horizontal d'un plan

Critères liés au contexte des images

Parmi les critères que nous avons définis, certains découlent des connaissances physiques (liées à la formation de l'image) et décrivent la qualité de l'image. Il s'agit en particulier du **type** et de la **quantité de bruit** que l'on trouve dans l'image et de la **qualité du contraste**. Ce sont des critères particulièrement importants pour le choix d'un prétraitement.

D'autres critères correspondent plutôt à la description perceptuelle de l'image (description symbolique en termes d'indices visuels). Ils comprennent la **présence** ou l'**absence d'un fond** et l'**aspect des objets** présents ou recherchés (*niveau de gris homogène*, *couleur claire*, *textures*, *contours épais*, …).

Enfin, le dernier groupe de critères est issu des connaissances sémantiques (analyse de la scène et de ses éléments) abstraites du domaine de l'image et porte sur ce que l'on cherche à mettre en évidence. Il s'agit de la **forme** des objets recherchés (*convexe*, *concave*, *allongé*, *compact*, *carré*, *rond*, …), de la **taille relative** des objets par rapport à l'ensemble des objets présents dans l'image, de la **position** des objets recherchés (*droite*, *milieu*, *gauche*, *bas*, *centre*, *haut*), et des **relations** entre ces objets (proches, connexes, inclus, …).

3.2. Calcul de la similarité entre deux cas

On peut distinguer deux principes pour déterminer si deux cas sont proches : maximiser la similarité [2] et minimiser l'effort d'adaptation [13] [18]. L'absence de méthode automatique d'évaluation des résultats en TI nous conduit à choisir la première approche. Nous décrivons d'abord les fonctions utilisées pour le calcul de la similarité. Puis nous détaillons les modes de comparaison définis pour les différents types de critères. Enfin, nous expliquons le mode de gestion de l'absence de valeur pour un critère, car comme c'est le cas dans ISAC [18], les critères énoncés précédemment ne sont pas tous pris en compte pour une application donnée.

Les fonctions de similarité

Parmi les critères que nous avons définis, le premier groupe, c'est-à-dire celui des critères liés à la définition de la tâche, caractérise l'action effectuée et est dépendant de notre modèle de représentation. Ces critères définissent un ensemble de tâches résolvant un « type de problème ». Ce sont des critères « obligatoires » (un cas cible possède obligatoirement une valeur pour chacun d'eux) qui vont nous permettre de réduire l'espace de recherche. Une première fonction de similarité Φ_t utilisant ces trois critères va donc servir à restreindre l'ensemble de cas sources candidats. Elle est définie par la formule (1) comme étant la moyenne pondérée des valeurs de similarité sur chacun des critères : S est le cas source, C est le cas cible, α_{Cr} est le coefficient d'importance du critère Cr et $\varphi_{Cr}(S,C)$ est la similarité entre S et C suivant le critère Cr. La valeur retournée par une fonction φ_{Cr} varie de 0, si les valeurs de Cr sont très différentes entre les deux cas, à 1 si elles sont identiques. Les coefficients α_{Cr} sont également compris entre 0 et 1, ce qui permet de normaliser la fonction Φ_t entre 0 et1.

$$\Phi_t(S,C) = \frac{\sum \alpha_{Cr} \times \varphi_{Cr}(S,C)}{\sum \alpha_{Cr}}$$

$$avec\ Cr \in \{\ critères\ liés\ à\ la\ tâche\} \quad (1)$$

Le deuxième groupe de critères, c'est-à-dire celui des critères liés au contexte des images, caractérise les objets manipulés et est dépendant de l'image traitée. Ces critères n'ont pas de sens pour toutes les images : par exemple la qualité du contraste n'a pas de sens lorsque l'on traite une carte de régions. Ce sont des critères « optionnels » (un cas cible n'aura pas forcément de valeur pour chacun de ces critères). La recherche de cas sources possédant des valeurs de critères proches de celles du cas cible s'effectue sur l'ensemble de cas retenus à l'aide de la fonction Φ_t. Une seconde fonction de similarité Φ_i permet donc de réduire cet ensemble de cas de manière à constituer une liste présentable à l'utilisateur. Cette fonction est définie par la formule (2) comme étant la moyenne pondérée des valeurs de similarité sur chacun des critères : les notations utilisées ont la même signification que pour la formule (1). Les coefficients α_{Cr}, les fonctions φ_{Cr} ainsi que la fonction Φ_i possèdent les mêmes propriétés que dans la fonction Φ_t, c'est-à-dire qu'ils sont normalisés entre 0 et 1.

