# Resolution of image processing problems by dynamic planning within the framework of the blackboard model

Régis Clouard, Christine Porquet, Abderrahim Elmoataz, Marinette Revenu

# Resolution of image processing problems by dynamic planning within the framework of the blackboard model

Régis Clouard, Christine Porquet, Abderrahim Elmoataz, Marinette Revenu

LAIAC (Laboratoire d'Algorithmique et d'Intelligence Artificielle de Caen)
ISMRA (Engineering School of Caen)
6, boulevard du Maréchal Juin
F-14050 CAEN CEDEX

## ABSTRACT

Recent works dealing with the development of automatic Image Processing (IP) systems non-dedicated to any specific application are all based on the search of a plan of treatments adapted to the nature of the problem and the images, among a base of predefined plans. In our approach on the contrary, we are interested in solving IP problems by building plans of treatment in a dynamic way and to use explicit knowledge for reasoning. Our system hinges upon hierarchical, incremental and opportunistic planning within the blackboard architecture. The system reasoning makes use of explicit knowledge about expertise in IP, in order to find out, set the value of parameters and form a sequence of classical IP operators.

## 1. INTRODUCTION

Non specialized Image Processing (IP) is no easy task because the processing techniques are very different from one problem to another. It is difficult to conceive general enough algorithms to deal with images that come from various origins, although a large variety of IP operators have been devised. In expert systems in IP, the reasoning is based on explicit knowledge, in order to find out, set the value of parameters and form a sequence of classical IP operators that meet the requirements of specific user tasks. It is necessary that such systems have a capability for self configuration to different IP requests and application contexts by using explicit knowledge representation about image processing techniques.

We are interested in solving IP problems by building plans of treatment in a dynamic way, according to the specific features of the problem and the characteristics of the images. This includes the selection of operators stored in a run-time library, the adaptation of their parameter values and their execution in a precise order, as well as the assessment of the relevance of the solution produced by the system.

This research involves two objectives: One the one hand, our purpose is to exhibit the various concepts related to Image Processing used in a knowledge-based system in IP, that should be automatic and non-dedicated to any specific application. This implies the finding out of knowledge, methods and a vocabulary that are specific to IP and not related to the application domain. This task is all the more difficult as treatments seem to depend more on data about the application than on the pixels of the image. On the other hand, we try to formalize all the domain knowledge, whether it is related to the expertise in IP or to the control of the resolution process, in terms of explicit, modular and independent knowledge sources. We are not only interested in solving IP problems, we also want to understand the reasoning of the system, in order to improve its results and behavior, and, in the long term, to allow explaining and teaching Image Processing to non-expert users.

In this paper, we first give an overview of image processing issues, then we present the choices and principles followed to build dynamically plans of treatments adapted to the problem data and the images. Section 4 describes our implementation based on the blackboard architecture, and more precisely, on the BB1 model, in which explicit knowledge is used to solve control problems and section 5 gives concrete examples of such knowledge.

## 2. IMAGE PROCESSING ISSUES

The field of IP covers all the operations dealing with the reduction of the amount of data contained in an image, as well as the enhancement of its quality. Marion [1] details among others:

- the enhancement of the subjective and objective quality of images,
- the restoration of deteriorated images,
- the detection of shape and texture primitives,
- the segmentation into homogeneous regions exhibiting specific features,
- the analysis and understanding by extracting synthetic attributes.

IP is achieved by means of operators. An operator is a program characterized by its inputs, parameters and outputs. The inputs consist in images which can be either purely digital or more symbolic. Parameters are necessary to adapt the behavior of the operator with regard to the specificities of input images and the objectives to be reached. The outputs can take the form of digital or symbolic images, as well as digital or symbolic attributes. There exists a great variety of operators in the literature; some of them can physically modify pixel values (smoothing, thresholding...), others ensure the construction of a new representation of the image data (regions, adjacency graph of regions, quad-tree...), others can also calculate the value of attributes (texture, number of regions...). All these operators are grouped together into a run-time library that should be large and varied enough to deal with all kinds of treatments, because it is well known that one cannot build an exhaustive library of IP operators [2].

