CBR for the reuse of Image Processing knowledge : a recursive retrieval/adaptation strategy

Valérie Ficet-Cauchard, Christine Porquet, Marinette Revenu

CBR for the reuse of Image Processing knowledge : a recursive retrieval/adaptation strategy

Valérie FICET-CAUCHARD, Christine PORQUET & Marinette REVENU

GREYC-ISMRA - 6 Bd du Maréchal Juin - F14050 CAEN cedex FRANCE
tél: +33 (0)2-31-45-27-21 fax: +33 (0)2-31-45-26-98 e-mail: Valerie.Ficet@greyc.ismra.fr

Abstract. The development of an Image Processing (IP) application is a complex activity, which can be greatly alleviated by user-friendly graphical programming environments. Our major objective is to help IP experts reuse parts of their applications. A first work towards knowledge reuse has been to propose a suitable representation of the strategies of IP experts by means of IP plans (trees of tasks, methods and tools). This paper describes the CBR module of our interactive system for the development of IP plans. After a brief presentation of the overall architecture of the system and its other modules, we explain the distinction between an IP case and an IP plan, and give the selection criteria and functions that are used for similarity calculation. The core of the CBR module is a search/adaptation algorithm, whose main steps are detailed: retrieval of suitable cases, recursive adaptation of the selected one and memorization of new cases. The system's implementation is presently completed; its functioning is described in a session showing the kind of assistance provided by the CBR module during the development of a new IP application.

1. Introduction

We are doing research work in the design of an interactive system that can provide assistance during the working out of Image Processing (IP) applications; the system's architecture has been detailed in [5]. Our system is composed of several modules dealing with the tuning out of IP applications through interactive acquisition and representation of IP knowledge coming from IP experts, the execution of such IP applications and the reuse of applications following a Case-Based Reasoning approach (CBR). This paper is dedicated to a detailed description of the CBR module: in particular, our description of IP cases, similarity calculation between two cases and recursive search/adaptation algorithm are presented and discussed.
In section 2, the framework of our research is briefly presented along two axes: our objectives with regards to IP and our modeling of IP application. Sections 3 and 4 are entirely dealing with the CBR module: first, our definition of an IP case and the functions used for similarity calculation are given (section 3). Then the search/adaptation algorithm is described and explanations are given about the process

of case selection, recursive adaptation of the selected solution and memorization of new cases (section 4). Finally, a complete session showing how to use the CBR module for developing an IP application is described in section 5.

2. Research framework: the TMT model

Our primary objective is to represent and structure the knowledge of different IP experts so as to enable knowledge share and reuse. To achieve such a goal, we are advocating for an interactive system enabling knowledge acquisition from IP experts, as it comes to the fore through the development of IP applications. In this section, our approach for the building of applications is presented; we describe our model for the representation of applications and briefly give an idea of the functioning of two essential modules of the system: the interactive creation module and the execution module.

2.1. Representation of applications by hierarchical plans

Our approach to the development of IP applications is based on the smart supervision of libraries of operators. An operator is a program that performs one basic operation on one or several images. It takes as inputs the image(s) to be processed as well as parameters and produces as outputs one or several images as well as numerical and/or symbolical results. With such libraries, the building of an application then "simply" consists in linking operators and tuning their parameters. Users can thus stand back from computer codes and perform programming at the "knowledge" level.
However, a real-size application can lead to sequences of up to tens of operators. In order to represent the reasoning associated to such sequences, we suggest to use a representation based on trees of tasks, that we call "IP plans". Such trees correspond to hierarchical decompositions of problems into sub-problems, each problem or sub-problem being related to an IP task. As is shown in figure 1, a plan not only represents the linking of IP operators corresponding to the leaves of the tree, but also all the reasoning necessary for the creation of such a linking, which is represented by IP tasks schematized as gray boxes.

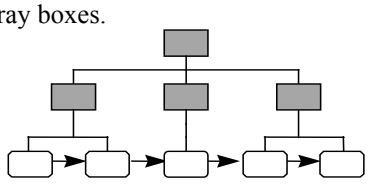

Fig. 1: representation of an IP plan

2.2. The TMT model

In our system, IP plans as well as control tasks (dealing with plan management and system control) are uniformly represented within the "task – method – tool" model. In this model, a task represents a goal or sub-goal; a method describes a know-how, it specifies how a task can be performed; a tool reifies a computer code (IP operator, Lisp or C function) in conceptual terms with a link to the code enabling to run it. There exist two types of methods: "terminal" methods (fig. 2a) that achieve a task by calling to a computer code through the medium of a tool and "complex" methods (fig. 2b) that decompose a task into sub-tasks by means of a "THEN" tree. Finally, as there may exist several strategies to solve an IP problem, a task can be associated to several methods (fig. 2c) by means of an "OR" tree, the choice of the method to be applied being made at the time of execution.