$$\Phi_i(S,C) = \frac{\sum \alpha_{Cr} \times \varphi_{Cr}(S,C)}{\sum \alpha_{Cr}}$$

$$avec\ Cr \in \{\ critères\ liés\ au\ contexte\ des\ images\} \quad (2)$$

Les définitions des fonctions φ_{Cr} calculant la similarité suivant les différents critères sont données dans le paragraphe suivant. L'utilisation des fonctions de similarités Φ_t et Φ_i par l'algorithme de sélection/adaptation ainsi que le réglage des coefficients d'importance font l'objet de la section 4.

Les différents modes de comparaison sur un critère

Il est clair que la liste des critères relatifs au contexte des images proposée ne peut pas être exhaustive : ceux-ci ont émergé de notre étude de la littérature du TI et de la mise en œuvre de nos applications. Nous avons donc défini des types de critères associés chacun à un mode de comparaison de façon à pouvoir intégrer facilement un nouveau critère. La comparaison entre la valeur d'un cas cible et celle d'un cas source suivant un type de critère donné est définie par une fonction générique. Les types de critères que nous avons retenus sont définis ci-dessous ; pour chaque type, nous donnons également un exemple de critère correspondant avec son domaine de définition.

- critère numérique strict : la valeur est un nombre entier ou réel et la comparaison entre deux valeurs est 1 si elles sont égales, 0 sinon, (pas d'exemple actuellement)
- critère symbolique strict : la valeur est un symbole et la comparaison entre deux valeurs est 1 si elles sont égales, 0 sinon, (ex. : présence d'un fond → *oui*/*non*)
- critère numérique gradué : la valeur appartient à un intervalle d'entiers ou de réels et la comparaison entre deux valeurs est la valeur absolue de l'écart des deux valeurs rapportée à la plage de valeurs définie par l'intervalle, (ex. : taille relative des objets → *[1,5]*),
- critère symbolique gradué : la valeur appartient à un ensemble ordonné de symboles ; la comparaison entre deux valeurs est la valeur absolue de l'écart entre les deux valeurs (suivant l'ordre défini), rapportée à la largeur de la plage des valeurs de l'ensemble, (ex. quantité de bruit → *{très faible, faible, moyen, fort, très fort}*),
- critère multivalué : la valeur est un ensemble fini de symboles et/ou de nombres et la comparaison entre deux valeurs est le rapport entre le nombre d'éléments communs aux deux listes et le nombre d'éléments de la liste du cas cible, (ex. : liste de verbes pour la définition du problème → {*amincir, binariser, calculer, classifier, …*}).

L'absence d'une valeur pour un cas peut avoir plusieurs causes (non sens, inutilité, …) et peut être considérée de plusieurs façons (ne pas en tenir compte, la considérer comme une valeur particulière, …). Notre point de vue sur l'absence d'une valeur est différent suivant qu'il s'agit du cas source ou du cas cible :

- l'absence d'une valeur pour un cas cible signifie que la valeur est non pertinente pour le cas (soit elle n'a pas de sens, soit l'utilisateur l'a jugée inutile), cette

absence n'aura donc aucun impact sur le calcul de la similarité, ce qui correspondra à un calcul de similarité sur le critère et un coefficient d'importance nuls,

- l'absence d'une valeur pour un cas source (si cette valeur est présente pour le cas cible) signifie qu'une des conditions de similarité n'est pas respectée ; cette absence doit donc faire baisser le résultat du calcul de la similarité, ce qui correspondra à un calcul de similarité sur le critère nul et un coefficient d'importance non nul.