Solving an IP problem consists in selecting operators, finding the optimal values for their parameters, and organizing operators into suitable sequences. The purpose of an automatic IP system is to compose complex IP processes from IP operators according to the characteristics of the image and the specifications of the problem that is set. The reasoning is done in four steps [3,4]:

1. *Building a plan* describing the set of actions to be coordinated so that the system can reach a desired goal.
2. *Instantiating the plan* with operators of the library by selecting operators and setting the value of their parameters.
3. *Executing the plan* with the latter operators following a definite order. To execute an operator, one takes as input the images resulting from the execution of the preceding operators in the plan, and produces output images that are used as inputs to the succeeding operators.
4. *Assessing the quality* of results in relation to the goal to be reached and the required quality criteria, leading eventually to a correction of the initial plan.

Reasoning mechanisms are based on knowledge that should be represented dynamically and explicitly. It is essential to notice here that, in order to remain as general as possible, the knowledge and vocabulary used must be "weeded out" of all technical references to the domain of application. That is the reason why we restrict to vocabulary in use in IP (regions, boundaries, lines, pixels...) and to mathematical vocabulary (geometry, algebra...) for describing relations and features of objects. On the contrary, notions related to the domain of application (cells, roads...) must not be mentioned in this way but described only by means of the authorized vocabulary (shape, texture...). This is a very strong and restrictive constraint but it is the only way to exhibit knowledge that is sufficiently general and reusable. Four types of knowledge are taken into account:

1. *Knowledge about the domain of IP and its context,* enabling to understand the problem data and to raise ambiguities when translating the problem into IP tasks. Tasks are our means of setting IP problems.
2. *Knowledge about the expertise in IP*, used to describe a problem as a sequence of primitive actions which constitute the core of our IP system but are not easy to extract because expertise being essentially intuitive, IP experts often proceed by a trial-and-error process. Formalizing IP knowledge is among the challenges of knowledge-based IP systems.
3. *Knowledge about the control of the resolution,* used to direct choices and to solve resolution conflicts. This knowledge takes generally a procedural form in problem solving mechanisms. On the contrary, we intend to express this knowledge declaratively and explicitly.
4. *Knowledge about the operators of the library* for selecting operators and setting the values of their parameters. This knowledge is totally independent from the implementation of the run-time library. On the other hand, it must deal with problems such as the following ones [4,5]:
   - The selection of operators is made difficult because a library has to group together many operators whose effects on images are not easy to estimate a priori.
   - Finding the optimal value of each parameter of an operator is delicate, all the more as the performances of an operator depend heavily on these values. Our proposition is to incorporate into our library only *atomic* operators, i.e. operators performing one and only one action. This principle has a two-fold advantage: reducing semantic

knowledge related to the selection of operators, as well as syntactic knowledge related to the choice of the values of parameters.

In our system, a problem is given by the user as a request on an image. This request is expressed as goals to be reached and constraints on solutions. Goals describe the nature of the problem, whereas constraints define the quality of the expected results. The user has the possibility to select IP tasks among a given set of tasks, and then, to precise the context of these tasks thanks to specific constraints also selected among a given set of constraints.

Because characteristics of images are essential to direct treatments, the input images must be described thanks to symbolic attributes. This is done by means of an attribute-value list describing physics about image formation (acquisition conditions, type of camera...), perceptual information (homogeneous background, texture of objects, size of objects...) and knowledge about the semantics of the scene (relations between objects...) [2,3].

Also there, the descriptive vocabulary used must be strictly limited to IP and mathematics. As it is impossible to describe the characteristics of an image exhaustively, attributes that are chosen must be related to the request. Moreover, the system is provided with an interface to interactively ask, during the resolution, for unknown or non-automatically calculable values of some attributes that are essential to direct processing. It is also important to devise a more sophisticated interface in order to get the specifications of the request through a dialogue with the user, to have him clarify his objectives, including a graphical interface that can be used to show directly on images details that cannot be easily specified through dialogue (minimal size of objects, localization of details...) [6,7].