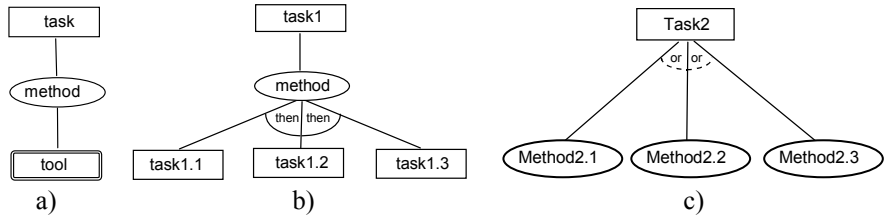

Fig. 2: various possible links between tasks, methods and tools

2.3. System's functionalities

Our system is provided with a graphical interface, in which several functionalities have been defined for the interactive construction and the interactive execution of IP applications. In particular, they include the visualization of applications as trees of tasks, so that users can study the reasoning associated to any given IP plan.

In order to create a new IP plan, the user has to define his/her tasks and tools by filling in fields in appropriate windows; he/she can link then by defining methods and data flows between tasks and sub-tasks or tasks and tools. There are three ways for specifying the way to get the values of parameters for tasks and tools: computed from another task or tool, fixed once and for all, or to be required from user.

When they want an application to be executed, users simply have to select the root task of the corresponding plan in a menu and the plan is immediately visualized on screen as a schematic tree of tasks. The plan can then interactively be executed: users are required to choose between methods when several methods exist to perform some given task, and also to provide values for "user" parameters. Once the execution is completed, they can have access to any information about tasks and tools that have actually been executed and moreover, visualize any intermediate image in order to assess critical points.

In addition to the creation and execution functionalities that have just been described, the third and most original functionality integrated into the system consists in a

second mode for creating applications through CBR. The corresponding CBR module is detailed in the next two sections.

3. Case representation and similarity

A case is broadly composed of two parts: description of the solution and description of the problem. In our system, a solution is represented as a TMT tree, which can be accessed through its root task. In Case-Based Planning [9] [12] or Case-Based Design [11], a solution is generally built by combining parts of several plans coming from several cases. In order to make this kind of design possible, we have decided to associate several cases to one single plan: the first case is associated to the root task of the plan, the others to some sub-tasks of the same plan, that are considered as representative of specific IP techniques. In the example of figure 3, cases are associated to tasks Ta1, Ta2 et Ta4 that correspond to some specific strategy in IP; by contrast, no case is associated to tasks Ta3, Ta5, Ta6, Ta7 et Ta8.

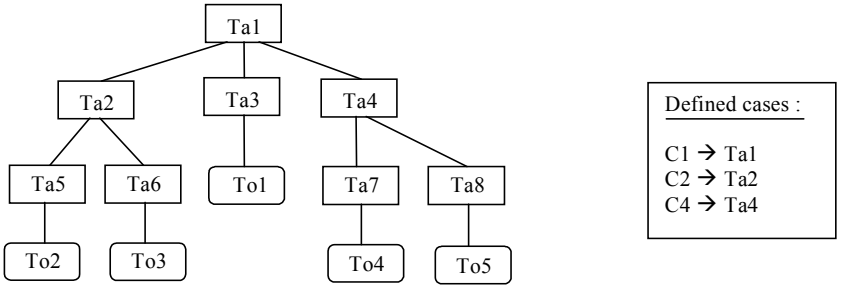

Fig.3: association of a set of cases with a TMT plan

The problem's description is made thanks to a set of discriminative criteria, which have been found out from a thorough study of the IP domain. Results of this study are presented in section 3.1; the similarity functions for comparing cases are described in section 3.2.

3.1. Criteria for case selection

The finding out of a relevant set of similarity criteria enabling to characterize an IP problem is based, on the one hand, on a study of IP systems detailed in [6] and, on the other hand on the study of books and Ph.D. dissertations dedicated to IP techniques [4] [10].
The major issue is here to choose an indexing vocabulary that can be shared and accepted by any IP programmer. Except for low-level actions (corresponding to operators from an IP library), there really exists no consensus on IP terms. In particular, this can be explained by difficulties to cut oneself off from the domain of

application (most IP programmers work on one type of application at a time and thus only use terms from their current domain of application).
The criteria we put forward come from a classification of the most often encountered terms used to describe IP actions and data. We have made a distinction between two broad categories of criteria: criteria related to the task definition and criteria related to the image description.