Ces deux conditions sont respectées par l'ensemble des fonctions génériques calculant la similarité pour tout type de critère.

4. Algorithme de sélection /adaptation

Notre module de RàPC doit permettre à l'utilisateur de construire un plan de TI à partir du problème courant défini à l'aide des critères présentés dans la section précédente. Contrairement à la plupart des systèmes de planification à partir de cas [16] [18] [19] qui procèdent par affinages successifs d'un plan abstrait, dans notre système, chaque étape de la mise au point de la solution fournit un plan hiérarchique complet qui peut être testé et évalué. Ce choix est dû, d'une part à notre volonté de fournir un système d'aide et non pas un résolveur automatique de problèmes, et d'autre part, au problème posé par l'évaluation des résultats d'une application de TI (il n'existe pas de fonction générale qui permette de comparer le résultat produit et le résultat souhaité). On obtient ainsi très rapidement une solution qui va servir de base au travail de l'expert en TI et l'on utilise la seule méthode d'évaluation généralement applicable à toute application de TI : l'évaluation visuelle des images résultats par l'expert.

Dans les processus de sélection/adaptation des systèmes de RàPC existants, on notera d'une part l'existence d'une étape préliminaire dans le processus de sélection visant à réduire l'espace de recherche [1] [3] [14], et d'autre part la présence d'un cycle sélection/adaptation applicable itérativement pour les problèmes de planification [16] [18].

Nous proposons donc un processus (fig. 6) qui met en œuvre un cycle sélection/adaptation applicable itérativement à différents niveaux du plan solution et dont chaque exécution comporte une phase de réduction de l'espace de recherche.

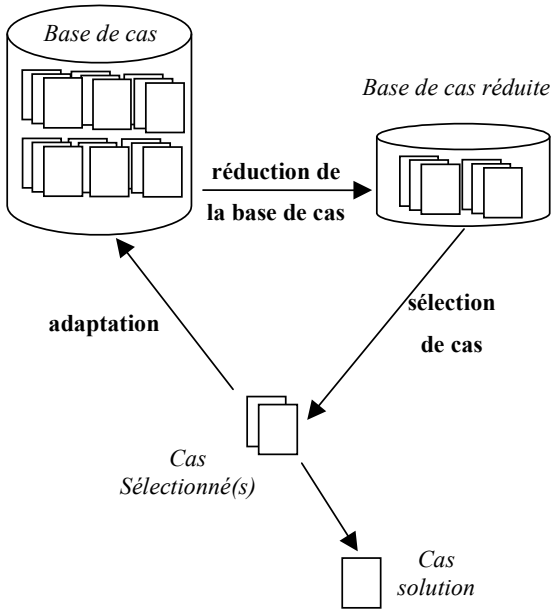

Fig. 6 : schématisation du processus de sélection/adaptation mis en œuvre

On peut réaliser la phase de réduction de l'espace de recherche, soit en utilisant des critères définissant des contraintes strictes, soit en considérant que deux cas ne peuvent être comparés que s'ils sont définis à l'aide du même ensemble de critères. La deuxième technique n'est pas adaptée à notre domaine. En effet, parmi les critères liés au contexte des images, certains peuvent être non informants pour le cas cible sans être disqualifiants pour le cas source. La phase de réduction de la base de cas est donc réalisée à l'aide de la fonction Φ_t utilisant les critères « obligatoires » liés à la description de la tâche et la phase de sélection de cas à l'aide de la fonction Φ_i utilisant les critères « optionnels » liés au contexte de l'image.