## 3. THE PLANNING MODEL

We are now going to focus on the choices and principles followed to dynamically build a plan of treatments adapted to the problem data and the images.

### 3.1. Hierarchical planning

Building the optimal plan is done through hierarchical planning of treatments [8]. The use of multiple levels of abstraction results in an efficient searching strategy and a perspicuous representation of knowledge. The IP domain is studied under five abstraction levels, corresponding to the steps generally considered in IP to go from the problem specification to the selection of operators:

| | |
|---|---|
| Request: | Definition of the problem to be solved, given by the user in terms of goals to be reached and constraints on these goals. |
| Task: | All the primitive tasks to be solved can be found at this level. They are either deduced from the request or from the resolution strategy. |
| Functionality: | Functionalities describe general-purpose IP functions that must be implemented in order to solve some task. |
| Procedure: | Procedures correspond to classical IP operators, but defined independently from any specific implementation. |
| Operator: | Our run-time library of operators is our specific implementation of procedures. |

The first two levels specify the problem completely, through the determination of all the tasks to be solved. The next two levels are here to determine the various solutions advocated by the IP expert, in order to solve each of these tasks. The last level ensures the instantiation of the plan with operators from the library in use.

Building the plan is achieved by successive refinements of goals at one level into subgoals at the next lower level. Each level corresponds to a more or less coarse version of the solution and is decomposed into an ordered sequence of more technical subgoals.

The decomposition of a goal can either be a set of subgoals that must be executed sequentially, or a conjunction of subgoals, in which case the execution order is not specified. Input images to subgoals come either from the decomposed goal or from one of the preceding goals in a sequence. Output images are those explicitly mentioned in the decomposition of the goal (fig. 1).

At the last level, operators are executed in the order chosen by the plan. Because it is difficult to determine the optimal values of parameters a priori, operators are executed in a trial-and-error process by modifying values of parameters, so as to optimize some evaluation function [4].

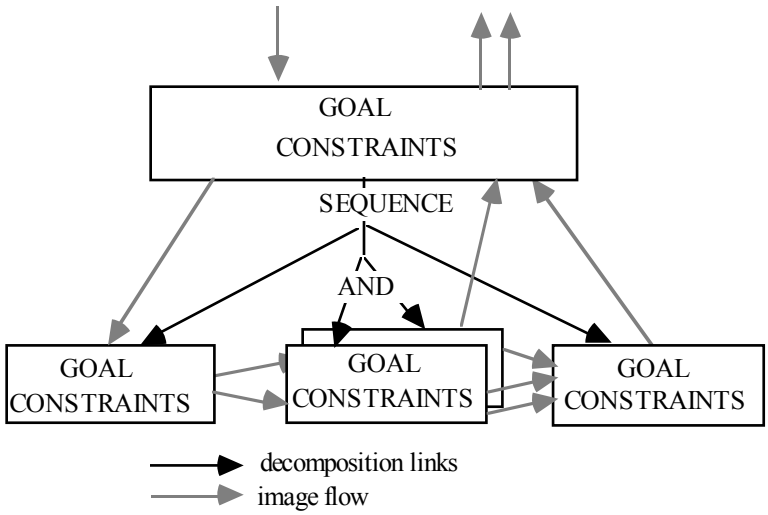


*Fig. 1 : Decomposition of a goal into subgoals*

More concretely, the request is split up into primitive tasks by reformulation, elimination of ambiguities and translation. The tasks are transformed into functionalities, which, in turn, are broken down into procedures. Procedures are implemented by means of operators, for which the values of each parameter must be calculated. Images resulting from the execution of operators are brought back upwards within the hierarchy and the output images of the request constitute the final result of the resolution. The user is in charge of the final visual evaluation and can reformulate the request if he is not satisfied.