Criteria related to the task definition
This first category includes data related to the operation performed by a task and to its position in the plan in relation to other tasks. Such criteria include IP type or phase, problem definition and abstraction level.
IP type or **phase** broadly corresponds to the type of problem that is solved by a task. According to the task's abstraction level, one can take into account:

- either the IP type: the root task of a complete plan defines a high-level processing, which belongs to an IP type (*detection*, *segmentation, classification,…*),
- or the IP phase: each sub-task of a plan defines one part of the complete processing, which corresponds to one specific step (*pre-processing*, *seed determination*, *region determination, …*).

The various IP phases correspond to a vertical division of the plan (fig. 4); for some types of problems, some phases may be optional.

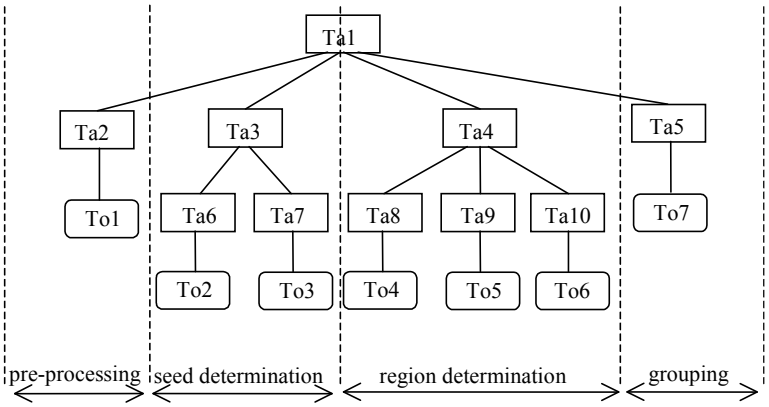

Fig.4: vertical division of a plan solving a segmentation problem

The **definition of the problem** is composed of a set of keywords selected among three pre-defined lists: 1. a list of **verbs** describing the operations performed by the task (*detect*, *classify*, *binarize, smooth, …*), 2. a list of **nouns** corresponding, either to objects on which the action is performed (*contours*, *regions, image background, …*), or to IP techniques (*region growing, region division, …*) and 3. a list of **adjectives** qualifying, either the objects on which the action is performed *(small, local*, …), or the action itself (*partial*, *strong, …*).
As can be noticed in previous examples, the vocabulary from these three lists of keywords is completely independent from the domain of application.

Finally the **abstraction levels** that correspond to a horizontal division of the plan (fig. 5) are based on the abstraction levels of the automatic planner BORG [3].

- Tasks belonging to the *intentional level* answer question such as "what to do ?" and deal with IP objectives.
- Tasks belonging to the *functional level* answer questions such as "how to do ?" and refer to some IP technique, leaving aside technical constraints related to their implementation.
- Tasks belonging to the *operational level* answer questions such as "by means of what ?" and represent IP technical know-how that can be implemented as algorithms.

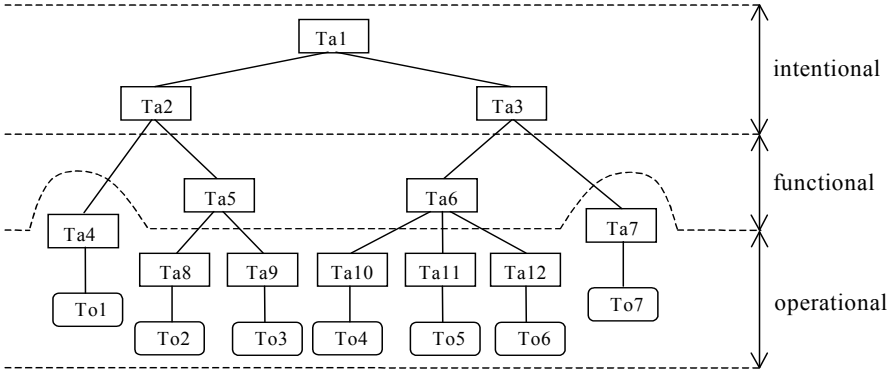

Fig.5: horizontal division of a plan

Criteria related to the image description

Among the criteria related to the context of images, some correspond to physical knowledge (related to image formation) and describe image quality (e.g. **type of noise**, **amount of noise** and **quality of contrast**). These criteria are of paramount importance for the choice of the pre-processing steps.