L'objectif de notre module de RàPC est de fournir une aide au traiteur d'images lors de la construction d'applications en l'aidant à sélectionner les solutions des problèmes déjà résolus proches de son problème courant. La mise en œuvre du processus de sélection/adaptation est donc réalisée en coopération avec l'utilisateur selon l'**algorithme** suivant :

1) Demander à l'utilisateur les valeurs des critères liés à la description du problème

2) Rechercher l'ensemble Σ des cas correspondant aux valeurs désirées en utilisant Φt

3) Demander à l'utilisateur les valeurs des critères liés au contexte des images

4) Tant que l'ensemble Σ n'est pas de taille convenable (par défaut entre 2 et 5 cas)

- Régler les poids des critères

- Réduire l'ensemble Σ en utilisant la fonction Φi

5) Demander à l'utilisateur de choisir un cas source parmi l'ensemble Σ

6) Présenter le plan solution à l'utilisateur et lui proposer de modifier les sous tâches qui ne lui conviennent pas, soit à l'aide du module de RàPC, soit à l'aide du module de création interactive.

Les étapes 1 et 3 correspondent à la saisie de la description du cas cible. L'étape 2 réalise la phase de réduction de l'espace de recherche. La sélection des cas sources candidats est mise en œuvre par l'étape 4 et est finalisée par le choix de l'utilisateur dans l'étape 5. Enfin l'étape 6 a pour but l'adaptation du cas source sélectionné au problème courant.

Les principes de sélection et d'adaptation utilisés dans cet algorithme, ainsi que la phase de mémorisation, sont détaillés et illustrés sur un exemple dans les sections 4.1, 4.2 et 4.3.

4.1. Sélection d'un cas source

Au cours de l'étape 2 de l'algorithme, la réduction de l'espace de recherche consiste à sélectionner les cas sources qui traitent du même type de problème que le cas cible *C*. Cela correspond à sélectionner les cas *S* tels que $\Phi_t(S,C) > \alpha_t$ où α_t est un seuil fixé à l'avance (la fonction Φ_t retournant une valeur entre 0 et 1, α_t sera fixé à 0,5 par défaut). Les valeurs des poids de chaque critère pour la fonction Φ_t sont également fixes : la même importance est accordée à chacun des trois critères (type ou phase de traitement : 1 / définition du problème : 1 / niveau d'abstraction : 1), pour la définition du problème nous accordons une importance un peu plus grande aux verbes qu'aux noms et adjectifs (verbes : 0,4 / noms : 0,3 / adjectifs : 0,3) car l'action réalisée par la tâche est d'abord décrite par le verbe. Cette étape fournit un premier ensemble de cas Σ.

Pour que l'utilisateur puisse faire son choix à l'étape 5, il faut que l'ensemble de cas résultant de l'étape 4 soit de taille raisonnable (par défaut entre 2 et 5 cas). Si l'on propose trop peu de cas, l'avis de l'utilisateur perd de son importance, et s'il y en a trop, il lui sera difficile de faire son choix. Le caractère itératif de l'étape 4 permet de déterminer un ensemble de cas présentable à l'utilisateur sous forme de liste : ce dernier peut alors examiner chaque cas en détail avant de faire son choix. La modification de l'ensemble Σ à chaque itération est effectuée par relaxation en modifiant les poids des critères et/ou le seuil de sélection. Pour cela, lorsque l'utilisateur saisit les valeurs des critères de son cas cible, il indique pour chaque critère s'il le considère comme important ou pas. Tous les coefficients des critères sont initialisés à 0,5. A chaque itération, le système sélectionne les cas *S* de l'ensemble Σ tels que $\Phi_i(S,C) > \alpha_i$ où α_i est le seuil de sélection. Si l'ensemble résultant n'est pas de taille convenable, on augmente les coefficients des critères les plus importants de 0,1 et on diminue ceux des critères les moins importants de 0,1 pour l'itération suivante. Lorsque l'on ne peut plus modifier les coefficients (coefficients des critères les moins importants à 0), si l'ensemble de cas sources n'est toujours pas de taille convenable, un deuxième mode de relaxation consistant à diminuer le seuil α_i est utilisé.