### 3.2. Incremental formation of the solution

Our approach consists in building the plan dynamically, without its ever being totally or partially predefined. The plan is built step by step, so that it can match the current state of the solution. Each decomposition of a goal is made according to its specific constraints and characteristics of its input images. So it is necessary to have knowledge about the description of each input image to any goal, and not only knowledge about the description of the initial image. Each time an output image to a goal is created by the system, a description of its contents must be associated with it; here can one recognize the specific features of the *frame problem* [9]. The description of images is a difficult problem that we intend to solve thanks to a hierarchical description. Each goal at each level has to update the description of its own output images: at the highest level of the hierarchy, modifications are concerned with semantic attributes of images, whereas at the lowest level, they are rather concerned with physical attributes. The complete description is thus reconstituted when bringing back the results upwards within the hierarchy.

The same problem occurs when assessing the quality of the plan, and it is solved in the same way: the plan evaluation is done in a hierarchical and delocated manner. To each goal are associated rules for assessing its own results. At the highest level, these rules are concerned with semantic features and at the lowest level with syntactic features. This principle enables to organize and distribute rules into the hierarchy.

To summarize, each goal is responsible for the modifications that it can bring to the description of its own output images, and for the rules used to assess the relevance of its own results. Moreover, as a consequence of the incremental approach, it is practically impossible to forecast the plan that will come out as final solution.

### 3.3. Opportunistic problem-solving behavior

There are no predefined algorithms to solve a given IP problem, because of the size of data and the large number of alternatives to be taken into account. At each step of the resolution, several actions are feasible, so that the solution can make progress: should we decompose some goal into subgoals, assess the result of the execution of an operator, bring back up results of an evaluation or determine the evaluation rules of some other goal? Selecting the "best" action is very important for it influences directly the quality and rapidity of convergence of the solution [10]. This falls within the competence of the control of resolution mechanism.

Choice is done dynamically, according to the state of progress of the current solution. It hinges upon general heuristics and specific focuses of attention. This model enables to benefit from several resolution modes, either cooperative, or competitive. Several approaches are available: the goal-directed approach (choosing the actions that create important data), the action-directed approach (choosing the intrinsically important actions), the plan-directed approach (choosing an action in accordance with the current resolution strategy), either in a forward-chaining manner (deduction) or in a backward-chaining manner (induction).

To that purpose, one has to consider the data of the solution as hypotheses associated with a coefficient of plausibility and a coefficient of importance towards the current state of the solution. These coefficients are used by the control mechanism to calculate priorities in the development of the solution.

A system that combines the above-mentioned features is general, powerful and flexible [11] and can be directly implemented by means of a blackboard architecture.

## 4. IMPLEMENTATION

In this section, we are now going to describe the distinctive features of our system which is based on the blackboard architecture: the database, the knowledge sources and the way we solve control problems. The blackboard model is at once a conceptual, high-level organization of information and knowledge and a general prescription for the dynamic control and use of knowledge for incremental, opportunistic problem solving [11].

### 4.1. The database

Goals, input data, results, hypotheses are all stored into a global database, called the blackboard, which is organized vertically following the five levels previously detailed (fig. 2). A partial solution is represented as a five-level graph of goals. Each node of the graph is connected to nodes at the next lower level by *sequence*, *conjunction (AND)* or *disjunction (OR)* links. This graph is a means to represent the various alternatives of decomposition of a goal. But in the final solution, only one alternative must be kept, whether it is the best, or the first acceptable one. Links between goals can be seen as channels for the transmission of input and output images from one node to another.

