Pilotage de procédures de traitement d'images pour la description morphologique de galaxies

Monitoring of Image Processing Programs for Morphological Description of Galaxies

Véronique CLÉMENT
INRIA Sophia Antipolis
2004, Route des Lucioles
F-06565 Valbonne Cedex, France

Après une formation en mathématiques (Paris XI Orsay), en statistique (Paris VI Jussieu) et en informatique (Nice), Véronique Clément a présenté une thèse de doctorat en informatique à l'université de Nice en 1990, sur le pilotage de programmes de traitement d'images. Chargé de recherche à l'Institut National de Recherche en Informatique et en Automatique depuis 1990, ses axes de recherche concernent l'étude de la vision par ordinateur à l'aide des méthodes proposées par l'intelligence artificielle. Outre le pilotage de programmes, elle s'intéresse actuellement à l'interprétation de scènes par coopération de plusieurs sources de connaissance et à la fusion de données multi-capteurs dans le cadre, par exemple, de la télédétection.

Monique THONNAT
INRIA Sophia Antipolis
2004, Route des Lucioles
F-06565 Valbonne Cedex, France

Après son diplôme de l'École Nationale Supérieure de Physique de Marseille en 1980, Monique Thonnat a préparé une thèse au Laboratoire d'Astronomie Spatiale du Centre National de la Recherche Scientifique de Marseille ; cette thèse en optique et traitement du signal a été soutenue à l'université de St Jérôme de Marseille en 1982. En 1983, elle a rejoint l'Institut National de Recherche en Informatique et en Automatique comme chargé de recherche. Directeur de recherche depuis 1990, ses domaines de recherche comprennent la compréhension d'images, le traitement d'images et l'intelligence artificielle. Elle a participé au développement de plusieurs systèmes experts de vision en astronomie, biologie et robotique.

RÉSUMÉ

Cet article présente un apport des techniques de l'intelligence artificielle au domaine du traitement d'images. Le problème étudié est l'intégration sémantique de procédures de traitement d'images. Dans l'intégration sémantique, la fonction des programmes ainsi que la manière d'optimiser leur utilisation sont explicitées ; ceci permet une utilisation automatique, robuste et adaptable des algorithmes de traitement. Tout d'abord, nous décrivons le modèle utilisé, qui est celui du noyau de systèmes experts OCAPI. Le système expert PROGAL est ensuite présenté : tout d'abord, son application qui concerne l'automatisation du traitement d'images contenant une galaxie en vue de sa description morphologique, puis sa base de connaissance illustrée à l'aide d'exemples, et enfin une exécution commentée sur un cas particulier.

MOTS-CLÉS

Traitement d'images, intégration de programmes, système expert, description morphologique, galaxies.

ABSTRACT

This paper is concerned with the technics of artificial intelligence applied to the field of image processing. More precisely the problem studied here is semantical integration of image processing programs. In semantical integration of programs, the fonctionality of the programs, and the way to optimize their use are expressed ; this allows the use of the programs in an automatic, robust, and flexible way. First, the shell of expert systems used for the application, OCAPI, is described. Then the application on morphological description of galaxies is detailed : the design of the global system, the knowledge base of the expert system PROGAL, and an illustrated session of the expert system.

KEY WORDS

Image processing, programs integration, expert system, morphological description, galaxies.

1. Introduction

La plupart des applications de traitement d'images doivent maintenant être automatiques soit à cause de la grande quantité de données à traiter, soit à cause de la cadence de traitement à tenir, soit encore afin de soulager un opérateur de tâches fastidieuses. Pour le développement de ces applications, des algorithmes complexes sont étudiés et implémentés. Leur utilisation est toutefois généralement limitée à un type particulier d'images ; en effet, des hypothèses fortes doivent être faites par exemple sur les conditions d'acquisition des données (conditions d'éclairage, règles de positionnement des objets...). Ainsi une application pourra être performante dans le strict cadre des hypothèses envisagées. Mais bien souvent, un capteur se dégrade, un éclairage change ou un signal parasite intervient. Dans ce cas, il est difficile de détecter les changements de situations ou de s'y adapter puisque le système ne connaît pas les hypothèses sous lesquelles il travaille. Pour résoudre ce type de problèmes, les techniques d'intelligence artificielle semblent prometteuses. En effet, une méthodologie comme celle des systèmes experts permet de prendre en compte de nombreuses connaissances liées à l'application et en particulier la connaissance sur les méthodes mises en œuvre pour la traiter. Il devient alors possible de gérer les hypothèses faites par les algorithmes (et donc de ne déclencher ceux-ci qu'à bon escient) ou de contrôler la qualité des résultats (et d'être capable de chercher à les améliorer).

Un système de traitement d'images peut alors devenir intelligent et avoir un degré d'autonomie certain. Un tel système facilite le développement de nouvelles applications de traitement d'images puisque certaines tâches peuvent être automatisées (comme la sélection des meilleurs programmes pour atteindre un but ou l'initialisation de certains paramètres) ; cela rend l'application non seulement robuste au bruit mais également adaptable à des conditions variées. De plus, cela permet d'étendre l'utilisation des systèmes de traitement d'images qui peuvent être directement utilisés par des non spécialistes en traitement d'images (biologistes, astronomes...) ; non seulement l'intégration physique des algorithmes sur une architecture matérielle donnée et leur intégration syntaxique sous forme de langage de commandes sont réalisées, mais également leur intégration sémantique qui permet, par exemple, d'éviter des erreurs d'utilisation.

Récemment, grâce à l'utilisation de la méthodologie des systèmes experts, de tels systèmes intégrés intelligents ont commencé à faire leur apparition ; ils contiennent la connaissance nécessaire pour la sélection et l'utilisation des programmes vus comme des boîtes noires. Certains des algorithmes peuvent bien sûr faire de la reconnaissance d'objet, et ils peuvent être intelligents ; mais ils sont considérés par le système intégré comme une boîte noire. Dans de tels systèmes, le but des programmes et leurs conditions d'utilisation sont exprimés explicitement. Ils sont généralement développés en utilisant la méthodologie des systèmes experts. Certains sont eux-mêmes des systèmes experts comme Expert Assistant [Joh 87] développé à l'aide de KEE ; d'autres sont des noyaux de systèmes experts développés pour prendre en compte les spécificités du problème du pilotage d'algorithmes, tels que DIA-ES [TS 84, ST 85] ou le système de Toriu [TIY 87]. En fait, cette méthodologie permet la séparation entre des connaissances particulières sur un domaine d'application et des programmes d'une part, et le contrôle du raisonnement pour l'intégration d'autre part ; ainsi le développement et la réutilisation de tels systèmes sont facilités. Divers types d'applications ont été réalisés : Expert Assistant a été utilisé dans le domaine de l'astronomie pour la calibration CCD, le système de Toriu pour des applications de comptage ou DIA-ES pour la conception d'algorithmes dans un problème de reconnaissance de formes [SKT 88]. Les buts de leurs auteurs sont divers : aide interactive (Toriu) versus aide semi-automatique (DIA-ES), génération de programmes ou de scripts (Expert Assistant) versus exécution de commandes (DIA-ES).

Ces travaux, qui portent sur la modélisation des programmes et de leur utilisation, doivent être distingués de ceux portant sur la modélisation des données en analyse d'images ou en segmentation comme les travaux de Levine [NL 84] et Matsuyama [Mat 89, MH 90], et des travaux portant sur la modélisation des objets à reconnaître comme dans [Bro 81], [TG 88] et [IK 88].

Dans la suite de cet article, nous détaillons le noyau de systèmes experts OCAPI que nous avons développé. Nous présentons ensuite le système expert PROGAL, réalisé à l'aide d'OCAPI ; ce système automatise le traitement d'images contenant une galaxie en vue de leur description morphologique. La base de connaissance est illustrée par de nombreux exemples. Une session du système expert est ensuite commentée.