Prenons l'exemple d'un nouveau problème consistant à extraire les différents objets présents dans une image d'origine industrielle (image (2), fig. 7). L'utilisateur commence par définir son cas cible à l'aide d'une fenêtre de saisie : le **type de traitement** est *segmentation*, le **problème** est défini par les termes *extraire* et *objet*, la tâche est de **niveau** *intentionnel*, la **quantité de bruit** est *faible*, la **qualité du contraste** *moyenne*, un **fond** d'image est *présent*, on recherche des objets d'**aspect** *niveau de gris clair*, de **forme** *convexe*, de **taille** relativement *grande* et liés par une **relation** de *connexité*. Les critères présence d'un fond, aspect, forme et relation sont considérés importants par l'utilisateur.

Le système lance alors l'algorithme de sélection et retourne une liste de quatre cas parmi lesquels l'utilisateur choisit celui qui lui paraît le mieux adapté à son problème. Il peut alors visualiser le plan solution du cas source sélectionné pour étudier la stratégie proposée et l'exécuter pour le tester.

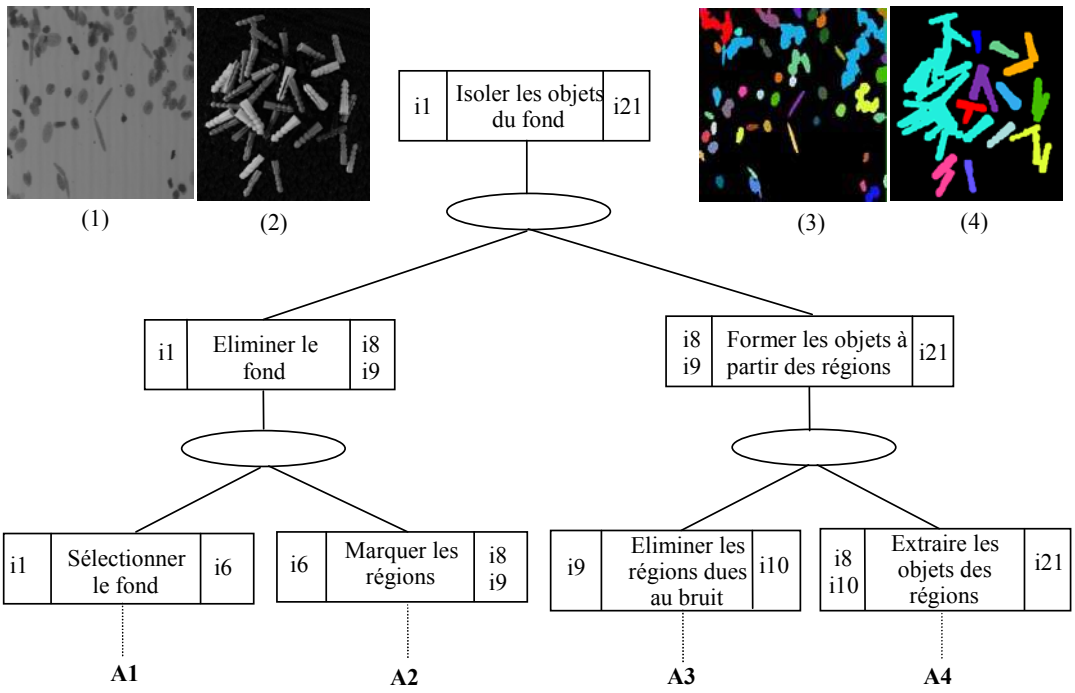

Fig. 7 : plan solution du cas sélectionné et images d'entrée et de sortie

Le plan sélectionné (fig. 7) a pour tâche principale « isoler les objets du fond » et a été conçu pour s'appliquer à des images de cytologie (image (1)) dans lesquelles on cherche à extraire un certain type de cellules (image (3)).

La réalisation de la phase de sélection de façon interactive permet de laisser suffisamment d'importance à l'aspect intuitif qui caractérise le mode de travail d'un expert en TI.