Each level is described by a list of attributes. The attributes common to all five levels are the following ones:

| | |
|---|---|
| Goal: | a string defining the goal to be reached |
| Constraints: | a list of constraints describing quality requirements on results |
| Input images: | a list of input images together with their symbolic description |
| Output images: | a list of output images |
| Decomposition: | the set of nodes representing the decomposition of the present node |
| Result: | the path leading to output images of other nodes from the present node |
| Evaluation rules: | *if-then* rules to assess results |
| Judgment: | a value telling whether the results are acceptable or not |
| Importance: | a value representing the importance of the present decision towards the current solution. |

| Plausibility: | a value giving the confidence degree in the present decision related to the current solution |
|---|---|

Other specific attributes are defined for each level. For instance, nodes at the operator level have a *parameter* attribute, containing the domain of possible values for each parameter of the operator, as well as a *prototype* attribute giving the usage of the operator.

*Fig. 2 : The Image Processing blackboard*

**4.2. The knowledge sources**

The knowledge base is divided into independent and autonomous modules that totally ignore one another. A knowledge source (KS) contains expertise to solve some part of the global problem, the solution being built by the cooperation of several KSs. A KS constitutes a link between two nodes of the blackboard, one using it as its input and the other as its output.

KSs are defined in the traditional *condition-action* style. The condition part is responsible for determining when the KS can contribute to the problem resolution. The action part acts on the solution by creating or modifying data.

The knowledge base is composed of various kinds of KSs:

1. *Description KS* : Their function is to calculate the description of output images at each node of the graph, from effects estimated according to the current goal and the description of its input images, and also to define evaluation rules used to check the accuracy of the results with regard to the goals to be reached and the associated constraints. There are as many description KSs as there are types of goals.
2. *Decomposition KS* : They contain the knowledge to decompose some goal into subgoals at the next lower level. For that purpose, the description of input images and the constraints associated to the goal are used to build a set of subgoals and to specify their constraints and the relations existing between subgoals. For each potential goal are defined as many KSs as there are different potential decompositions.
3. *Execution KS* : There is only one KS of this kind. It executes operators according to an optimization mode, with the various combinations of possible values for each parameter. The result is determined by optimization of some evaluation function.
4. *Evaluation KS* : This is also a unique KS which only has to interpret evaluation rules associated with each node of the solution. It is triggered when some result has been calculated and the output images have been created. The result is judged acceptable when the output images are in accordance with the expectations.
5. *Propagation KS* : This KS, also unique, propagates the results of an execution through the hierarchy of nodes. It is triggered when some result has been computed on a node. The output images of this node are thus updated with the data and the description of images that were specified during the decomposition step.

Figure 3 is a diagram of a node of the graph surrounded by the KSs that are involved there.

*Fig. 3 : A node of the graph and its KSs*

**4.3. The control**

The control task is to select the next KS to be executed. The control structure consists of a control blackboard and a set of control knowledge sources. The blackboard model is inspired by BB1 [10], the control blackboard being decomposed into the six following levels:

| | |
|---|---|
| Problem: | definition of the problem to be solved |
| Strategy: | cooperative or competitive strategies used to solve the problem |
| Focus: | the implementation of strategies as current goals |
| Heuristic: | general-purpose heuristics defining the profile of the desired KSs |
| Agenda: | a list of KSs that are candidates for execution |
| Choice: | the KS selected and executed, together with the events it created |

The first four levels define, at each step of the resolution, the profile of the desired actions, with regard to the current state of the solution, whereas the two last levels define the profile of the actions that are actually feasible at this step of the resolution (the actions that can contribute to the progress of the solution).

The control problem is solved by KSs specific to control tasks, following the same principles as those dealing with the domain of application. There are no differences between control-related decisions and domain-related ones. The choice of the next KS to be executed is the result of a compromise between desired KSs, focuses and executable KSs. The profile of the next KS to be executed is determined by general-purpose heuristics and local focuses of attention, resulting from general or specific strategies. Executable KSs are put into the agenda. At each cycle, the scheduler has to decide whether to change the control or to continue the development of the current solution on the domain blackboard.