2. Modèle du système OCAPI

Nous sommes intéressées par la construction de systèmes de traitement d'images capables de générer, exécuter et optimiser le traitement de données en vue de résoudre une certaine tâche. Notre principal objectif est de construire des systèmes ayant un degré d'autonomie maximal. Nous avons vu que de tels systèmes facilitent le développement de nouvelles applications de traitement d'images et qu'ils sont absolument nécessaires pour construire des systèmes complètement automatiques intégrés sur des systèmes autonomes fonctionnant dans des environnements complexes et variables (comme des robots mobiles ou des bras articulés). De plus, plutôt que de développer un système spécifique intelligent, nous sommes intéressées par la définition d'un outil indépendant du domaine d'application, et indépendant du système de traitement d'image ou de la bibliothèque de programmes.

Avec ces objectifs, nous avons développé l'outil OCAPI que nous décrivons ici brièvement (voir détails, justifications et discussions dans [Cle 90] et [CT 90]. Celui-ci est un noyau de systèmes experts spécialisé dans le pilotage de traitement d'images : c'est-à-dire qu'il offre des structures

de représentation des connaissances ainsi que des mécanismes de raisonnement particulièrement adaptés au traitement d'images. Dans cette partie, nous décrivons tout d'abord les connaissances prises en compte par le système en définissant la typologie des différentes sortes de connaissances puis les relations entre ces concepts. Nous présentons ensuite le raisonnement mis en œuvre.

2.1. LES CONCEPTS ET LEURS RELATIONS

Quant nous traitons des images, nous avons toujours en tête un objectif ou un but à atteindre. Cette notion de but est proche de celle de tâche à réaliser. Dans le formalisme d'OCAPI, un **but** représente une fonctionnalité de traitement d'images. Cette fonctionnalité ou cet objectif peut être atteint par un programme ou par un traitement complexe. par exemple, *lisser une image* et *compter les objets* sont des buts.

Les programmes implémentent des fonctionnalités de traitement d'images mais ils ne sont pas les fonctionnalités. Nous distinguons la notion abstraite de *but* de celle d'*opérateur*. Les **opérateurs** sont des actions qui peuvent être exécutées. Ils contiennent des connaissances spécifiques pour atteindre un certain but. Un opérateur peut être un programme particulier (nous parlons d'**opérateur élémentaire**) ou une séquence particulière de traitements (nous parlons d'**opérateur complexe**). Par exemple, l'opérateur élémentaire *median3* est un programme qui implémente un filtre médian avec une fenêtre 3×3, et l'opérateur complexe *compte-objets1* est une succession de plusieurs traitements (*lissage, segmentation, tri, énumération*).

Deux autres notions sont importantes : la notion de contexte et celle de requête. Le **contexte** est la description des données d'entrée, leurs conditions d'acquisition et même de l'information sémantique sur le contenu supposé de la scène. Par exemple, un contexte peut être : *c-robot : le domaine d'application est la robotique, l'instrument d'optique est une caméra CCD, le digitaliseur est standard, les scènes sont des scènes d'intérieur.*

Un problème en vision est exprimé par son but, mais aussi par les données à traiter et des contraintes éventuelles sur les résultats à obtenir. Nous regroupons ces informations et les désignons par le terme de **requête.** Une requête précise quel est le but à atteindre, la qualité souhaitée pour les résultats et le contexte particulier. Une requête est donc une demande pour la résolution d'un but sur un cas particulier. Par exemple, une requête pourrait être : *r-régions : segmentations en régions de l'image view2, la taille des régions doit être supérieure à 2 500 pixels, le contexte est c-robot.*

Les opérateurs (élémentaires et complexes) et les buts ont des **arguments d'entrée et de sortie.** Par exemple, le but *lissage* a deux arguments : un argument d'entrée (l'image d'entrée) et un argument de sortie (l'image lissée). L'opérateur *median1* qui implémente une façon de résoudre le but *lissage* a trois arguments : deux arguments d'entrée (l'image d'entrée et la taille de la fenêtre) et un argument de sortie (l'image lissée). Nous distinguons deux classes d'arguments : les **données** et les **paramètres.** Les données ont des valeurs fixes qui sont fournies (entrées) ou calculées (sorties). Dans l'exemple précédent, l'image d'entrée et l'image lissée sont les données. Les valeurs des paramètres sont ajustables ; les paramètres sont généralement associés à des opérateurs et sont toujours des arguments d'entrée. Les opérateurs ont au moins les mêmes arguments (données et paramètres) que le but auquel ils sont associés. De plus, les opérateurs ont souvent des paramètres ajustables privés comme par exemple la taille de la fenêtre pour l'opérateur *médian1*. Ces deux types d'arguments, données et paramètres, possèdent chacun une description détaillée que nous ne présentons pas ici (voir [CT 90]).

Au cours de l'exécution d'un traitement d'image, il faudra savoir résoudre plusieurs problèmes : comment choisir entre les méthodes (ou opérateurs), comment initialiser les valeurs des paramètres, comment évaluer les résultats, et éventuellement comment modifier le traitement (valeurs de paramètres ou choix d'opérateurs) pour améliorer les résultats. Dans OCAPI, la connaissance permettant de résoudre ces tâches de raisonnement est structurée en 4 parties :

- choix : comment sélectionner parmi tous les opérateurs disponibles celui qui est le plus pertinent en fonction des données à traiter et du contexte. Par exemple : *si l'image a un fort contraste, utiliser un opérateur de segmentation basé sur les contours* ;
- évaluation : comment qualifier les résultats fournis par l'exécution de l'opérateur sélectionné en prenant en compte les contraintes exprimées dans la requête. Par exemple : *si le nombre de régions est supérieur à 100, la segmentation est trop fine.*
- initialisation : comment initialiser les valeurs des paramètres. Par exemple : *la valeur par défaut du paramètre taille-fenêtre est 3.*
- ajustement : comment modifier les valeurs des paramètres après une évaluation négative des résultats. Par exemple : *si la segmentation est trop fine, augmenter la valeur du seuil th-sigma.*

Ces différents concepts sont bien sûr reliés les uns aux autres. Par exemple, un opérateur sait résoudre un certain but ; de même, la manière d'évaluer des résultats est dépendante du but suivi. Le formalisme retenu pour la représentation de la base de connaissance permet d'exprimer ces relations explicitement. D'une part, les buts, opérateurs, requêtes et contexte sont représentés par des objets structurés reliés les uns aux autres. D'autre part, certaines connaissances dynamiques comme la manière d'évaluer les résultats sont représentées sous forme de règles de production qui sont elles-mêmes reliées à certains objets. Les relations entre les objets et les règles de la base de connaissance sont montrées (*fig.* 1). Une requête (par exemple *Requete-00*) est relative à un but particulier (ici *But-1*) ; ce but peut être résolu par plusieurs opérateurs (*Opérateur-1*, *Opérateur-2* et *Opérateur-3* dans ce cas). Une décomposition d'un opérateur complexe (*Opérateur-2* ou *Opérateur-3*) est un arbre de requêtes à d'autres sous-buts. Chaque but et chaque opérateur a deux bases de règles privées : une base de règles de choix et une

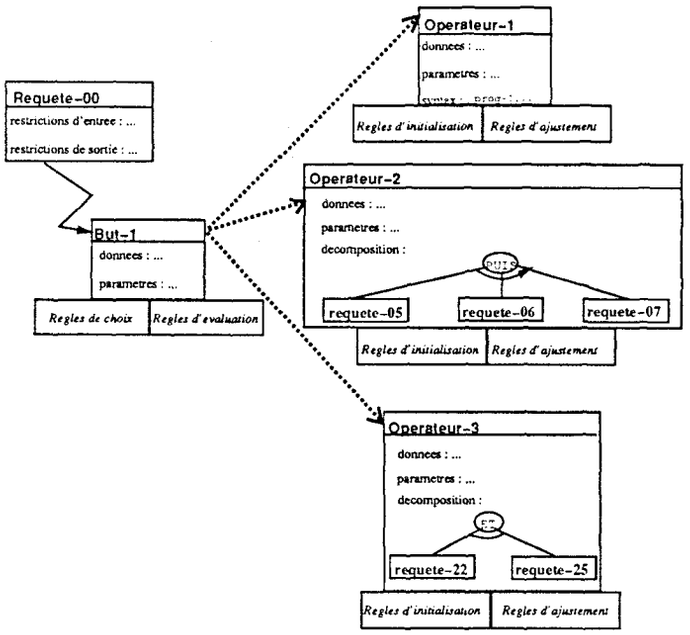


Fig. 1. — Relations entre les objets et les bases de règles dans OCAPI.

d'évaluation pour un but ; une base de règles d'initialisation et une d'ajustement pour un opérateur.