Other criteria rather correspond to perceptual knowledge (symbolic description in terms of visual primitives). They include the **presence** or **absence** of an image **background** and the **aspect of objects** (*homogeneous gray level, light color, texture, thick boundaries,* …).

The third group of criteria corresponds to semantic knowledge (scene analysis and components of the scene) and describes the appearance of what is to be detected, but in abstract terms, independent from the domain of application. These latter criteria include the **form** of objects (*convex*, *concave*, *elongated*, *compact*, *square, round*, …), the **relative size of objects**, their **position** (*left, middle, right, top, bottom, center)* and **inter-object relations** (*proximity, connectivity, inclusion,* …).

3.2. Similarity calculation between two cases

One can consider two principles for the determination of similar cases, either maximize similarity [2] or minimize adaptation effort [11]. Owing to the absence of any automatic method for evaluating IP results, we have chosen the former. First the functions used for similarity calculation between a source case and a target case are described. Then comparison modes for each type of criterion are detailed. Finally, the management of missing values for a criterion is explained; in fact, as it is the case in ISAC [1], all previously enumerated criteria need not be taken into account in any application.

Similarity functions

Our first group of criteria (i.e. criteria related to the task definition) is here to characterize the action performed by a task, and is thus closely dependent on the TMT model. Such criteria define a set of tasks that can solve one "type of problem". They are "compulsory" (each criterion of the target case must have a value) and are used to reduce the search space. A first similarity function Φ_t using the criteria related to the task definition will thus be applied to reduce the set of candidate target cases. This function is defined by formula (1) as the weighted average of the similarity results for each criterion: S is the source case, T is the target case , α_{Cr} is the importance coefficient associated to criterion Cr and $\varphi_{Cr}(S,T)$ is the similarity between S et T related to criterion Cr. The result value of any φ_{Cr} function is between 0 (if values of Cr between both cases are very different from each other) and 1 (when they are deemed identical). All α_{Cr} coefficients are also comprised between 0 and 1, in order to normalize the Φ_t function (return values between 0 and 1).

$$\Phi_t(S,T) = \frac{\left(\sum \alpha_{Cr} \times \varphi_{Cr}(S,T)\right)}{\sum \alpha_{Cr}} \qquad \forall Cr \in \{\text{criteria related to the task definition}\} \qquad (1)$$

The second group of criteria (i.e. criteria related to the context of images) characterizes the objects to be detected and depends on the current image. Such criteria are not meaningful for any application: for instance, contrast quality has no sense when processing a region map. This second group of criteria are "optional" ones (all criteria of the target case need not be filled in); they enable to select the nearest cases among the candidates obtained after applying function Φ_t. The second similarity function Φ_i is thus used to reduce the set of selected cases, in order to get a list of reasonable size. This function is defined by formula (2) as the weighted average of similarity results on each criterion; notations and properties are the same

$$\Phi_t(S,T) = \frac{\left(\sum \alpha_{Cr} \times \varphi_{Cr}(S,T)\right)}{\sum \alpha_{Cr}} \qquad \forall Cr \in \{\text{criteria related to the image description}\} \qquad (2)$$

as in formula (1).

The definitions of functions φ_{Cr} that are in charge of similarity calculation for each category of criterion are given in the next paragraph. The use of similarity functions

Φ_t et Φ_i in the selection/adaptation algorithm, as well as the adjustment of importance coefficients are explained in section 4.

Criterion comparison modes

It is clear that the list of criteria related to the context of images cannot be exhaustive: the criteria we put forward are coming from our study on IP literature and the development of our own applications. It should be completed in the course of further applications. Each criterion type is associated to a generic similarity function, in order to easily integrate new criteria. Here are the types of criteria that are presently available:

- strict numerical criterion: the value must be of integer or real type and comparison between two values returns 1 when values are strictly equal and 0 otherwise,
- strict symbolical criterion: the value is a symbol and comparison between two values returns 1 when values are strictly equal and 0 otherwise (e.g. presence of an image background),
- gradual numerical criterion: the value belongs to integer or real intervals and comparison between two values returns the difference between the two values divided by the interval length (e.g. relative size of objects),
- gradual symbolical criterion: the value belongs to an ordered set of symbols and comparison between two values returns the difference between the two values according to their order in the set, divided by the interval length (e.g. noise amount),
- multi-valued criteria: the value is defined as a non-ordered list of symbols and/or numbers and comparison between two values returns the ratio of the number of common elements in both lists to the length of the target case list (e.g. verbs used in the problem's definition).