Here are some examples of general-purpose heuristics we use; we can give preference to:

- control actions versus domain actions,
- intrinsically important actions, such as the execution of the evaluation KS,
- actions that work on important data,
- actions that work on data with a low coefficient of plausibility to eliminate more rapidly parts of the graph that give unsuccessful results.

The general strategy we use is a breadth-first strategy. It is implemented as a sequence of focuses of attention on successive levels of the database. Specific strategies can also be considered, in order to give first priority to some parts of the graph that are judged important at this step of the resolution (e.g. a temporary depth-first strategy used during the decomposition of procedures into operators).

## 5. EXAMPLES OF KNOWLEDGE

In this section, we illustrate on a few examples how knowledge is used in the system. These examples are not complete and somewhat simplified, but they give a good idea of the functioning of the system.

### 5.1. Decomposition

Let us consider the following request given by the user of the system:

*"Isolate the objects from the background without separating clusters."*

This request means that the image is composed of objects on a background and that the characteristics of both objects and background are given in the description of the input image. It also implies that the system has to perform a segmentation into regions so as to associate each region to one object, and it must also eliminate the region corresponding to the background. The constraint specifies that objects that really overlap one another must not be separated; on the contrary, objects that partially touch one another must be isolated i.e. each of them must be associated to one region. The output image is thus a region map, each region of which corresponds either to an isolated object or to a cluster of objects.

To solve this request, two decomposition alternatives can be considered (fig. 4):

1. If the background is homogeneous and the objects can be easily distinguished, then the request can be decomposed into the following tasks: *"elimination of the background"* (consisting in building a map of all the regions that are not considered as the background), then *"construction of the objects from the regions"* (by using knowledge on the objects to merge and/or divide regions so as to build objects).

2. If the objects are homogeneous, then the decomposition can be *"segmentation into regions"* (according to criteria on the description of objects), followed by *"construction of the objects from the regions"* (which is the same task as in the first decomposition, but associated to a constraint consisting in eliminating from the region map the largest region corresponding to the background).

*Fig. 4: decomposition of the request into tasks*

Let us now see how to decompose the first task *"elimination of the background"*: two a priori equivalent alternatives can be considered:

1. The background can be eliminated thanks to an extraction of regions by binary thresholding if objects can easily be distinguished from the background (i.e. if the histogram is bimodal).

2. If the boundaries of objects are highly contrasted, one can also consider a boundary detection, followed by a construction of the regions from the boundaries.

In this particular context (homogeneous background, objects easily distinguished), the first alternative will be preferred by the system because it is less time-consuming and more robust.

### 5.2. Evaluation rules

Let us now examine the second task in the decomposition of the request: *"construction of the objects from the regions"* . We have to define rules for evaluating the quality of the segmentation into regions corresponding to objects or clusters of objects. Such rules perform a verification of some properties of objects that are specified in the description of the input image. For instance, if the size of the objects is informative, the evaluation rule can verify that the histogram of the size of the regions after the segmentation matches a reference histogram corresponding to objects of average size (fig. 5).

*Fig. 5 : an example of evaluation rule*

### 5.3. Description of the output images

We are now taking as an example the procedure *"smoothing by an averaging filter"*. Its purpose is to improve the quality of the image, by reducing the noise. The description of the output image is built from the description of the input image by modification of attributes of physical and perceptive nature: here, the noise attribute is changed from high to low, the image becomes blurred and the contrast on the boundaries decreases (fig. 6).

The description of the output image can be determined before the output image is produced, i.e. before the corresponding procedure has been decomposed into operators. It is calculated from the estimated effects of the smoothing procedure, by taking into account the description of the input image and the constraints associated to the goal.