2.2. LE RAISONNEMENT

Le besoin de deux types de raisonnements a été mis en évidence (voir [CT 90]) : il s'agit de la planification et du contrôle d'exécution (*fig.* 2). Le rôle de la planification est d'élaborer les séquences d'actions à exécuter en fonction de la requête de l'utilisateur ; le contrôle d'exécution permet d'une part de fournir les résultats et d'autre part d'assurer qu'ils sont conformes aux spécifications.

Dans OCAPI, une phase de planification raisonne pour trouver un plan approprié. Cette étape détermine la liste des opérateurs à exécuter en fonction de la requête de l'utilisateur : ceci est fait par une stratégie de raffinements successifs. Le contrôle d'exécution du plan offre un

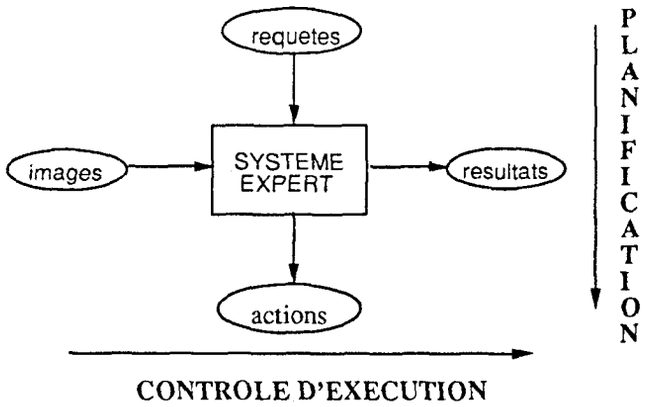


Fig. 2. — Deux types de raisonnement.

mécanisme d'essai-erreur. Après exécution d'un opérateur, la qualité des résultats obtenus peut être testée (évaluation des résultats) et en cas de résultats non satisfaisants, un mécanisme adapte les valeurs de paramètres des algorithmes de traitement d'images (ajustement des paramètres). Dans le cas d'un opérateur complexe, les paramètres ajustés sont utilisés dans les sous-buts des requêtes de sa décomposition. La séquence d'opérateurs peut également être modifiée par replanification si nécessaire. Les mécanismes de choix, d'évaluation, d'initialisation et d'ajustement fonctionnent chacun en plusieurs phases qui peuvent être automatiques et/ou interactives.

L'algorithme présenté (*fig.* 3) contrôle l'exécution de requêtes. Bien sûr, pour traiter la requête initiale, cet algorithme fonctionne sur une seule requête.

-1- mise en correspondance globale

tant qu'il y a des requêtes à traiter

-2- sélection de la requête à traiter

-3- classement des opérateurs (en utilisant les règles de choix)

tant que la requête n'est pas satisfaite

-4- sélection du meilleur opérateur

-5- exécution de l'opérateur (en utilisant les règles d'initialistion ou d'ajustement)

-6- évaluation des résultats obtenus (en utilisant les règles d'évaluation)

Fig. 3. — L'algorithme d'exécution des requêtes.

L'étape 1 est une phase de mise en correspondance globale qui vérifie qu'il existe au moins un opérateur susceptible de répondre à chacune des requêtes. L'étape 2 sélectionne une requête à traiter (quand il s'agit de la requête initiale, cette étape est immédiate). L'étape 3 élimine les opérateurs non valides et classe les opérateurs valides en utilisant les règles de choix. L'étape 4 sélectionne un opérateur parmi les opérateurs valides. Avant l'exécution de l'opérateur choisi (étape 5), les valeurs de ses paramètres sont déterminées en utilisant les règles d'initialisation lors du premier essai, et en utilisant les règles d'ajustement lors des essais suivants. Si l'opérateur est élémentaire, l'exécution est lancée ; si l'opérateur est complexe, cet algorithme est appelé récursivement pour contrôler l'exécution des requêtes de sa décomposition. Puis selon la requête, l'évaluation des résultats (étape 6) est effectuée automatiquement en utilisant les règles d'évaluation ou par présentation graphique des résultats à l'utilisateur qui les évalue interactivement. En cas de résultats non satisfaisants, une autre exécution est réalisée après l'ajustement des paramètres de l'opérateur ou le choix d'un autre opérateur.

2.3. CONCLUSION

OCAPI est un noyau de systèmes experts spécialisé dans le pilotage de traitement d'images : c'est-à-dire qu'il offre des structures de représentation des connaissances ainsi

que des mécanismes de raisonnement particulièrement adaptés au traitement d'images. Les structures de représentation des connaissances sont implémentées sous deux formes : objets et règles de production. Les notions de but, requête, opérateur (élémentaire et complexe), contexte, et arguments (données et paramètres) sont représentées sous forme d'objets ; les notions de choix, initialisation de paramètres, évaluation de résultats et ajustement de paramètres sont représentées sous forme de règles de production. Quant au raisonnement, il repose sur deux tâches principales : la planification et le contrôle d'exécution. Ces tâches utilisent l'ensemble des connaissances exprimées dans la base de connaissance.

Lors du développement d'un système expert, seules des connaissances du domaine d'application et de traitement d'images sont à expliciter : les structures d'expression de cette connaissance sont adaptées et les mécanismes généraux du raisonnement sont déjà implémentés.

3. Application à la description morphologique de galaxies

Nous nous intéressons ici à l'automatisation du traitement d'images contenant une galaxie, dans le but de classer les galaxies par rapport à une classification donnée. En effet, l'amélioration des instruments et l'utilisation d'un large ensemble de télescopes dans de nombreuses longueurs d'ondes (rayons X, ultraviolet, visible, infra-rouge, radio) nous permettent de disposer de données de plus en plus nombreuses ; seule l'automatisation de leurs traitements peut permettre une exploitation réelle de ces données [TB 89]. L'architecture générale du système complet est présentée (*fig.* 4). Deux phases sont actuellement nécessaires pour résoudre cette application. La première phase réalise, à partir de l'image, l'extraction de paramètres numériques qui sont des mesures décrivant la morphologie de la galaxie. Elle est implémentée par le système PROGAL qui met en œuvre les algorithmes de traitement d'images ; ce système a été réalisé avec le noyau de systèmes experts OCAPI présenté dans la partie précédente. La seconde phase classe la galaxie décrite par ses paramètres numériques en s'appuyant sur une taxonomie élaborée par des experts du domaine ; elle fournit en résultat outre le nom de la classe à laquelle la galaxie appartient, une description symbolique détaillée de la galaxie ; elle est implémentée par le système SYGAL [Tho 89, TB 89] qui a été développé avec le noyau de systèmes experts CLASSIC [Ilo 87] spécialisé en classement.