4.2. Adaptation interactive d'un plan

L'adaptation d'un cas à l'aide de parties d'autres cas est une technique particulièrement intéressante dans le domaine de la planification. Dans notre système, un cas peut être adapté à plusieurs niveaux et de plusieurs façons : localement ou globalement, à l'aide du module de RàPC ou du module de création interactive.

Le plan solution d'un cas peut demander uniquement des modifications locales. Il s'agit par exemple de régler les paramètres d'un opérateur ou de remplacer un opérateur par un autre mieux adapté. Ce type de modification peut être fait à l'aide des fonctionnalités de modification de plan proposées par le module de création interactive.

Mais ce plan peut également demander des modifications plus globales, c'est-à-dire nécessiter le remplacement d'un sous-plan par un autre. Pour cela, l'étape 5 de l'algorithme propose à l'utilisateur d'adapter la solution de son cas en remplaçant la tâche racine d'un sous-plan par une autre tâche. La tâche remplaçante peut être obtenue, soit en relançant l'algorithme pour trouver un cas correspondant, soit par construction via le module de création interactive.

Le plan sélectionné dans l'exemple de la section 4.1 a été adapté en plusieurs étapes. La première modification concerne la tâche « sélectionner le fond » : le plan initial a été conçu pour isoler des objets foncés sur un fond clair alors qu'ici les objets à isoler sont clairs sur un fond foncé. Une première adaptation consistant à inverser la sélection des objets (plan F1, fig. 8) est donc réalisée à l'aide du module de création interactive de TMO. Les résultats de l'exécution étant insatisfaisants (localisation imprécise des contours, mauvaise séparation des objets sur l'image (4) de la figure 7)), l'utilisateur va procéder à une deuxième étape d'adaptation en relançant l'algorithme de sélection pour trouver un autre sous-plan pour former les objets. Il définit donc un nouveau cas cible correspondant au sous-problème à résoudre et choisit le sous-plan remplaçant (plan F2, fig. 8) parmi ceux proposés par le système pour l'insérer dans le plan initial. Le plan résultant (fig. 8) peut ensuite être amélioré par des modifications locales (par ex. remplacement d'un opérateur par un autre).

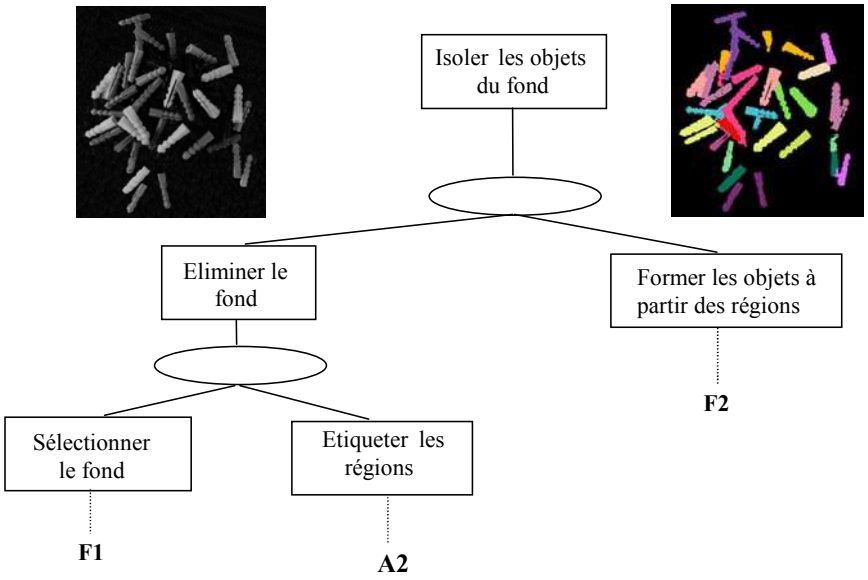

Fig. 8 : représentation partielle du plan résultat adapté

Cet exemple montre l'intérêt de l'aspect récursif de l'algorithme : un plan solution peut être adapté à n'importe quel niveau dans l'arbre de tâches et autant que nécessaire.