*Fig. 6: determination of the description of the output image*

### 5.4. Execution of an operator

Our last example deals with the procedure *"binary thresholding based on edge-contrast"* (fig. 7). Here, the problem is to determine a threshold that allows the detection of the maximum of objects. An initial threshold ***t*** is obtained from a gradient image and a smoothed image of the initial scene. The optimal value of the threshold is then searched in the interval [ ***t - d, t + d*** ], where ***d*** represents a small number of gray levels. This optimal value gives boundaries of objects that closely match the edges of the gradient image. It should be noticed that we only keep amplitudes of the gradient that are maximal in

the direction of the gradient. The optimization function ***f1*** examines the consistency between the amplitude of the gradient image and the region boundaries obtained by the binarization operator; it is defined as follows:

*(sum-of-pixels (mask (amplitude-image by (boundary-image (binary-image)))))*

It can be interpreted as "calculate the sum of the values of pixels in the image resulting of a logical mask of the amplitude of the gradient image by the image of the region boundaries obtained by binarization".

The operator is executed with different values of the threshold in the interval [ ***t - d , t + d*** ], so as to optimize function ***f1***.

*Fig. 7 : execution of the operator "binary thresholding based on edge-contrast"*

## 6. CONCLUSIONS

In order to develop an automatic IP system non-dedicated to any specific application, one has to consider and solve many difficult issues. Planning treatments within the blackboard architecture enables to reason on imprecise and uncompleted knowledge; it also makes the updating of the knowledge base easier and achieves opportunistic and incremental planning by using a high number of KSs.

We are interested in implementing a keen and relevant reasoning rather than in finding out a solution efficiently and rapidly. If dynamic reasoning, flexibility and exhibiting explicit knowledge are favored, the elicitation of expertise is no easy

task, but we are helped by our incremental approach, which enables to progressively refine KSs through specialization or generalization schemes.

In the present state of the project, we are increasing the knowledge base and the library of operators by studying various IP problems in domains such as biomedical or geographical applications. One of the greatest difficulties encountered concerns the finding out of relevant evaluation rules.

Future improvements will be directed towards the integration of a better interface enabling a user, who is neither an IP specialist nor a programmer, to formulate its request and describe the images he wants to process. This interface must establish a dialogue with the user, in order to have him clarify the specifications of its request, and also include graphical tools to show details directly on images. Moreover, there is a lot of work to do on the explanation module, if we intend to use it for teaching Image Processing.

## 7. REFERENCES


1. A. Marion, "Introduction aux techniques de traitement d'images", *Eyrolles*, Paris, 1987.
2. A. Elmoataz-Billah, "Mécanismes opératoires d'un segmenteur d'images non dédié: définition d'une base d'opérateurs et implantation", *Thèse de doctorat*, Caen, 1990.
3. V. Clément, M. Thonnat "A Knowledge Approach to integration of Image Processing Procedures", *CVGIP: Image Understanding*, Vol 57 (2), pp-166-184, 1993.
4. T. Matsuyama, "Expert Systems for Image Processing: Knowledge-based Composition of Image Analysis Processes", *Computer Vision, Graphics and Image Processings* (48), Academic Press, pp 22-49, 1989.
5. D.G. Bailey, "Resarch on computer-assisted generation of image processing algorithms", *in proceedings IAPR Workshop on Computer Vision Special hardware and Industrial Applications*, Tokyo, pp 294-297, 1988.
6. R.C. Vogt, "Formalized approches to image algorithm developements using mathematical morphology", VISION'86, Detroit, pp 5_17-5_37, 1986.
7. C. Théot, E. Gallier, D. Mischler, "VSDE, un environnement de développement automatisé de système de vision", *Revue Technique Thomson-CSF* , Vol 24, pp 867-886, 1992.
8. E. D. Sacerdoti, "A structure for plans and behavior", *Elsevier*, New-York, 1977.
9. J. Mc Carthy, Patrick J. Hayes, "Some philosophical problems from the standpoint of artificial intelligence", *Machine Intelligence* (4), Edinburg University Press, Scotland, pp 463-502, 1969.
10. B. Hayes-Roth, "A blackboard architecture for control", *Artificial Intelligence*, vol(26), pp 251-321, 1985.
11. P. H. Nii, "Introduction", *Blackboard architecture and applications*, Academic Press, pp xix-xxix, 1989.