Dans cette application, le rôle de la phase de traitement d'images est de décrire la morphologie de la galaxie par des paramètres numériques caractéristiques (voir [Tho 89] pour une présentation complète des procédures de traitement d'images). Un exemple d'une telle description est montré (*fig.* 5). Certains de ces paramètres sont des mesures concernant l'orientation, l'élongation et la compacité de la galaxie et de différentes régions de la galaxie délimitées par des courbes d'iso-intensité ; d'autres repré-

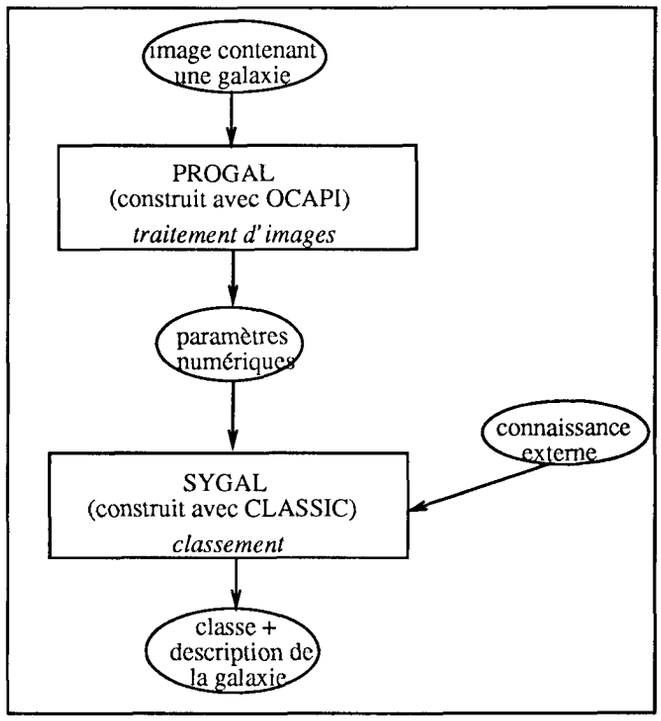


Fig. 4. — Architecture générale.

```
galaxie :   uVclt :
            surface          : 67886.7 ;   elongation    : 0.35 ;
            errlineaire      : 0.034 ;     profil        : 1.96 ;
            orientation      : 86.46 ;
            contour c1 : {   errcentre     : 0 ;
                             errellipse    : 0.19 ;
                             compacite     : 1.8 ;
                             orientation   : -9.5 ;
                             elongation    : 0.22 };
            contour c2 : {   errcentre     : 7 ;
                             errellipse    : 0.37 ;
                             compacite     : 8.1 ;
                             orientation   : -4.4 ;
                             elongation    : 0.34 };
            contour c3 : {   errcentre     : 3 ;
                             errellipse    : 0.30 ;
                             compacite     : 14.1 ;
                             orientation   : 50.4 ;
                             elongation    : 0.40 };
            contour c4 : {   errcentre     : 12 ;
                             errellipse    : 0.25 ;
                             compacite     : 25.7 ;
                             orientation   : 55.4 ;
                             elongation    : 0.31 };
            contour c5 : {   errcentre     : 20 ;
                             errellipse    : 0.48 ;
                             compacite     : 38.6 ;
                             orientation   : 84.2 ;
                             elongation    : 0.54 }.
```

Fig. 5. — Exemples de paramètres à extraire.

sentent des erreurs par rapport à des modèles photométriques théoriques (errlinéaire, profil) ou par rapport à des modèles géométriques (errellipse). Le traitement à effectuer sur chacune des images est globalement le même. Il se décompose en 5 étapes principales : une phase d'initialisation (création et initialisation du fichier contenant les paramètres), l'extraction dans l'image de la galaxie (détermination des limites de la galaxie et suppression du fond), le calcul des paramètres globaux de la galaxie, la construction d'images de contours d'iso-intensité (contours des régions obtenues pour différentes valeurs de seuillage de l'image de la galaxie), et, pour chacun des contours, le calcul de paramètres le décrivant. Parmi ces étapes, certaines sont relativement complexes. Par exemple, l'extraction de la galaxie a trois sous-étapes : sa détection, son isolation effective et un débruitage.

A cause de la diversité des images à traiter qui proviennent d'instruments d'observation différents (plaques photographiques ou CCD, basse ou haute résolution) et dont quelques exemples sont montrés (*fig.* 6), la même méthode de traitement ne peut être appliquée à l'ensemble de ces images : la séquence des algorithmes et les valeurs de leurs paramètres ont besoin d'être adaptées. L'ensemble des algorithmes utilisés ici est composé de procédures de vision d'une bibliothèque standard (INRIMAGE [Cip 84]), de procédures développées spécifiquement pour le traitement de ces images (en C et Fortran 77) et de fonctions de stockage des paramètres extraits écrites en Le_Lisp. Les algorithmes standards comprennent le calcul d'histogramme, des filtrages (médian, morphologique), l'extraction de sous-images ou des convolutions. Les algorithmes spécialisés concernent par exemple des déterminations de seuils, l'isolation de la galaxie et les calculs des paramètres morphologiques.

3.1. BASE DE CONNAISSANCE

Dans cette partie, nous présentons la base de connaissance du système PROGAL. Les exemples présentés sont exprimés dans le formalisme d'OCAPI, c'est-à-dire à l'aide d'objets et de règles de production. Comme nous l'avons vu, une base de connaissance pour OCAPI est structurée par des objets de différents types. D'autre part, les heuristiques concernant le choix entre opérateurs, l'évaluation des résultats intermédiaires ou finaux, l'initialisation et l'ajustement des paramètres sont exprimées à l'aide de règles. Nous détaillons ici quelques exemples d'objets importants et les différents types de règles présents dans PROGAL.

Contexte

L'objet **contxte** décrit les valeurs possibles des caractéristiques des données d'entrée ; cette information est importante car elle est utilisée dans les règles (règles de choix pour décider quels sont les opérateurs pertinents en fonction du contexte, règles d'initialisation pour adapter les valeurs des paramètres au contexte courant...). Actuellement, nous utilisons dans les règles de production deux types de critères : les critères relatifs à l'instrument d'observation tels que le niveau de bruit dans l'image d'entrée, le type de l'image (densité ou intensité) et la taille maximale des objets pouvant occulter partiellement l'objet d'intérêt (ici la galaxie), et les critères concernant le contenu sémantique de l'image tels que le positionnement dans l'image de l'objet d'intérêt, la présence éventuelle d'autres objets (étoiles brillantes) et leur répartition (table 1).

Buts

Cette base de connaissance contient différents buts tels que la détection et l'extraction grossière de la zone d'un objet (*détection-zone-objet, extraction-zone-objet*), la détection et l'extraction précise d'un objet (*détection-objet, extraction-objet*) ou le calcul de paramètres globaux sur un objet (*paramètres-globaux*)... Par exemple, le but *extraction-zone-objet* (*fig.* 7) a un argument d'entrée (l'image originale d'entrée) et 5 arguments de sortie (les positions en x et y de l'objet, la sous-image contenant l'objet ainsi que ses tailles en x et y). Des règles de choix sont attachées à ce but (elles sont commentées dans la prochaine section) ; mais il n'y a pas ici de règles d'évaluation.

Opérateurs

Différents opérateurs ont été décrits dans la base de connaissance, chacun d'entre eux correspondant à un but particulier comme nous le montrons dans la table 2. Par exemple, l'opérateur *O-détection-zone-objet-1* qui résout le but *détection-zone-objet* (*fig.* 8) a les mêmes données d'entrée que son but (l'image initiale) et les mêmes données de sortie (les positions et tailles en x et y de la sous-image englobant l'objet détecté ainsi que la taille de l'objet). Pour détecter la zone de la galaxie, cet opérateur complexe cherche un objet brillant étendu. Il est décomposé en plusieurs sous-étapes (requêtes à d'autres buts) : il effectue tout d'abord un seuillage, puis un filtrage, puis il construit la chaîne de contour de la région la plus grande et enfin calcule les positions et taille de la boîte englobant cette région. Cet opérateur est utilisé au cours de l'extraction de la zone de l'objet dans le cas où on n'a pas de connaissance a priori sur le positionnement de l'objet dans l'image (c'est-à-dire si l'objet n'est pas centré dans l'image ou a une position inconnue). Des règles d'initialisation et d'ajustement permettent de déterminer la valeur du seuil *smuls* à chacune des exécutions de cet opérateur.

Requêtes

Chaque étape de la décomposition d'un opérateur complexe est une requête, appelée sous-requête. Le contrôle des flux de données d'entrée et de sortie de l'opérateur et des sous-requêtes est exprimé directement dans les requêtes et les arguments de l'opérateur. Nous montrons en détail (*fig.* 9) une requête qui spécifie des contraintes sur les résultats du traitement ; les restrictions expriment que la spécification minimale *specifs-min* de la taille de l'objet détecté est *taille-max-étoiles* qui est une information contenue dans le contexte.

Règles de choix

Quand plusieurs opérateurs sont disponibles pour résoudre un même but, il est nécessaire d'exprimer comment sélectionner l'opérateur pertinent en fonction du contexte.