A missing criterion value for a given case can be due to several causes (no meaning, usefulness, …) and can be taken into account in several ways (do not take into account, consider as a specific value, …). Our point of view on that issue differs whether one considers the source case or the target one:

- the absence of a value in a target case means that the value is considered as irrelevant for this case (either it is meaningless, or it has been judged as useless by user), that absence will have no consequence on similarity calculation ($\varphi_{Cr}(S,T)=0$ and $\alpha_{Cr}=0$),
- the absence of a value in a source case (while this value is present in the target one) means that one similarity condition is not respected; that absence should lower the result of similarity calculation ($\varphi_{Cr}(S,T)=0$ and $\alpha_{Cr}\neq 0$).

Both conditions are respected by the set of generic functions that compute similarity for each criterion type.

4. Recursive selection/adaptation algorithm

In the selection/adaptation process of most CBR systems, one can notice, on the one hand, the existence of a preliminary step in the selection process, aiming at reducing the search space [1] [8], and on the other hand, the fact that the selection/adaptation cycle must be applied iteratively, in particular in CBR planning [9] [11].
Our approach (fig. 6) is also based on a selection/adaptation cycle, iteratively applied at various levels of the plan, but in addition, at each cycle loop, a reduction step of the search space has been included.

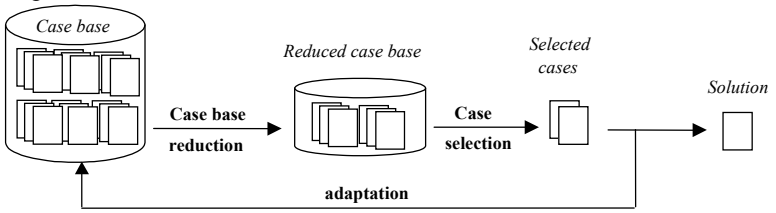

Fig. 6 : schema of our selection/adaptation process

The reduction of the search space can be achieved, either by using criteria corresponding to strict constraints, or by considering that two cases can only be compared when defined by the same set of criteria. The latter technique is not adapted to our domain. As a matter of fact, among the criteria related to the context of images, some of them bring nothing new about the target case, without disqualifying the source case. The reduction step can thus be achieved by means of function Φ_t using the "compulsory" criteria related to the task definition, while the selection step makes use of function Φ_i with the "optional" criteria related to the image description.
The objective of our CBR module is to provide some assistance to IP programmers when they are building applications, by helping them reuse solutions of previously-solved problems that are somewhat analogous to their current problem. The selection/adaptation process must thus take place in cooperation with the user, according to the following algorithm:

```
1. Ask user for values of criteria related to the task
   definition
2. Determine the set Σ of cases matching the desired
   criteria by means of Φt
3. Ask user for values of criteria related to the image
   description
4. While Σ is not of reasonable size do
          Modify the weight of criteria
          Reduce the set Σ by mean of Φi
5. Ask user to choose a case among the set Σ
6. Present the plan associated to the chosen case to user
   and propose him/her to modify the unsuitable sub-tasks,
   either by re-running the algorithm, or by building it
   from scratch, via the interactive creation module
```

Steps 1 and 3 correspond to the input of the description of the target case. Step 2 is the reduction step of the search space. The selection of candidate source cases is done in step 4; step 5 corresponds to the user's final choice. Finally, step 6 consists in adapting the plan associated to the selected source case to the current problem.
Principles for selection and adaptation of cases used in our algorithm are detailed in next two sections.

4.1. Selection of a source case

In the course of step 2 of the algorithm, the reduction of the search space consists in selecting source cases that solve the same type of problem as target case T. It corresponds to a selection of cases *S* such that Φ_t (*S*,T) > α_t where α_t is a threshold fixed beforehand (as function Φ_t returns a value between 0 and 1, α_t is fixed to a default-value of 0.5). The weights of each criterion in function Φ_t are also fixed: the same importance is granted to all criteria. This step provides a first set of cases Σ.
So that the user can choose a case at step 5, the set of cases resulting from step 4 must be of reasonable size. If the set is too small, the user's choice will loose importance, and if it is too large, the user's choice will be difficult. The iterative nature of step 4 enables to get a set whose size can be shown to the user as a list: he/she can then examine each case in detail, before making the final choice, which well accounts for the intuitive aspect that characterizes the way IP experts work. The modification of set Σ at each iteration is done by means of a relaxation process, by modifying the weights of criteria and/or the selection threshold. To implement this kind of relaxation, when the user enters the values of criteria for the target case, he/she must indicate whether the criterion is considered as important or not. All importance criteria are initialized with 0.5. At each iteration, the system keeps the cases *S* from set Σ such that Φ_i (*S*,T) > α_i where α_i is the selection threshold. If the size of the resulting set is too small or too large (by default between 2 and 5 cases), the coefficients of the most important criteria are raised by 0.1, whereas those of the least important ones are lowered by 0.1 for the next iteration. When it is no longer possible to modify coefficients (coefficients of the least important criteria have reached 0), if the set of source cases is still too small or too large, a second relaxation mode consisting in lowering threshold α_i is applied.