4.3. La phase de mémorisation

La mémorisation d'un nouveau cas n'a d'intérêt que s'il apporte de nouvelles connaissances à la base [18]. Cela implique qu'un cas doit respecter deux conditions pour être intégré : les connaissances associées doivent être correctes et elles doivent être suffisamment différentes de celles des cas déjà présents dans la base.

Vérifier le respect de la première condition consiste à tester la cohérence et l'efficacité du plan solution. Un plan est cohérent s'il s'exécute normalement et il est efficace s'il fournit des résultats satisfaisants. La cohérence peut être vérifiée par le bon déroulement d'une exécution mais l'efficacité ne peut être approuvée que par l'utilisateur, puisqu'il est seul juge de la pertinence des résultats. L'intégration d'un nouveau cas sera donc réalisée à la demande de l'utilisateur après que celui-ci ait validé sa solution par un ensemble de tests.

Un plan solution complet peut donner lieu à l'intégration de plusieurs cas dans la base : en effet si un plan complet représente la solution d'un problème de haut niveau, les différents sous-plans inclus représentent les solutions de problèmes de niveaux inférieurs. Lorsque l'intégration d'un cas est demandée, la première étape consiste donc à déterminer la liste des plans et sous-plans constituant les solutions des cas candidats à une intégration. Cette liste correspond à l'ensemble des plans ayant subi une adaptation, c'est-à-dire aux ascendants des sous-plans remplacés qui sont de taille suffisamment importante (au minimum sur trois niveaux de tâches). Si le plan remplaçant a été construit à l'aide du module de création interactive, on l'intègre également dans la liste.

Ainsi dans l'exemple présenté à la section 4.2, le système sélectionne les plans de tâches racines « sélectionner le fond », « former les objets à partir des régions », « isoler les objets du fond » et « éliminer le fond ».

Puis pour chacun des plans de cette liste, l'utilisateur doit préciser les valeurs de critères du cas correspondant qui ont été modifiées. Le système recherche alors le cas de la base ayant la plus grande similarité avec le nouveau cas et intègre ce dernier si cette similarité est inférieure à un seuil donné (i.e. si le cas est suffisamment différent des autres cas de la base). La similarité calculée ici correspond au minimum entre la similarité suivant les critères liés à la tâche et la similarité suivant les critères liés au contexte des images.

Dans notre exemple, les cas associés aux tâches « former les objets à partir des régions » et « isoler les objets du fond » seront intégrés à la base de cas, par contre les cas associés aux tâches « sélectionner le fond » et « éliminer le fond » ne sont pas suffisamment différents des cas de la base pour y être insérés.

5. Conclusion

Dans cet article, nous avons décrit un module de RàPC pour l'aide à la réutilisation des connaissances en TI. A l'aide de celui-ci, l'expert en TI peut rechercher un plan existant qui résout un problème proche du sien et l'adapter à la nouvelle situation. Il peut ainsi réutiliser ses propres connaissances ou celles qu'un autre expert en TI aura modélisées au préalable, réalisant alors un « partage des connaissances ».

L'algorithme récursif de RàPC mis en œuvre alterne les phases de remémoration et d'adaptation, permettant ainsi de concevoir un plan par assemblage de parties d'autres plans. Les critères de sélection d'un cas concernent la définition des traitements réalisés et la description des images sur lesquelles sont réalisés ces traitements. Les tests que nous avons effectués ont montré la pertinence du choix de ces critères.

Pour restreindre l'étendue du problème, les tests ont été réalisés sur des applications de segmentation d'images. Il sera intéressant de diversifier le contenu de nos bases (base de plans et base de cas) en intégrant des applications réalisant des traitements plus diversifiés (allant de la restauration à l'interprétation) et s'appliquant à des images de différents domaines. Ceci nous permettra également d'enrichir le vocabulaire utilisé pour la description des cas et ainsi de compléter notre ensemble de critères et de proposer à l'utilisateur des listes de termes plus exhaustives.