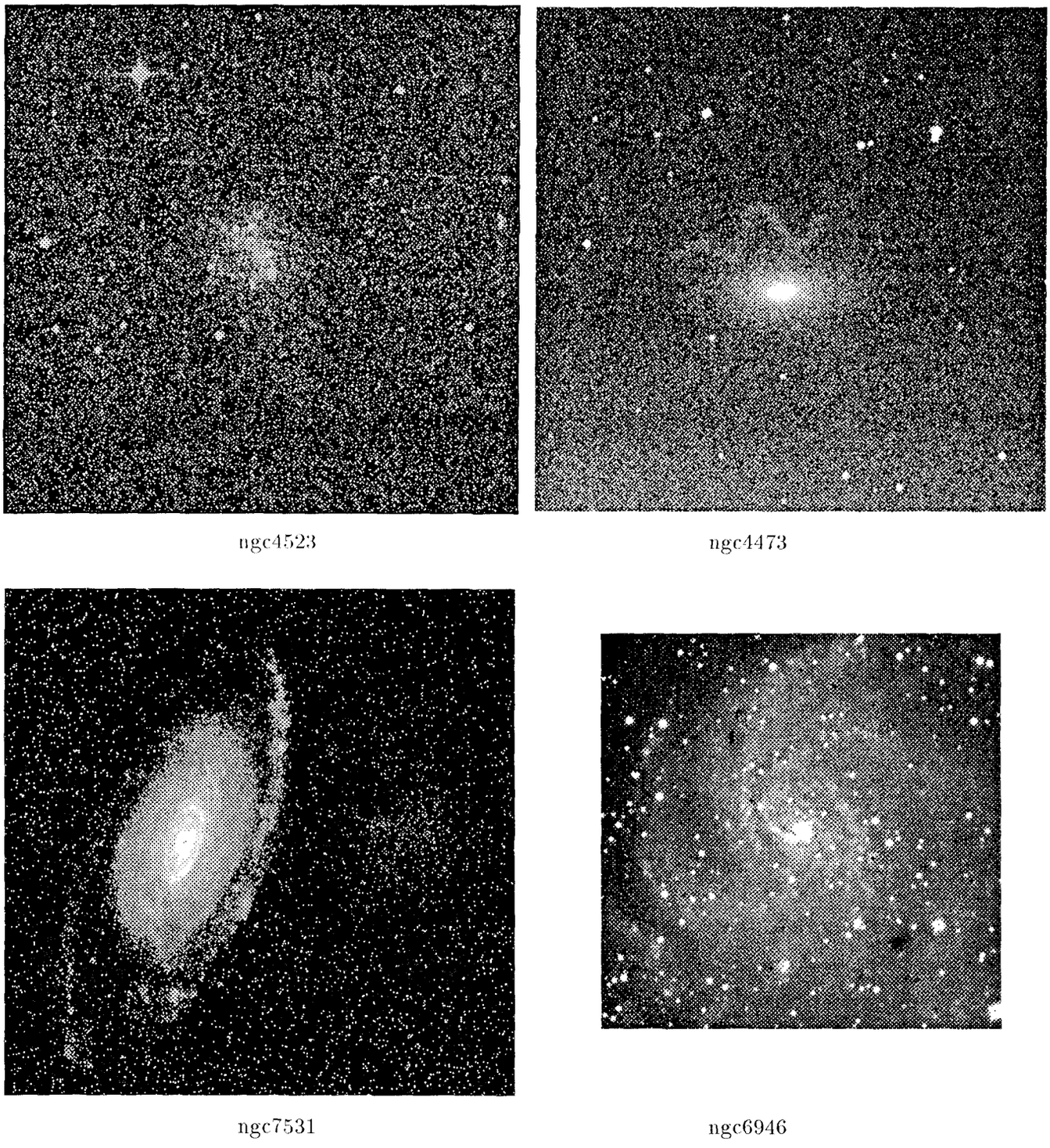


Fig. 6. — Quelques images de galaxies.

Table 1. — Définition du contexte

INFORMATION DE CONTEXTE	VALEURS POSSIBLES	COMMENTAIRES
bruit	*faible, fort*	image bruitée ou non
type-image	*intensité, densité*	calibrage photométrique
taille-max-étoiles	[1, 1000]	taille maximum des éventuelles étoiles exprimée en pixels (fonction de l'observation)
position-objet	*centrée, non-centrée, inconnue*	positionnement de l'objet dans l'image
étoiles	*sur_galaxie, en_dehors_de_galaxie, absentes, partout*	présence éventuelle d'étoiles

But extraction-zone-objet
données d'entrée : image
données de sortie : pos-x, pos-y, taille-x, taille-y, zone
règles de choix : { si position-objet centrée
alors :utiliser-opérateur O-extraction-zone-objet-centré ... }
règles d'évaluation : { }

Fig. 7. — Le but *extraction-zone-objet.*

Par exemple, comme nous avons vu dans la section précédente (table 2), deux opérateurs *O-extraction-zone-objet-centré* et *O-extraction-zone-objet-général* peuvent résoudre le but *extaction-zone-objet*. Ces deux opérateurs réalisent l'extraction d'une sous-image contenant l'objet recherché ; le premier simplement l'extraction d'une sous-image centrée dans l'image originale en supposant que l'objet est positionné au centre de l'image ; le second cherche tout d'abord à détecter la zone de l'objet avant d'extraire la sous-image correspondante (c'est sur cette étape de détection de la zone de l'objet que nous illustrerons les règles d'initialisation, d'ajustement et d'évaluation). Une des règles de choix attachées au but *extraction-zone-objet* est montrée (*fig.* 7) ; elle traite le cas particulier où l'on a de la connaissance a priori sur le positionnement de l'objet. D'autres règles existent pour adapter le choix aux scènes pour lesquelles on n'a pas d'information a priori, ou si l'objet est décentré dans l'image.

Règles d'évaluation

La plupart du temps il est très difficile d'exprimer des critères d'évaluation sur les résultats des buts de bas niveau (comme le seuillage par exemple) puisqu'il ne peut y avoir aucune interprétation associée. Par contre, à un niveau d'abstraction plus élevé, il est possible de prendre en compte des mesures, par exemple. Deux règles d'évaluation associées au but *détection-zone-objet* sont

Table 2. — Quelques buts et les opérateurs qui leurs correspondent, extraits de la base PROGAL

but	opérateurs
extraction-zone-objet	O-extraction-zone-objet-centré, O-extraction-zone-objet-général
détection-zone-objet	O-détection-zone-objet-1
isolation-objet	O-isolation
filtrage	O-f-morpho, O-f-médian
...	...

```
But                    détection-zone-objet
données d'entrée :     image
données de sortie :    ix, iy, x, y, taille
règles de choix :      { }
règles d'évaluation :  { si :juger taille trop-petite/specifs et non étoiles absentes
                           alors :juger détection ambigüe et :échec
                         si détection non ambigüe et taille < specifs-min + 2
                           alors :juger détection limite ... }

Opérateur              O-détection-zone-objet-1
données d'entrée :     image "image initiale"
paramètres :           smuls
données de sortie :    ix "position en x"
                       iy "position en y"
                       x "taille en x"
                       y "taille en y"
                       taille "taille de l'objet"
décomposition :            requête r-seuille
                       puis requête r-filtre
                       puis requête r-construit-chaine-contour
                       puis requête r-calcule-boite-englobante
règles d'initialisation : { si sm < saire2
                           alors :initialiser 'smuls (saire1 + (saire2 - saire1)/3) ... }
règles d'ajustement :  { si ...
                           alors :ajuster-par '% 'smuls et :%-coeff 'smuls 0.05 et :%-min 'smuls 'saire2
                         si détection ambigüe
                           alors :diminuer smuls ... }
```

Fig. 8. — Le but *détection-zone-objet* et un de ses opérateurs complexes associés *O-détection-zone-objet-1.*

```
Requête        r-do       but     détection-zone-objet
restrictions : entrées :  image   = image de OPERATEUR O-extraction-zone-objet-général
               sorties :  taille  > (:contexte taille-max-étoiles)
```

Fig. 9. — Description d'une sous-requête de l'opérateur *O-extraction-zone-objet-général.*

montrées (*fig.* 8). La première exprime que si la taille de l'objet détecté est trop petite par rapport aux spécifications demandées dans la requête alors qu'il y a des objets potentiels d'occultation (les étoiles ne sont pas absentes a priori), la détection est donc ambiguë et un échec est provoqué ; cet échec va entraîner un bouclage, c'est-à-dire ue nouvelle exécution de l'opérateur avec des valeurs de paramètres modifiées. Un autre type de règle est montré dans la seconde règle d'évaluation ; cette règle permet simplement d'émettre un jugement (*si la détection n'est pas ambiguë, mais que la taille de l'objet est très proche du seuil d'ambiguïté [appelé specifs-min] la détection est limite*). Ce jugement pourra être utilisé dans la suite du raisonnement, mais sans provoquer d'échec immédiatement.