4.2. Interactive plan adaptation

Case adaptation by means of parts of other cases is particularly worthwhile in the domain of CBR planning. In our system, a case can be adapted at several levels and in several ways: locally or globally, either by means of the CBR module, or by means of the interactive creation module.
The plan solution to a case may only require minor local modifications. For instance, the parameters of an operator must be tuned, or an operator should be replaced by another one that better matches the current problem. This first type of modification

can be taken into account by using the modification menu of the interactive creation module.
But a plan may also require broader modifications, i.e. necessitate the replacement of a whole sub-plan by another one. To achieve such modifications, step 5 of the selection/adaptation algorithm offers a means to adapt the solution of the current case by replacing the root task of any sub-plan of the current plan by another task. The substitution task can be obtained, either by re-running the algorithm in order to retrieve a similar case, or by building it from scratch, via the interactive creation module. In the example of figure 7, a plan is adapted along three successive steps:

- replacement of sub-plan A by sub-plan A', which is obtained by re-running the selection algorithm,
- replacement of sub-plan B by sub-plan B', which is built via the interactive creation module.
- transformation of tool C into tool C', simply by changing the operator linked to tool C.

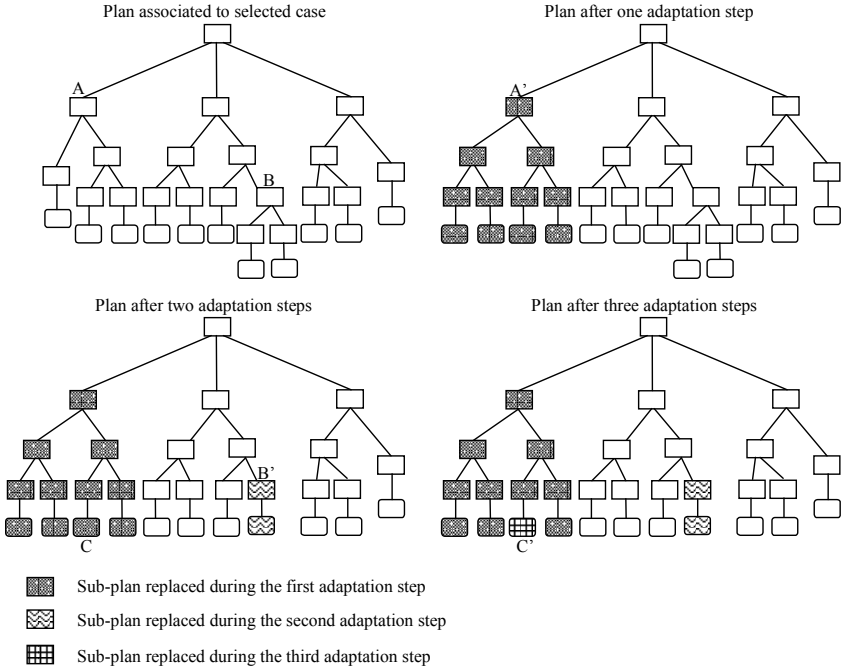

Fig. 7: adaptation of a solution plan along three steps

This example shows the interest in having a recursive algorithm: a plan can be adapted, whatever its level within the tree of tasks (A is a high-level task, B a low-level task, C an operator) and as long as necessary (A is replaced by A', then A' is adapted by replacing C by C'). Once a new plan is completed, one has to decide whether new cases associated to this plan should be added to the case library. This issue is discussed in the next section.

4.3. The memorization step

Memorizing a new case should only be considered if it brings new knowledge to the base. It implies that a case must respect two conditions in order to be integrated: the corresponding knowledge must be correct and it must bedifferent enough from the knowledge of the cases that are already in the base.
Checking the first condition consists in verifying the consistency and efficiency of the produced plan. A plan is consistent when its execution is normal and it is efficient if it produces satisfactory results. Consistency can be checked by the correct progress of the plan execution, while its efficiency must be assessed by the user, who is the only judge of its relevancy. The integration of new cases will thus be achieved, on user's requirement, once the solution has been validated through a set of tests.
Several cases associated to one complete plan can be integrated into the base: in fact, if the complete plan represents the solution of a high-level problem, its various sub-plans represent solutions of problems at lower levels. When the integration of a case is required, a first step consists in determining the list of plans and sub-plans that are candidates to integration. This list corresponds to the plans that have been adapted, i.e. the ancestors of replaced sub-plans that are large enough (at least three levels of tasks). If the substitution plan has been built via the interactive module, it will also be inserted into the list. Figure 8 takes up again the plan adapted in figure 7; the determination of the candidates to integration is achieved by examining the three replaced sub-plans:

- sub-plan of root A': D is inserted into the list; A' is not inserted because it stems from a case of the base,
- sub-plan of root B': plans of roots E and F are inserted into the list; B' has been manually built but it is not inserted because it has only two levels.
- sub-plan of root C': plans of roots A' et G are inserted into the list, whereas H and C' are not because they have less than three levels.

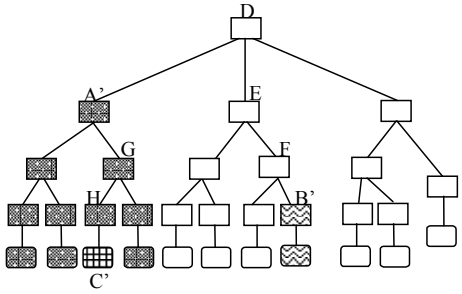

Fig. 8: determination of candidate cases to memorization

Then, for each plan in the list, the user has to provide values for the criteria of the corresponding case that have been modified. The system searches the case base for the most similar case to the new case and integrates the latter if similarity is lower than a given threshold (i.e. the new case is different enough from all base cases). The

similarity here considered corresponds to the minimum between similarity on task criteria related and similarity on image criteria.

5. The CBR module at work: an example

In this section, a session showing how the CBR module can be used during the creation of a new application is described. The new problem consists here in extracting objects in an image from industrial origin (image (2), fig. 9). The user begins by defining his/her target case through an input window: **IP type** is *segmentation*, **problem** is defined as *extract* and *object*, task's **level** is *intentional*, **amount of noise** is *low,* **quality of contrast** is *medium*, there is an image **background**, objects are characterized by their *light gray level* **aspect**, *convex* **form**, **size relatively** *large* and *connectivity* **relation**. Background, aspect, form and relation are considered as important by the user.
The selection algorithm is then run and a list of four cases is returned, among which the user chooses the case that seems to be the best match for his/her problem. The plan solution to the selected case can be visualized, so as to study its strategy and it can also be executed.
The root task of the selected plan (fig. 9) is "isolate objects from background"; this plan has been built for a cytology application (images (1) and (3)), for the extraction of some categories of cells.

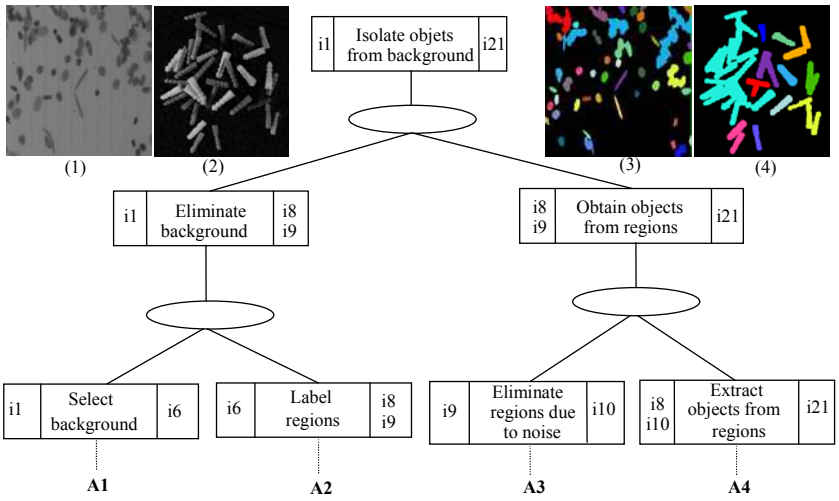

Fig. 9 : plan associated to the selected case with input and output images

The user can then start adapting the proposed plan to his/her new problem. The first modification deals with the "select background" task: in the initial plan, the problem was to isolate dark objects on a light background, whereas here, objects are light and background is dark. The first adaptation step simply consists in inverting the selection

of objects (sub-plan F1, fig. 10) and is thus achieved via the interactive module. As results after execution are still unsatisfactory (imprecise localization of contours, objects not properly separated, image (4)), the user considers a second adaptation step by re-running the selection algorithm in order to find another sub-plan for the task "obtain objects from regions". A new target case corresponding to this sub-problem is thus defined, the algorithm is re-run and the user finally chooses substitution sub-plan F2 (fig. 10). After replacement, the resulting plan (fig. 10) may further be improved by local modifications (e.g. replacement of an operator by another one).