Il sera également intéressant de réduire la tâche de l'utilisateur lors de l'utilisation du module de RàPC, en particulier dans la phase d'adaptation. L'utilisation de règles fondées sur la comparaison des valeurs de certains critères permettra de guider l'utilisateur en lui indiquant les parties de plans nécessitant une adaptation.

6. Bibliographie

[1] A. Bonzano, P. Cunningham & B. Smyth, Using introspective learning to improve retrieval in CBR : A case study in air traffic control, *ICCBR'97*, Rhode Island, USA, pp. 291-302, July 1997.

[2] P. Caulier & B. Houriez, Apport de la modélisation des connaissances à partir de cas pour la capitalisation et la réutilisation des connaissances, *JAVA'95*, Grenoble, France, Avril 1995.

[3] B. Chiron & A. Mille, Aide à la conception d'environnements de supervision par réutilisation de l'expérience, *JICAA'97*, Roscoff, France, pp. 617-628, Mai 1997.

[4] Clément V. & Thonnat M., A Knowledge-Based Approach to Integration of Image, Processing Procedures*, CVGIP: Image Understanding*, vol. 57, n° 2, Academic Press, pp. 164-184, 1993.

[5] R. Clouard, *Planification incrémentale et opportuniste appliquée à la construction de graphes d'opérateurs de Traitement d'Images*, Thèse de doctorat, Caen, Février 1994.

[6] J.P. Cocquerez & S. Philipp, *Analyse d'image : filtrage et segmentation*, Masson, Paris, 1995.

[7] A. Elmoataz, *Mécanismes opératoires d'un segmenteur d'images non dédié : définition d'une base d'opérateurs et implémentation*, Thèse de doctorat, Caen, Juillet 1990.

[8] V. Ficet-Cauchard, M. Revenu & C. Porquet, Aide à la conception d'applications de Traitement d'Images : une approche basée sur le Raisonnement à Partir de Cas, *IC'98*, Nancy, pp. 1-10, Mai 1998.

[9] V. Ficet-Cauchard, C. Porquet & M. Revenu, An Interactive Case-Based Reasoning System for the Development of Image Processing Applications, *EWCBR'98*, Dublin, Irlande, pp. 437-447, September 1998

[10] V. Ficet-Cauchard, *Réalisation d'un système d'aide à la conception d'applications de Traitement d'Images : une approche basée sur le Raisonnement à Partir de Cas*, Thèse de doctorat, Caen, Janvier 1999.

[11] M. Grimnes & A. Aamodt, A two layer case-based reasoning architecture for medical image understanding, *EWCBR'96*, Lausanne, Suisse, November 1996.

[12] *IRIS Explorer User's Guide*, Silicon Graphics, Inc., Mountain View, California, n°007-1371-020, 1993.

[13] H. Muñoz-Avila & J. Hüllen, Feature Weighting by Explaining Cased-Based Problem Solving Episodes, *EWCBR'96*, Lausanne, Suisse, November 1996.

[14] B.D. Netten & R.A. Vingerhoeds, Structural Adaptation by Case Combination in EADOCS, *5th German Workshop on Case-Based Reasoning*, G*WCBR'96*, Bad Honnef, Germany, March 1997.

[15] P. Perner, Different Learning Strategies in a Case-Based Reasoning System for Image Interpretation, *EWCBR'98*, Dublin, Irlande, pp. 251-261, September 1998.

[16] B. Prasad, Planning With Case-Based Structures, *AAAI Fall Symposium*, MIT Campus, Cambridge, Massachusetts, November 1995.

[17] Rasure J. & Kubica S., The Khoros Application Development Environment, Experimental Environments for Computer Vision and Image Processing, editor H.I. Christensen and J.L. Crowley, *Word Scientific*, Singapore, n° 1-32, 1994.

[18] B. Smyth, *Case-Based Design*, Doctoral Thesis of the Trinity College, Dublin, Ireland, April 1996.

[19] M. Veloso, H. Munoz-Avila & R. Bergmann, cased-based planning : selected methods and systems, *AI Communications*, vol. 9, n° 3, September 1996.