Règles d'initialisation

L'initialisation des paramètres dépend fortement du contexte. Par exemple, l'opérateur *O-isolation-objet*, qui réalise l'isolation effective de la galaxie en calculant sa frontière exacte, a besoin d'initialiser deux paramètres : *sf1* et *sf2*. Ces paramètres sont deux seuils et leurs valeurs dépendent du type de l'image (image de densité ou d'intensité. Une de ces règles permettant l'initialisation d'un paramètre en fonction du contexte est :

> **si** type-image densité **alors** :initialiser 'sf1 0.1
> **et** :initialiser 'sf2 1

D'autre part, l'opérateur *O-détection-zone-objet-1* a besoin d'initialiser la valeur du seuil *smuls*. Cet argument d'entrée est défini comme un paramètre qui est une valeur numérique comprise entre 0 et 1 et qui n'a pas de valeur fixe par défaut. Sa valeur dépend des valeurs d'autres variables (*sm, saire1, saire2*) décrivant la forme de l'histogramme. Aussi des règles d'initialisation sont nécessaires pour définir sa valeur en fonction du contexte d'utilisation de l'opérateur. Par exemple, une règle d'initialisation du paramètre *smuls* est montrée (*fig.* 8). Si *sm* est inférieur à *saire2, smuls* est calculé par une formule dépendante de *saire1* et *saire2*. Ces règles sont déclenchées lors de la première exécution de l'opérateur (pour une requête donnée au but *détection-zone-objet*).

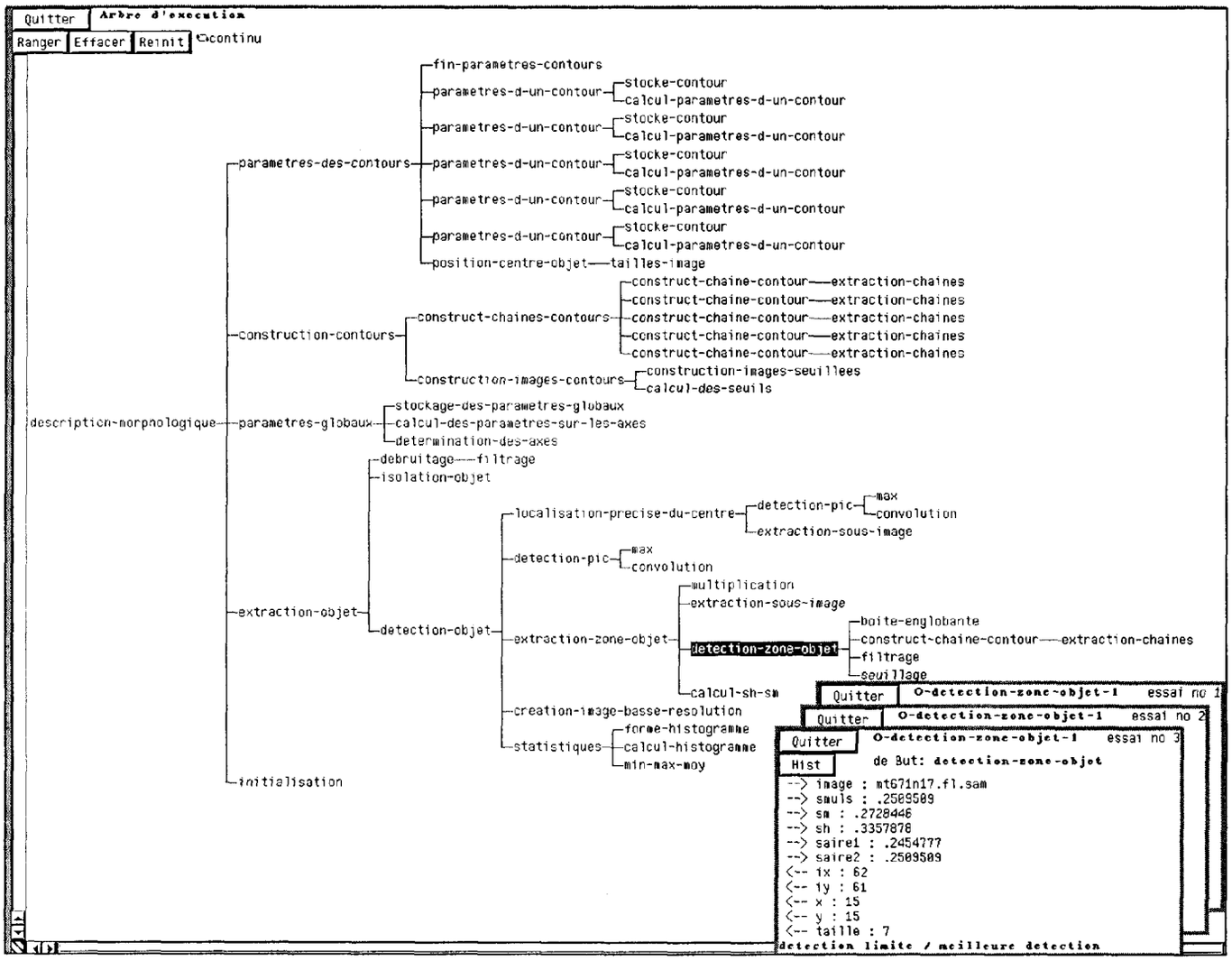
Règles d'ajustement

Quand un opérateur échoue à cause de résultats jugés non satisfaisants, il est nécessaire d'exprimer comment modifier ses paramètres en fonction du type de l'échec. Il existe trois types de règles d'ajustement : celles qui permettent de choisir une méthode d'ajustement pour un paramètre donné (*par pourcentage, par dichotomie...*), celles qui déterminent un sens de modification de la valeur pour la prochaine exécution (*augmenter, diminuer*) et celles qui fixent de manière impérative une nouvelle valeur. Toutes ces règles d'ajustement s'appuyent sur les jugements qui ont été formulés par les règles d'évaluation ; c'est en fonction de ces diagnotics que la manière d'ajuster le paramètre est déterminée. La première des règles d'ajustement présentée (*fig.* 8) permet de choisir une méthode d'ajustement pour la paramètre *smuls* ; il s'agit de la méthode par pourcentage avec un coefficient de 5 %, sachant que ce paramètre doit avoir pour borne minimum la valeur du paramètre *saire2*. La règle suivante exprime qu'en cas de détection ambiguë, le seuil *smuls* doit être diminué. Il le sera en fonction de la méthode précédemment choisie.

3.2 SESSION

Nous illustrons ici le fonctionnement du système en présentant une exécution effective sur une image représentant la galaxie ngc4523 de l'amas de Virgo (*fig.* 6). Nous demandons au système l'exécution d'une requête dont le but est *description-morphologique* : l'argument d'entrée *mt671n17.fl* est le nom de fichier de l'image contenant la galaxe ngc4523. Dans ce cas particulier, l'image est une image de densité, elle est bruitée, la position de l'objet dans l'image est inconnue, il y a des étoiles partout dans l'image et elles sont de taille inférieure à 5 pixels.