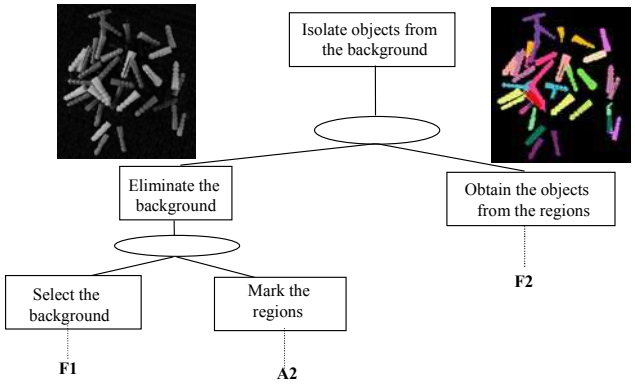

Fig. 10 : partial representation of plan after adaptation

Once all adaptations are completed, one has to define the new cases to be integrated into the base. The system produces the candidates to integration: they are the plans of roots "select background", "obtain objects from regions", "isolate objects from background" and "eliminate background". For these four tasks, the user is required to define the corresponding cases: two of these four cases are integrated into the base.
The assistance provided by the CBR module for the tuning of this plan shows the aptness of our selection criteria and the efficiency of the selection/adaptation algorithm: the interactive and recursive nature of this algorithm enables to rapidly get a satisfactory solution. However, the number of further local adaptations that must be made reveals the scarcity of our present case base, which must now be enlarged by systematically integrating all plans and cases corresponding to the applications developed within our research team.

6. Conclusions

In this paper, a CBR module providing assistance to knowledge reuse has been described. It enables an IP expert to retrieve an existing plan that solves a problem similar to his/her current problem and adapt it to the new situation. He/she can thus reuse his/her own knowledge or knowledge previously modeled by other IP experts. Our recursive selection/adaptation algorithm alternates retrieval and adaptation steps,

thus enabling to build a plan by combining parts of other plans. Criteria for selecting cases are based on a definition of IP tasks and a description of images.
Similar ideas can be found in HICAP [7], a general-purpose planning architecture that is applied to the planning of military evacuation operations. It is also a CBR system that can assist users during the construction of hierarchical plans of tasks. The system integrates a user-friendly task editor conducting an interactive conversation with the user. For tasks that can be decomposed in multiple ways (i.e. problem-specific tasks), a case is associated to each available decomposition method (whereas in our system, cases are associated to tasks and not to methods). So in HICAP, the user has to define a case in order to select each method used in the plan, which seems to be more constraining and time-consuming for the user.
The TMT system has presently been used to develop eight distinctive applications, in order to test the system along three main axes: validation of the model and architecture, experimentation of the interface by a novice and search for similarities between applications from different fields.
In order to restrain the scope of the problem, tests have presently been limited to segmentation applications. Further work will consist in diversifying the content of our libraries (plans and cases) by integrating applications dealing with more varied treatments (from image restoration to image interpretation) and applied to images from various domains. This should also enable to enrich the vocabulary used for the description of cases, and thus complete our set of criteria, so as to get a more exhaustive lists of terms.
In addition, one should consider means to alleviate the user's task in the course of the adaptation step. By using "simple" rules based on the comparison of some criterion values, the system could provide more assistance to user by indicating which parts of the plan need an adaptation.

References

[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, July 1997.

[2] P. Caulier & B. Houriez, A Case-Based Reasoning Assistance System in Telecommunications Networks Management, *XPS'95*, Kaiserslautern, Germany, 1995.

[3] R. Clouard, A. Elmoataz, C. Porquet, M. Revenu, Borg : A knowledge-based system for automatic generation of image processing programs*, IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 21, n. 2, pp. 128-144, February, 1999.

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

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

[6] 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, January 1999.

[7] H. Munoz-Avila, D. Aha, L. Breslow & D. Nau, HICAP: An Interactive Case-Based Planning Architecture and its Application to Noncombatant Evacuation Operations. IAAI-99.

[8] B.D. Netten & R.A. Vingerhoeds, Structural Adaptation by Case Combination in EADOCS, G*WCBR'96*, Bad Honnef, Germany, March 1997.

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

[10] Russ, John C. (1995) *The Image Processing Handbook*, second edition, CRC Press, 1995.

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

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