Au cours de l'exécution, le système construit dynamiquement l'arbre des sous-buts qu'il résoud (*fig.* 10). L'arbre représente la décomposition du traitement final en sous-étapes jusqu'à des opérateurs élémentaires (programmes).



De gauche à droite : de la requête initiale vers les programmes concrets.

De bas en haut : de la première sous-étape à la dernière.

Fig. 10. — Arbre d'exécution complet.

Le niveau d'abstraction est représenté sur l'axe horizontal : du plus abstrait à gauche au moins abstrait (les programmes concrets) à droite. L'ordonnancement temporel est représenté sur l'axe vertical : le nœud le plus bas correspond à la première étape, le plus haut à la dernière. Dans cette figure, les nœuds sont des buts résolus. Après l'initialisation (création d'un fichier pour stocker les paramètres), a lieu la phase d'extraction de l'objet qui commence par une détection de l'objet. Pour cette détection, le système calcule les statistiques de densité, crée une image échantillonnée (image originale : 512 × 512 ; facteur d'échantillonnage : 4 × 4), puis il cherche à extraire la zone de la galaxie dans cette image à basse résolution. Pour extraire effectivement la sous-image correspondant à cette zone, une étape cherche à détecter grossièrement la galaxie (but *détection-zone-objet* figurant en noir sur (*fig.* 10)).

Les résultats de la première exécution de l'opérateur *O-détection-zone-objet-1* pour résoudre le but *détection-zone-objet* sont montrés dans la figure 11. Il faut remarquer que c'est l'étoile en haut à gauche qui a été détectée et non la galaxie : en effet, la position de l'objet détecté est rendue dans les arguments *ix* et *iy* qui valent respectivement, après cette exécution, 32 et 16 ; l'image seuillée est une image 128 × 128, calculée à partir de l'image échantillonnée. La taille de l'objet détecté (*taille* = 5) étant trop

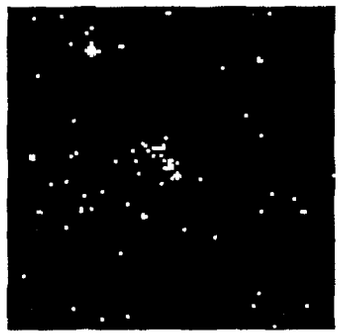


image seuillée

objet détecté

```
Quitter   O-detection-zone-objet-1    essai no 1
Hist      de But: detection-zone-objet
--> image : mt671n17.fl.sam
--> smuls : .2728446
--> sm : .2728446
--> sh : .3357878
--> saire1 : .2454777
--> saire2 : .2509509
<-- ix : 32
<-- iy : 16
<-- x : 15
<-- y : 15
<-- taille : 5 trop-petit/specifs
detection ambigue
```

Fig. 11. — Premier essai.

faible, la première règle d'évaluation présentée (*fig.* 8) est activée : un bouclage va être effectué.

Après déclenchement des règles d'ajustement de l'opérateur *O-détection-zone-objet-1*, le paramètre *smuls* a été diminué par une méthode de pourcentage et sa valeur est 0.2592023 pour cette deuxième exécution. Les résultats sont montrés dans la figure 12. La taille de l'objet détecté est toujours trop petite. Toutefois, c'est bien la galaxie qui a été détectée : la position de l'objet détecté est rendue dans les arguments *ix* et *iy* qui valent respectivement 64 et 62. Sans autre critère d'évaluation pour ce but que la taille de l'objet, le système boucle une nouvelle fois.

Un troisième essai a donc lieu avec la valeur 0.2509509 pour *smuls*. Les résultats de la troisième exécution sont montrés dans la figure 13. C'est toujours la galaxie qui a été détectée (en position 62/61) et la taille a augmenté.

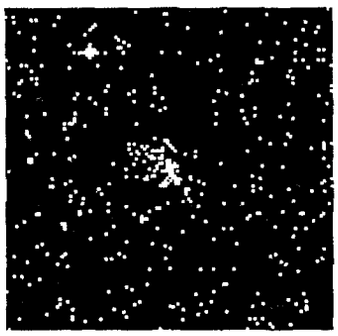


image seuillée

objet détecté

```
Quitter   O-detection-zone-objet-1    essai no 2
Hist      de But: detection-zone-objet
--> image : mt671n17.fl.sam
--> smuls : .2592023
--> sm : .2728446
--> sh : .3357878
--> saire1 : .2454777
--> saire2 : .2509509
<-- ix : 64
<-- iy : 62
<-- x : 15
<-- y : 15
<-- taille : 5 trop-petit/specifs
detection ambigue
```

Fig. 12. — Deuxième essai.

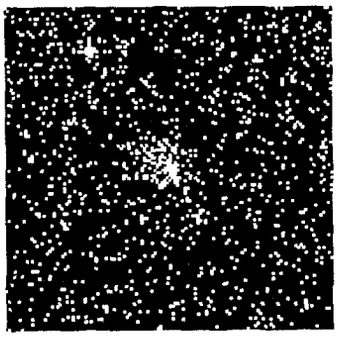


image seuillée

objet détecté

```
Quitter   O-detection-zone-objet-1    essai no 3
Hist      de But: detection-zone-objet
--> image : mt671n17.fl.sam
--> smuls : .2509509
--> sm : .2728446
--> sh : .3357878
--> saire1 : .2454777
--> saire2 : .2509509
<-- ix : 62
<-- iy : 61
<-- x : 15
<-- y : 15
<-- taille : 7
detection limite / meilleure detection
```

Fig. 13. — Troisième essai.

Comme la taille (*7*) est supérieure à la taille maximale d'une étoile, aucun bouclage supplémentaire n'est nécessaire. D'autre part, le paramètre *smuls* ayant atteint sa valeur minimum, au-delà de laquelle trop de bruit de fond est conservé, il est impossible de faire une meilleure détection.

Le raisonnement continue comme montré dans la figure 10 : les étapes suivantes de l'extraction grossière de la zone de l'objet sont l'extraction de la sous-image et la multiplication de cette sous-image avec l'image binaire seuillée ; ensuite le bulbe de la galaxie est estimé en maximisant la corrélation avec un modèle gaussien, puis cette estimation est précisée en travaillant à plus haute résolution. L'extraction de l'objet continue avec l'isolation effective de l'objet qui calcule ses limites en utilisant un modèle de variations photométriques de l'intensité à partir du bulbe, et se termine par une étape de débruitage. Les autres étapes *paramètres-globaux, construction-contours* et *paramètres-des-contours* calculent des paramètres globaux de forme (*surface, orientation, élongation*) et deux paramètres photométriques (*profil, errlinéaire*) de la galaxie, ainsi que la forme et la photométrie des contours de régions. Pour plus de détails voir [Tho 89].

Dans l'exécution présentée (*fig.* 10), concernant la résolution de la requête au but *description-morphologique* sur l'image *mt671n17.fl*, les règles de choix et d'évaluation ont utilisé les informations concernant la position de la galaxie dans l'image, la présence éventuelle de bruit et d'étoiles ainsi que le type de l'image. Comme ces informations n'avaient pas été précisées au système expert avant l'exécution, elles ont été demandées interactivement à l'utilisateur au fur et à mesure. A chaque nœud de l'arbre de la figure 10, correspond un ou plusieurs sous-arbres : les essais successifs pour une même requête. Ici, par exemple, pour le nœud correspondant au but *détection-zone-objet*, nous avons vu que trois sous-arbres ont été exécutés (une partie de l'historique est visible dans le bas de la figure). A chaque feuille, correspond l'exécution d'un opérateur élémentaire c'est-à-dire un programme. Cette exécution déclenche effectivement 49 exécutions de programmes : 40 sont des programmes de traitement d'images, aussi variés que calcul d'histogramme ou filtrages, parmi lesquels 4 sont exécutés chacun 3 fois lors de phases évaluation-ajustement ; 9 sont des calculs simples ou des fonctions de stockage. Les résultats fournis sont directement exploitables par le système de classification de galaxies SYGAL.

Sur les images des galaxies ngc4473 et ngc7531 présentées dans la figure 6, le système expert ne boucle pas ; les résultats obtenus sont satisfaisants après une seule exécution de chacun des opérateurs initialement choisis, bien que, d'une part, la galaxie ngc4473 soit centrée dans l'image, que celle-ci soit bruitée et que des étoiles soient présentes, et que, d'autre part, la galaxie ngc7531 ne soit pas centrée dans l'image, que l'image ne soit pas bruitée et qu'il n'y ait pas d'étoiles. Par contre sur l'image de la galaxie ngc6946 (*fig.* 6), l'étoile du coin inférieur droit est considérée comme l'objet d'intérêt dans une première étape, puis remise en cause ; la galaxie est alors détectée après ajustement de paramètres.

4. Conclusions

Nous nous sommes intéressées à l'intégration sémantique de programmes de traitement d'images pour permettre le développement d'applications robustes, automatiques et adaptables. Nous avons décrit OCAPI qui est un noyau de systèmes experts que nous avons développé pour réaliser des intégrations sémantiques de programmes. Puis nous avons présenté le système expert PROGAL, développé à l'aide d'OCAPI, dont le but est la description morphologique de galaxies. La base de connaissance de ce système a été présentée à l'aide de nombreux exemples et une exécution en a été commentée.

Cette présentation a permis de mettre en évidence le type de connaissance qu'un système intégré peut prendre en compte pour permettre le développement d'applications flexibles et accessibles à des non spécialistes (connaissance sur les traitements, leur utilisation, leurs enchaînements, l'optimisation des valeurs de leurs paramètres et l'évaluation de leurs résultats en fonction du contexte). La base de connaissance de PROGAL comprend actuellement 44 buts qui peuvent être résolus par 47 opérateurs. Parmi ces opérateurs, 28 sont élémentaires (programmes) et 19 sont complexes (décomposition en étapes). A ces buts et opérateurs sont attachées 21 règles. Ce système permet de traiter des images de provenances variées (capteurs différents, positionnement quelconque dans l'image de l'objet à mesurer) et de diverses qualités tout en assurant la qualité des résultats obtenus.

Outre une meilleure automatisation des traitements qui est son objectif premier, la méthodologie proposée a un réel impact sur le développement des applications du point de vue génie logiciel. En effet, un programme ou une méthode développé pour une application de traitement d'images réalise souvent plusieurs fonctions sans liens sémantiques entre elles, ou fait des hypothèses fortes sur son contexte d'utilisation ou sur les résultats des programmes qui le précède. Lors du développement d'un système intégré, l'expert prend conscience des nombreuses connaissances implicites dans les programmes utilisés ; en les exprimant dans la base de connaissance il généralise ainsi ses méthodes par explication de choix qui étaient effectués dans le corps des programmes, ou ajout de mécanismes d'évaluation-ajustement. Une caractéristique essentielle d'un système comme OCAPI est qu'il permet d'exprimer ces connaissances à différents niveaux d'abstraction.

Le système de classement de galaxies présenté (*fig.* 4) est actuellement constitué de deux étapes séquentielles : PROGAL (description morphologique) suivi de SYGAL (classification de galaxies à partir de leur description morphologique). Dans une prochaine extension, SYGAL sera intégré dans PROGAL comme un opérateur élémentaire (programme) bien qu'il soit lui-même un système expert. D'autre part, des mesures complémentaires sur l'image peuvent être nécessaires en cours du raisonnement de classification (une fois la classe de la galaxie déterminée grossièrement, par exemple) ; dans ce cas, il est possible de demander au système de traitement d'images PROGAL la résolution de requêtes spécifiques pour réaliser ces

mesures. Il s'agit donc ici d'un exemple de coopération entre deux systèmes experts de raisonnements différents (planification/contrôle d'exécution d'une part, classement d'autre part) dans le cadre de l'interprétation d'images.

Cette méthodologie générale est également utilisée pour d'autres applications, et plusieurs bases de connaissance de traitement d'images sont en cours de développement : en stéréovision sur la méthode présentée dans [MTB 90] pour la détection d'obstacles dans le cadre de PROMETHEUS, et en télédétection sur les travaux de [Ser 89] pour l'extraction des agglomérations, réseaux routiers et fluviaux. La méthodologie proposée ici est également suffisamment générale pour être utilisée au-delà du domaine du traitement d'images ; elle est actuellement en cours d'évaluation pour son utilisation pour le pilotage d'une bibliothèque de programmes d'astrophysiques spécialisée en modélisation de structure interne d'étoiles.

Manuscrit reçu le 27 juillet 1991.

BIBLIOGRAPHIE

[Bro 81] R. Brooks, Symbolic Reasoning Among 3-D Models and 2-D Images, *Artificial Intelligence Journal 17 :285-348*, 1981.

[Cip 84] P. Cipière, INRIMAGE : Manuel de l'utilisateur, INRIA, France 1984.

[Cle 90] V. Clément, *Raisonnements cognitifs appliqués au pilotage d'algorithmes de traitement d'images*, Thèse de doctorat de l'université de Nice, janvier 1990.

[CT 90] V. Clément et M. Thonnat, *Integration of image processing procedures : OCAPI, a knowledge-based approach* Rapport de Recherche INRIA n° 1307, octobre 1990.

[IK 88] K. Ikeuchi et T. Kanade, Automatic Generation of Object Recognition Programs, *Proceedings of the IEEE* **78.8 :1016-1035** 1988.

[Ilo 87] ILOG, CLASSIC Manuel de l'utilisateur, Paris, 1987.

[Joh 87] M. D. Johnston, An expert system approach to astronomical data analysis, *Proc. of Goddard Conf. on Space Applications of Artificial Intelligence and Robotics*, 1987, pp. 1-17.

[NL 84] A. M. Nazif et M. D. Levine, Low Level image Segmentation : An Expert System, *IEEE Transactions on pattern Analysis and Machine Intelligence 6.5 :555-577*, 1984.

[Mat 89] T. Matsuyama, Expert systems for image processing : Knowledged-based composition of image analysis processes, *Comput. Vision Graphics Image Process.* **48**, 1989, 22-49.

[MH 90] T. Matsuyama et V. Hwang, *SIGMA : a knowledge-based aerial image understanding system*, Advances in computer vision and machine intelligence, Plenum, 1990.

[MTB 90] A. Meygret, M. Thonnat et M. Berthod, A pyramidal stereovision algorithm based on contour chain points, *Lecture Notes in Computer Science* **427** (O. D. Faugeras, Ed.), pp. 83-88, Springer-Verlag, 1990.

[ST 85] S. Sakaue et H. Tamura, Automatic generation of image processing programs by knowledge-based verification, *Proc. of IEEE Computer Vision and Pattern Recognition*, San Francisco, 1985, pp. 189-192.

[SKT 88] H. Sato, Y. Kitamura et H. Tamura, A knowledge-based approach to vision algorithm design for industrial parts feeder, *Proc. of IAPR Workshop on Computer Vision, Special hardware and industrial applications*, Tokyo, 1988, pp. 413-416.

[Ser 89] M. A. Serendero, *Extraction d'informatique symbolique en imagerie SPOT : réseaux de commmunication et agglomérations*, Thèse de doctorat de l'université de Nice, décembre 1989.

[TS 84] H. Tamura et K. Sakaue, DIA (Digital Image Analyse) — Expert System : an approach to future vision system design, *Int. Symp. on Image Processing and its Applications*, Tokyo, 1984.

[TF 88] M. Thonnat et M. H. Gandelin, An expert system for the automatic classification and description of zooplanktons from monocular images, *9th Int. Conf. on Pattern Recognition* Roma, Italy, 1988.

[Tho 89] M. Thonnat, Toward an automatic classification of galaxies, *The world of galaxies*, pp. 53-74, Springer-Verlag, 1989.

[TB 89] M. Thonnat et A. Bijaoui, Knowledge-Based Classification of Galaxies, *Knowledge-Based Systems in Astronomy* (F. Murtagh and A. Heck Eds.), pp. 121-159, Springer-Verlag, 1989.

[TIY 87] T. Toriu, H. Iwase et M. Yoshida, An Expert System for Image Processing, *FUJITSU Sci. Tech. J.* **23.2**, 1987, 111-118.